

R:BASE eXtreme 9.0



Command Index

**The Programmer's Guide to Building
R:BASE Databases and Custom Applications**

R:BASE Technologies, Inc.



R:BASE eXtreme 9.0

Command Index

by R:BASE Technologies, Inc.

Welcome to R:BASE eXtreme 9.0

R:BASE eXtreme 9.0 for Windows is a completely new relational database development environment that has been amplified to include the latest menu-driven features to database professionals. With the R:BASE engine established as the foundation of stability, the program interface and designers have been enhanced to fully exhibit a higher quality of development usability.

The "eXtreme" in the product's name signifies the cutting-edge development interface incorporated into the R:BASE program making database and application development more productive.

R:BASE eXtreme 9.0 Command Index

Copyright © 1982-2009 R:BASE Technologies, Inc.

Information in this document, including URL and other Internet web site references, is subject to change without notice. The example companies, individuals, products, organizations and events depicted herein are completely fictitious. Any similarity to a company, individual, product, organization or event is completely unintentional. R:BASE Technologies, Inc. shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance, or use of this material. This document contains proprietary information, which is protected by copyright. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written consent of R:BASE Technologies, Inc. We reserve the right to make changes from time to time in the contents hereof without obligation to notify any person of such revision or changes. We also reserve the right to change the specification without notice and may therefore not coincide with the contents of this document. The manufacturer assumes no responsibilities with regard to the performance or use of third party products.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

The software described in this document is furnished under a license agreement. The software may be used or copied only in accordance with the terms of that agreement. Any unauthorized use or duplication of the software is forbidden.

R:BASE Technologies, Inc. may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from R:BASE Technologies, Inc., the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Trademarks

R:BASE®, Oterro®, R:BASE C/S:I®, RBAAdmin®, R:Scope®, R:WEB Suite®, R:Mail®, R:Charts®, R:Spell Checker®, R:Docs®, R:BASE Editor®, R:Scheduler®, R:BASE Plugin Power Pack®, R:Style®, R:Code®, R:Struc®, RBZip®, R:Fax®, R:QBDataDirect®, R:QBSynchronizer®, R:QBDBExtractor®, R:Mail Editor®, R:Linux®, R:Archive®, R:Chat®, RDCC Client®, R:Mail Editor®, R:Code®, R:Column Analyzer®, R:DF Form Filler®, R:FTPClient®, R:SFTPClient®, R:PDF Form Filler®, R:PDFWorks®, R:PDFMerge®, R:PDFSearch®, RBInstaller®, RBUpdater®, R:Capture®, R:RemoteControl®, R:Synchronizer®, R:Biometric®, R:CAD Viewer®, R:Twain2PDF®, R:Tango®, R:SureShip®, R:BASE Total Backup®, R:Scribbler®, R:SmartSig®, R:JobTrack®, R:TimeTrack®, R:Syntax®, R:WatchDog®, R:Manufacturing®, R:Merge®, R:Documenter®, R:Magellan®, R:WEB Reports®, R:WEB Gateway®, R:ReadyRoute®, R:Accounting®, R:Contact®, R:DWF Viewer®, R:Java®, R:PHP® and Pocket R:BASE® are trademarks or registered trademarks of R:BASE Technologies, Inc. All Rights Reserved. All other brand, product names, company names and logos are trademarks or registered trademarks of their respective companies.

Windows, Windows 7, Vista, Windows Server 2003-2008, XP, and Windows 2000 are registered trademarks of Microsoft Corporation.

Printed: December 2009 in Murrysville, PA

First Edition

Table of Contents

Part I Command Index	2
1 Reading Command Syntax	2
2 A	3
ABORT	3
ALERTTABLE	3
APPEND	8
ATTACH	9
AUTOCHK	11
AUTONUM	14
3 B	17
BACKUP	17
BEEP	20
BREAK	20
BROWSE	21
BROWSEUSING	24
4 C	25
CALL	25
CASCADE	26
CHDIR	26
CHDRV	27
CHKDSK	28
CHOOSE	29
OPTION parameters	34
Title	35
List	36
Buttons	38
CLEAR	39
CLOSE	40
CLOSEWINDOW	40
CLS	41
CODELOCK	41
COMMENT	43
COMMENT ON	44
COMMIT	46
COMPUTE	46
CONNECT	48
CONTINUE	50
CONVERT	50
COPY	51
CREATEINDEX	52
CREATE SCHEMA	55
CREATETABLE	57
CREATEVIEW	61
CREATEOBJECT	64
CROSSTAB	69
5 D	70

	DEBUG	70
	DECLARECURSOR	71
	DEFF	74
	DELETE	74
	DELETEDUPLICATES	75
	DELETEOBJECT	76
	DELROW	76
	DETACH	77
	DIALOG	78
	format	82
	OPTION parameters	88
	Message	88
	Buttons	90
	OK and CANCEL	90
	YES and NO	93
	Window	94
	Popup Menu	95
	LOOKUP	95
	DATE	98
	TIME	99
	DATETIME	100
	CALCULATOR	102
	DIR	102
	DISCONNECT	104
	DISPLAY	105
	DROP	106
	DUPROW	109
6 E		109
	EDIT	109
	EDIT USING	112
	ENTER	114
	ERASE	115
	EXIT	115
	EXITFORM	115
7 F		116
	FETCH	116
	FILLIN	117
	FLUSH	119
	FORMS	119
8 G		120
	GATEWAY	120
	EXPORT	121
	DB (Paradox)	122
	DBF (dBASE)	123
	CSV (Comma Separated Values)	124
	FIX (Fixed Width)	126
	HTM (HTML)	127
	XLS (MS Excel Spreadsheet)	128
	XLSW (MS Excel Workbook)	129
	DOC (MS Word Document)	131
	TAB (Tab Separated Values)	133
	SLK (Symbolic Link)	134
	DIF (Data Interchange Format)	135

	WK1 (Lotus)	136
	WQ1 (QuattroPro).....	137
	SQL (SQL Script).....	138
	XML (Extended Markup Language).....	140
	CLBR (MS Windows Clipboard).....	141
	RTF (Rich Text Format).....	142
	SAV (SPSS Format)	143
	PDF (Adobe Acrobat Portable Document)	144
	LDIF (Data Interchange Format).....	145
	IMPORT	146
	DB (Paradox)	147
	DBF (dBASE)	148
	ASCII Text (*.txt, *.csv, *.tab).....	148
	XLS (MS Excel).....	149
	WKS (Lotus 1-2-3).....	150
	HTM(HTML)	151
	WQ1 (QuattroPro).....	152
	XML (Extended Markup Language).....	152
	MDB (MS Access).....	153
	GET	154
	GETPROPERTY	155
	GOTO	156
	GRANT	157
9 H		161
	HELP	161
10 I		162
	IF/ENDIF	162
	INPUT	164
	INSERT	164
	INTERSECT	166
11 J		167
	JOIN	167
12 L		169
	LABEL	169
	LAUNCH	170
	LBLPRINT	172
	SCREEN	173
	PRINTER	177
	BMP	179
	EMF	181
	ETXT	183
	GIF	184
	HTML	186
	JPG	193
	PDF	196
	RTF	203
	TIFF	207
	TXT	209
	WMF	210
	XHTML	212
	XLS	220
	EMAIL SET ON.....	223
	LIST	225

	LIST ACCESS	228
	LOAD	230
13 M		236
	MAXIMIZE	236
	MINIMIZE	236
	MKDIR	236
14 N		237
	NEWPAGE	237
	NEWROW	237
	NEXTROW	237
	NEXTTAB	237
	NORMALIZE	237
15 O		238
	ONCONNECT/PACK	238
	ONERROR	239
	OPEN	239
	ORDERBY	240
	OUTPUT	241
16 P		244
	PACK	244
	PAUSE	246
	OPTION parameters	250
	Message	252
	Meter	254
	Gauge Meter	255
	PLUGINS	256
	GUI Specific	257
	R:Dialog Builder Plugin	257
	R:Dialog with Buttons Builder Plugin	260
	R:Pause Builder Plugin	263
	R:Whois Plugin	266
	R:Choose Builder Plugin	267
	R:MAPI Mail Plugin	270
	Command Line Specific.....	271
	LoadDirectoryName	271
	LoadFileName	272
	LoadFileNamePlus.....	273
	LoadFileNamePlusPlus.....	274
	RCalculator	276
	Create Custom Plugins	276
	PREVROW	282
	PREVTAB	282
	PRINT	283
	SCREEN	284
	PRINTER	288
	BMP	291
	DIF	293
	EMF	294
	ETXT	297
	GIF	298
	HTML	300
	JPG	308
	LOTUS	311

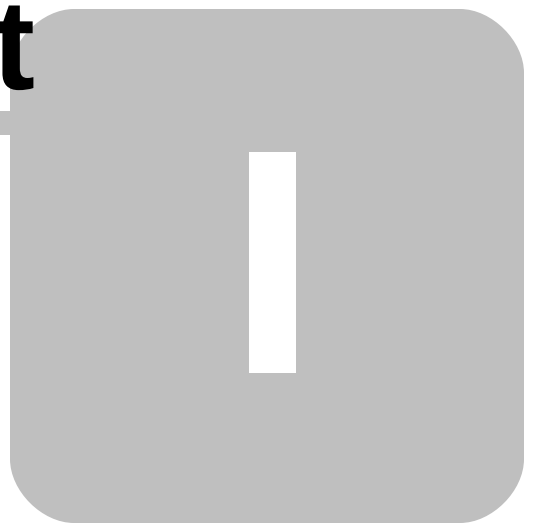
	PDF	312
	PNG	321
	QPRO	323
	RTF	325
	SVG	329
	SYLK	331
	TIFF	333
	TXT	335
	WMF	337
	XHTML	339
	XLS	347
	EMAILSETON.....	350
	PRNSETUP	352
	ProcessMessage	355
	PROJECT	357
	PROPERTY	359
	PUT	360
17 Q		362
	QUERY	362
	QUIT	363
18 R		364
	RBAPP	364
	RBBEDIT	364
	RBDEFINE	365
	RBEDIT	366
	RBLABELS	366
	RBSYNC	367
	RECALC	369
	RECOVER	370
	REFF	371
	RELOAD	371
	RENAME	372
	REPORTS	374
	RESETROW	374
	RESTORE	375
	RETURN	376
	REVOKE	377
	RHELP	378
	RHIDE	379
	RMDIR	379
	ROLLBACK	380
	RPHONE	380
	RSHOW	381
	RULES	381
	RUN	382
19 S		383
	SATTACH	383
	SAVEROW	386
	SCONNECT	387
	SCREENRESTORE	389
	SDETACH	389
	SDISCONNECT	390
	SELECT	390

SELECT Functions.....	393
TOP	396
INNERJOIN.....	396
INTO	398
FROM	399
LIMIT	400
OUTERJOIN	401
WHERE	403
Sub-SELECT.....	408
AS	408
GROUPBY.....	410
HAVING	412
ORDERBY.....	414
UNION	415
HTML	416
SET	417
AND	418
ANSI	418
AUTOCOMMIT	419
AUTOCONVERT.....	419
AUTODROP.....	419
AUTORECOVER.....	420
AUTOROWVER.....	420
AUTOSKIP.....	420
AUTOSYNC.....	420
AUTOUPGRADE.....	420
BELL	421
BLANK	421
CAPTION	421
CASE	421
CLEAR	421
CLIPBOARD.....	422
COLOR	422
COMPATIB.....	423
CURRENCY	424
DATE	425
DEBUG	426
DELIMIT	426
ECHO	427
EDITOR	427
EOFCHAR.....	428
EQNULL	428
ERRORMESSAGE.....	429
ERRORMESSAGES	429
ERROR VARIABLE	429
ESCAPE	431
EXPLODE.....	431
FASTFK	431
FASTLOCK.....	432
FEEDBACK.....	432
FILES	433
FIXED	433
FONT	433
HEADINGS.....	433

IDQUOTES	433
INDEXONLY	434
INSERT	434
INTERVAL	434
KEYMAP	434
LAYOUT	435
LINEEND	435
LINES	435
LOCK	435
LOOKUP	436
MANOPT	436
MANY	437
MAXTRANS.....	437
MDI	437
MESSAGES	437
MIRROR	438
MOUSE	438
MULTI	438
NAME	438
NAMEWIDTH.....	439
NOTE_PAD.....	439
NULL	439
ONELINE	439
PAGEMODE	440
PASSTHROUGH.....	440
PLUS	441
POSFIXED.....	441
PRINTER	441
PROCEDURE.....	441
PROGRESS	442
QUALCOLS	442
QUOTES	443
RADMIN.....	443
RECYCLE	444
REFRESH.....	444
REVERSE	444
ROWLOCKS.....	445
RULES	445
SCRATCH.....	445
SELMARGIN	446
SEMI	446
SEMI (Special Character)	446
SERVER	446
SHORTNAME.....	447
SINGLE	447
SORT	448
SORTMENU	448
STATICDB	448
TIME	448
TIMEOUT	449
TOLERANCE.....	450
TRACE	450
TRANSACT	450
USER	451

	VERIFY	452
	WAIT	453
	WALKMENU	453
	WHILEOPT	454
	WIDTH	454
	WINBEEP	454
	WRAP	455
	WRITECHK	455
	ZERO	456
	ZOOMEDIT	456
	SET VARIABLE	456
	SETFOCUS	460
	SHOW	460
	SHOWBUILD	464
	SHOWERROR	464
	SHOWVARIABLE	465
	SHOWVERSION	466
	SKIP	467
	SSQL	468
	SUBTRACT	468
	SWITCH/ENDSW	469
20 T		471
	TALLY	471
	TILE	471
	TRACE	471
	TURBO	473
	TYPE	474
21 U		474
	UNION	474
	UNLOAD	476
	UPDATE	481
	UPGRADE	484
22 W		485
	WHENEVER	485
	WHERE	486
	GROUPBY	491
	HAVING	494
	WHILE/ENDWHILE	496
	WRITE	498
23 Z		500
	ZIP	500
Index		502

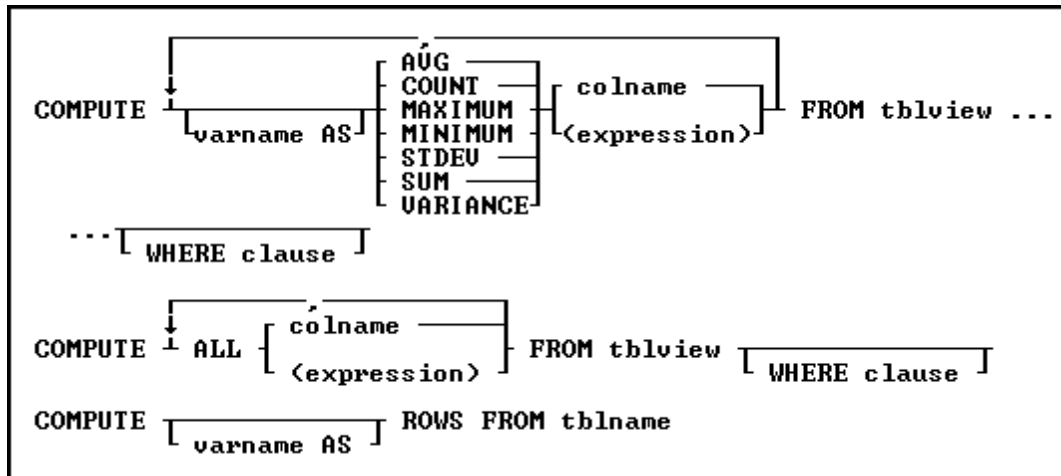
Part



1 Command Index

1.1 Reading Command Syntax

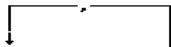
How To Read the Parts of the Command Syntax Diagram



Text Objects in the Command Syntax Diagram

- Keywords, which are uppercase, tell R:BASE what to do.
- Arguments, which are lowercase, represent specific information that you provide, such as a table name, column name, variable name, or expression.

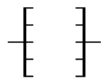
Graphical Parts of the Command Syntax Diagram



An arrow in a command syntax indicates what portion of the command can be repeated. Each part of the command that is repeated must be separated with a comma, or the current delimiter character.

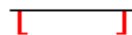
...

Ellipses indicate that the syntax continues to the next line.



If you have a choice of keyword or argument to use, the choices are enclosed in brackets.

This is the main line of the syntax. Any keywords or arguments on the main line are required.



This part of the syntax is below the main line and is therefore optional.

1.2 A

1.2.1 ABORT

Use the ABORT command in conjunction with the ON CONNECT/PACK command to stop the process, or with a Stored Procedure to stop a trigger.



Options

ON

The ON option is used in conjunction with the [ON CONNECT/PACK](#) command. It stops the connect or pack from being performed after the command file finishes.

TRIGGER

The TRIGGER option is used in a Stored Procedure to cancel the UPDATE, INSERT, or DELETE.

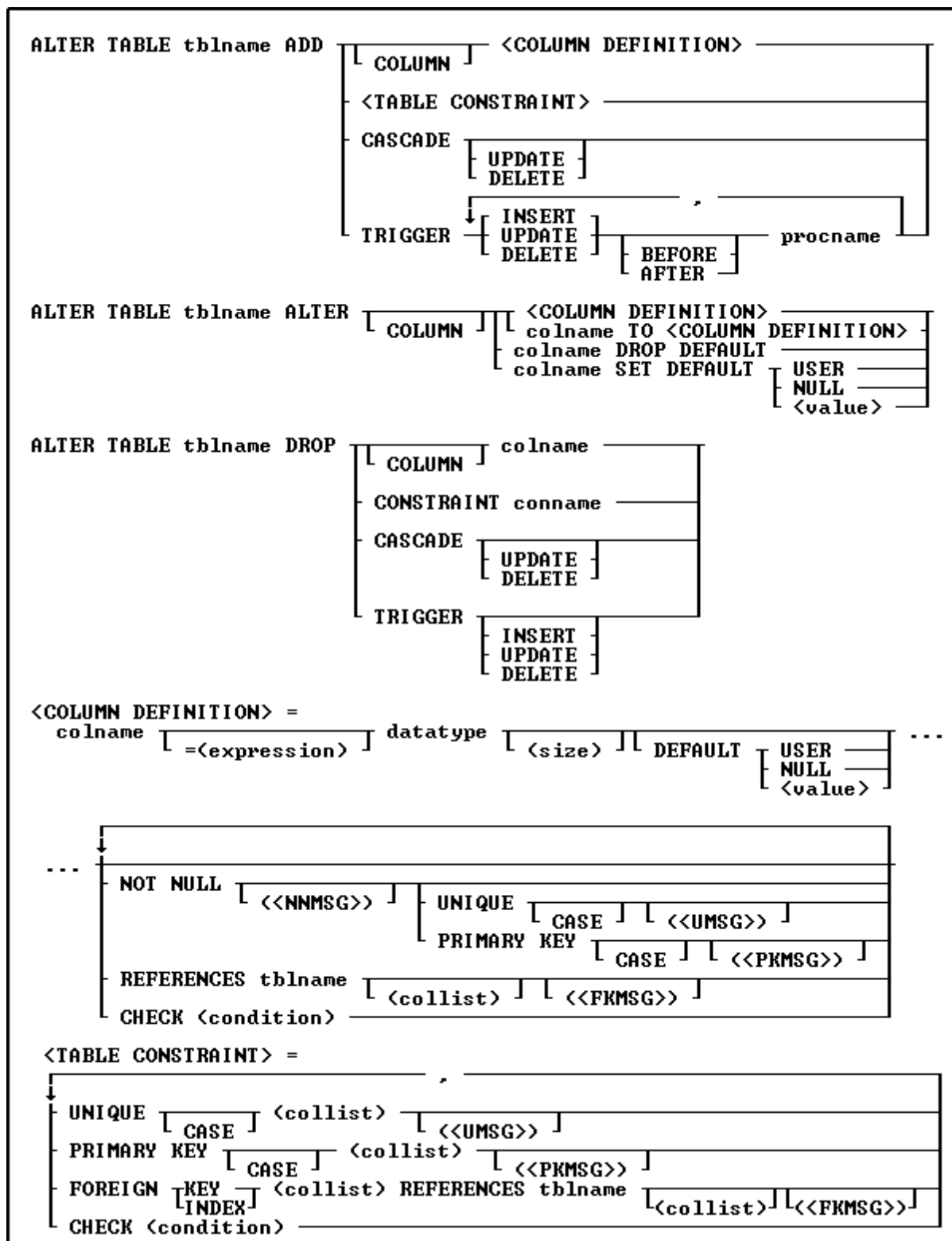
About the ABORT ON Command

The ABORT ON command is placed in the command file specified by the ON CONNECT/PACK command. It indicates that the CONNECT or PACK should not be performed after the command file finishes. For example, when using [AUTOCHK](#) with ON PACK to check the database for errors, use ABORT ON so the PACK is not performed if errors are found in the database.

Since a trigger runs a Stored Procedure before the row that triggered it is updated, inserted, or deleted, you can use the ABORT TRIGGER command to cancel the UPDATE, INSERT, OR DELETE.

1.2.2 ALTER TABLE

Use the ALTER TABLE command to modify an existing table.



Options

' Indicates that this part of the command is repeatable.

ADD

Specifies the column and its definition, or a table constraint to add.

ADD CASCADE

Maintains primary/foreign key relationships automatically. For example, if you either UPDATE or DELETE a primary key value from a table, the corresponding foreign key values are updated or deleted automatically. A CASCADE can applied to UPDATE, DELETE or BOTH to specific primary keys. By not specifying either UPDATE or DELETE, both CASCADE restrictions will be enforced upon the primary/foreign key tables. Separate UPDATE and DELETE data restrictions can allow a CASCADE to be enforced for records that are updated, but not enforced when records are deleted, in order to avoid an accidental or undesired record delete. CASCADE can only be added to tables with primary keys.

ADD TRIGGER

Adds the specified triggers to the table. Triggers run a Stored Procedure when an UPDATE, DELETE, or INSERT is executed. If you are using BEFORE and AFTER triggers, BOTH must be ADDED at the same time.

AFTER

Sets the trigger to activate after the INSERT, UPDATE or DELETE action.

ALTER

Modifies a column definition.

BEFORE

Sets the trigger to activate before the INSERT, UPDATE or DELETE action. This is the default setting if the BEFORE/AFTER parameter is unused.

CASE

Specifies that the data values will be case sensitive.

CHECK (condition)

Sets a condition to be satisfied before an update or insertion of a row can occur, which creates an R:BASE rule.

(collist)

Specifies a list of one or more column names, separated by a comma (or the current delimiter), used in the unique key specification. This option is only used when referencing a unique key.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

COLUMN

Specifies the column to add, drop, or alter.

conname

Specifies a constraint name.

datatype

Specifies an R:BASE data type.

DEFAULT

Specifies a default value for the column if no value is provided by the user.

DROP

Removes a column or a constraint. A column, including both its structure and data, is removed from the table. Dropping a constraint removes a primary key, foreign key, unique key, or a not-null constraint.

DROP CASCADE

Disables the CASCADE feature so that primary/foreign key relationships are not maintained automatically.

DROP CONSTRAINT

Removes a constraint.

DROP DEFAULT

Removes a column's default value.

DROP TRIGGER

Drops either all triggers for a table, or the specified INSERT, UPDATE, or DELETE trigger. There is no need to specify BEFORE and/or AFTER because both triggers will be deleted if they are present using this command.

= (expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as #date, #time, and #pi.

(<FKMSG>)

Creates a constraint violation message to appear whenever a foreign-key data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a foreign-key constraint violation. You can define two messages: one for inserting and one for updating. A constraint must be dropped, then recreated in order to modify the violation message.

FOREIGN INDEX

With the FASTFK setting on, creates a foreign key that has an index using row pointers for data retrieval on selected columns.

FOREIGN KEY

Specifies a column or set of columns required to match values in a particular primary key or unique key constraint defined in a table.

FOREIGN KEY (collist)

If (collist) comprises one column, this option is equivalent to FOREIGN KEY. If two or more columns are included in (collist), the values in the listed columns must be unique as a group in each row. Each column must be separated by a comma (or the current delimiter).

(<NNMSG>)

Creates a constraint violation message to appear whenever a not-null data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a not-null constraint violation. A constraint must be dropped, then recreated in order to modify the violation message.

NOCHECK

Optional NOCHECK parameter does not update references to views, tables, and columns in forms, reports, labels, access rights, and rules. In this case, user assumes the responsibilities to update any references to views, tables, and columns in forms, reports, labels, access rights, and rules. This condition is ONLY available for the **ALTER COLUMN** parameter.

NOT NULL

Prevents a column from accepting null values, but permits it to accept duplicate values. If this option is specified without a setting for a default value, you cannot insert rows without specifying values for the given column.

(<PKMSG>)

Creates a constraint violation message to appear whenever a primary-key data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a primary-key constraint violation. You can define three messages: one for uniqueness, one for deleting, and one for updating. A constraint must be dropped, then recreated in order to modify the violation message.

PRIMARY KEY

Specifies the column(s) to designate as a primary key constraint.

PRIMARY KEY (collist)

If (collist) comprises one column, this option is equivalent to PRIMARY KEY. If two or more columns are included in (collist), the values in the listed columns must be unique as a group in each row. Only columns defined as not null can be included in (collist). Each column must be separated by a comma (or the current delimiter).

procname

The procedure name. If a procedure by this name already exists in the database, an error is generated.

REFERENCES

Identifies the primary key or unique key table to which the foreign key refers.

SET DEFAULT

Changes a column's default value.

(size)

Defines the length of a column of either the CHARACTER data type (if not the default 1) or TEXT data type (if not the default 8). Defines the precision and scale of a column of the DECIMAL or NUMERIC data type, if not the default of precision 9 and scale 0 (9,0). VARBIT, VARCHAR, and BIT either require or can have a size.

tblname

Specifies a table name. In R:BASE eXtreme 9.0 (32), the table name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the table name is limited to 128 characters.

(<UMSG>)

Creates a constraint violation message to appear whenever a unique-key data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a unique-key constraint violation. You can define three messages: one for uniqueness, one for deleting, and one for updating. A constraint must be dropped, then recreated in order to modify the violation message.

UNIQUE

Requires the values in a column to be unique by creating a unique key constraint.

UNIQUE (collist)

If (collist) is one column, this option is equivalent to UNIQUE. If two or more columns are included in (collist), the values in the listed columns must be unique as a group in each row. Only columns defined as not null can be included in (collist). Each column must be separated by a comma (or the current delimiter).

USER**NULL****(value)**

Default USER: Specifies the default value to be the user identifier.

Default NULL: Specifies the default value to be null.

Default (value): Specifies the default to be the indicated value.

About the ALTER TABLE Command

ALTER TABLE creates a temporary internal table by copying a table's structure and data. You must have enough disk space to hold another copy of a table, and your database should not exceed the number of tables and columns R:BASE allows, which includes user-defined tables and system tables.

After the ALTER TABLE command has been executed, the temporary table goes away; however, the disk space the temporary table occupied is not available. To recover this space, pack or reload the database using the [PACK](#) or [RELOAD](#) commands.

The ALTER TABLE command is supported in EEPs.

Adding Columns

When you add a new column to a database, specify the name, data type, and length when the data type for the column is CHARACTER or TEXT, or precision and scale when the data type for the column is DECIMAL or NUMERIC. When the column is computed, specify the name and expression-a data type is optional. When the column already exists in the database, specify only the name-R:BASE uses the existing data type, and length, if applicable.

Database Access Rights with ALTER TABLE

When access rights for a table or view have been assigned using the [GRANT](#) command, ALTER TABLE requires the database owner's user-identifier or permission from the owner to alter specific tables.

Limitations of the ALTER TABLE Command

You cannot assign an index to a new column or transfer the index of an existing column with ALTER TABLE. If the added column should be indexed, use the [CREATE INDEX](#) command.

You also cannot add or transfer rules with ALTER TABLE. If you want a rule to apply to a column in the table, you must add it with the [RULES](#) command or use the Database Designer.

You cannot use ALTER TABLE to modify a view.

Examples

The following command adds mailadrs, a TEXT column 40 characters wide, at the end (or far right) of the customer table.

```
ALTER TABLE customer ADD mailadrs TEXT 40
```

The following command adds the profit column at the end of product table. The value of profit is computed from the current row values for listprice multiplied by 1.05. The data type specified is REAL.

```
ALTER TABLE product ADD profit=(listprice * 1.05) REAL
```

The following command defines columns one through three as a case-sensitive primary key. Before you use this command, you must add a not-null constraint to each of the columns.

```
ALTER TABLE tablename ADD PRIMARY KEY CASE (column1, column2, column3) ('This is a message from the primary key')
```

The following command line adds a foreign index to the custidcolumn and references the primary key in the customer table.

```
ALTER TABLE transmaster ADD FOREIGN INDEX (custid) REFERENCES customer
```

The following command line adds the test trigger for the SampleTriggers Table.

```
DROP PROCEDURE MySampleTrigger  
PUT AFTER.PRC AS MySampleTrigger  
ALTER TABLE SampleTriggers ADD TRIGGER INSERT AFTER MySampleTrigger  
RETURN
```

1.2.3 APPEND

Use the APPEND command to copy rows from a table or view to the end of a table.

```
APPEND tblview TO tblname [ WHERE clause ]
```

Options

tblview

Names the table or view from which you want to copy rows-the source.

TO tblname

The name of the table to which you want to copy rows-the destination.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the APPEND Command

R:BASE only copies values from the source table or view that have matching column names in the destination table. Columns in the destination table that are not in the source table or view are filled with null values.

Rows are copied, not removed, from the source.

Example

The following command adds the rows containing new employee information from the newemp table to the end of emptable, a table containing information about previous employees. A WHERE clause is not specified, so all rows are copied to emptable.

```
APPEND newemp TO emptable
```

1.2.4 ATTACH

Use the ATTACH command to attach a dBASE file to an open R:BASE database.

```
ATTACH filespec [ USING [ ndxlist ] ]
```

Options

filespec

A dBASE database name with a drive and path specification in the form D:\PATHNAME/FILENAME.

ndxlist

Specifies a list of index files to associate with the specified dBASE file. You do not have to include the extension for each index file. Separate index file names with a comma (or the current delimiter). Index files must be located with the specified dBASE file.

USING

Removes dBASE index files that were previously associated with the attached dBASE file when this option is used without a list of index files.

About the ATTACH Command

Before you can attach a dBASE file, an R:BASE database must be open. You can open an existing database or use the [CREATE SCHEMA](#) command to create a database.

Include the file specification when the file is located on a different drive or directory. You do not have to include the .DBF extension for the dBASE file.

R:BASE directly reads and writes dBASE III and dBASE III PLUS data and index files. R:BASE can also read and write dBASE IV data files and index files that have the .NDX extension, just as dBASE III and dBASE III PLUS can share files with dBASE IV. R:BASE cannot read encrypted files nor read and write to dBASE IV index files, which have .MDX extensions.

Attaching to dBASE Files from a Network

From a network, R:BASE, dBASE III, and dBASE III PLUS users can access the same file at the same time. R:BASE can lock a dBASE file just as dBASE III and dBASE III PLUS can lock a dBASE file. When R:BASE is in multi-user mode, it does not support dBASE IV use. When a dBASE IV file is open by dBASE, R:BASE cannot access that file; when R:BASE attaches to a dBASE IV file, that file cannot be accessed by dBASE IV.

Listing dBASE Files

Use the LIST command to list the dBASE files in an R:BASE database. R:BASE displays DBF in the Rows column to indicate a dBASE file.

Reattaching dBASE Files to R:BASE

A dBASE file stays attached unless you use the [DETACH](#) command, which removes a dBASE file and its associated index file from the R:BASE database. The dBASE files stay attached because R:BASE remembers the attached dBASE files and index files when you open a database. At that time, R:BASE searches the current directory and path to find the attached dBASE file; therefore, the location of the dBASE files must be included in your path.

Associating and Modifying dBASE Index Files

You can associate a maximum of seven dBASE index files, which have .NDX extensions, with a dBASE data file by using the ndxlist option. R:BASE remembers each index file you associate with the dBASE data file. Also, use the ndxlist option to modify or preserve a set of indexes. If you issue another ATTACH command with a list of index files, R:BASE removes the current index files from the dBASE data file and associates the new list with the dBASE data file.

R:BASE updates the information stored in the dBASE data and index files each time you add or edit information in a dBASE file.

R:BASE Commands that Work with dBASE

The following R:BASE commands work with dBASE files. Limitations are noted following the table.

Commands that Work with dBASE			
ATTACH	DROP LABEL	LIST CURSORS	RENAME FORM
BROWSE	DROP REPORT	LIST DATABASES	RENAME OWNER
CHOOSE	DROP RULE	LIST FORMS	RENAME REPORT
COMMENT ON	DROP TABLE	LIST LABELS	RENAME VIEW
COMMIT (4)	DROP VIEW	LIST REPORTS	REPORTS
COMPUTE	EDIT	LIST RULES	REVOKE
CONNECT	EDIT DISTINCT	LIST TABLES	ROLLBACK (4)
CONTINUE	EDIT USING	LIST VIEWS	RULES
CREATE VIEW	ENTER USING	LOAD	SELECT
CROSSTAB	FETCH	OPEN CURSOR	SET
DECLARE CURSOR	FUNCTIONS	PACK (1)	SHOW
DELETE	GRANT	PRINT	TALLY
DELETE DUPLICATES	INSERT INTO	PROJECT (2)	UNLOAD (3)
DETACH	LBLPRINT	QUERY	UPDATE
DISCONNECT	LIST ACCESS	RBLABELS	WHENEVER
DROP CURSOR	LIST ALL	RELOAD	ZIP
DROP FORM	LIST COLUMNS	RENAME	COLUMN

Notes:

- dBASE files are not affected when you use a [PACK](#) command.
- Using the PROJECT command, you can create a new table from an existing table from dBASE to R:BASE, but not from R:BASE to dBASE.
- You can unload dBASE tables as ASCII only.
- You cannot modify dBASE tables when transaction processing is on.
- dBASE memo fields can be 64K in size. If the dBASE memo field is larger than 4K (the maximum

size of an R:BASE note column), R:BASE reads as much as will fit. If you make changes and then write the record back to dBASE, the existing dBASE memo field is overwritten. The Carriage Return and Line Feed characters in dBASE are mapped to [Alt] + [0255].

R:BASE Commands that Do Not Work with dBASE

The following commands do not work with dBASE files in R:BASE.

Commands that Do Not Work with dBASE		
ALTER TABLE (1)	CREATE INDEX ON	JOIN
APPEND	CREATE SCHEMA AUTHOR	RENAME TABLE
AUTONUM	CREATE TABLE	RESTORE
BACKUP ALL	DROP COLUMN	SUBTRACT
BACKUP DATA	DROP INDEX	UNION
BACKUP STRUCTURE	INTERSECT	

- Column names for dBASE files can be changed with ALTER TABLE.

Example

In the following example, the first command opens the concompdatabase. The second command attaches the dBASE file SAMPGATE to the concomp database and associates the dBASE index files COMPID and PRODDISC with the R:BASE file table sampgate.

```
CONNECT concomp
ATTACH sampgate USING compid, proddesc
```

1.2.5 AUTOCHK

Use the AUTOCHK command to check the integrity of a database. You must be disconnected from the database first.

```
AUTOCHK [ dbspec ] [ FULL ]
```

Options

dbspec

Specifies a database other than the open database to check; otherwise, the open database is checked.

FULL

Provides detailed information about the processing being performed, and when AUTOCHK encounters an error, it continues processing.

About the AUTOCHK Command

Use the AUTOCHK command to ensure that the connected database is intact before using the [PACK](#) or [RELOAD](#) commands, or before making a backup of the database with either the [BACKUP](#) or [COPY](#) commands.

Please Note: If any user connected to the database has temporary tables or views created you may receive an abnormal amount of errors. This is expected and is a side effect of having temporary tables active during the check. For completely accurate results, have all users disconnect from the database to be checked.

AUTOCHK checks the following:

- The structure-file block sizes and locations.
- The timestamps for all database files.
- The database-file lengths.
- The number of tables and columns.
- The starting and ending pointers for tables.
- The location of columns.
- The File 4 data pointers.
- The data types of columns.
- The size and number of rows in each table.
- The row pointers in the data file.

AUTOCHK does not check indexes.

When you run AUTOCHK, it systematically checks the structure file of the open database, and the data files. AUTOCHK only checks the index file for the timestamp and length of the file. When opening a database, AUTOCHK ignores any user-identifier protection. AUTOCHK without the FULL option sets the R:BASE error variable to a non-zero value if errors are found.

The results of AUTOCHK with the FULL option are displayed on screen, or the current output device. First, AUTOCHK validates the timestamps in the database files, then systematically checks the structure of each table and view in the database, providing a list of columns, constraints, and indexes for each. Any structure errors are noted after each table listing.

Database Statistics

Next, AUTOCHK checks the data for each table, listing active rows and deleted rows. Any problems with data, such as broken pointers, are listed after the respective tables. Finally, AUTOCHK provides a summary of the database structure, including the number of tables, columns, and indexes, and the actual space that the data occupies in the data file (File 2). AUTOCHK shows the percent of space used for the items in each list to give an idea of how much space has been used, and to indicate the need to recover space in the database files. Any numbers less than 100 percent indicate the need to pack or reload the database using the [PACK](#) or [RELOAD](#) commands.

The following section contains information about using AUTOCHK in application files and capturing the error variables returned. This allows the application developer to prevent users from continuing to use a corrupted database.

```
SET ERROR VAR E1
WRITE 'Checking database for errors...'
AUTOCHK dbname
IF E1 > 40 THEN
    WRITE 'AUTOCHK has found errors in the database!'
    BEEP
ENDIF
IF E1 > 0 and E1 < 50 THEN
    WRITE 'AUTOCHK will not run - User Abort or Out of Memory'
    BEEP
ENDIF
IF E1 = 0 THEN
    WRITE 'AUTOCHK successful - No errors found'
ENDIF
PAUSE 2
RETURN
```

If AUTOCHK with no option finds an error, it stops checking the database and displays one error message. If the error message (see list below) indicates that the database is damaged, you might want to start using a backup copy of the database. Alternatively, you might want to use R:SCOPE, a database repair tool available from R:BASE Technologies, Inc.

If AUTOCHK finds no errors, it displays the message "NO ERRORS FOUND." If you press any key while AUTOCHK is checking the database, the program stops and displays the message "USER ABORT."

AUTOCHK automatically sets the error variable to the number corresponding to the message returned. For example, if the error "UNABLE TO OPEN DATABASE FILE 2" is returned, the error variable is set to 52.

Multi-User Databases

Use caution when running AUTOCHK in a multi-user environment. If the database being checked is currently open with **MULTI** set on, AUTOCHK places a database lock on the database. The database lock remains in effect until AUTOCHK stops checking the database. Database users are unable to make any changes to the data or structure of the database while this lock is in place.

If a user attempts to open a database being checked by AUTOCHK and the database does not have any other users, the user receives an error message indicating that the database is currently open in a mode that makes it unavailable. If other users have the database open with the **MULTI** set on and the database is being checked, the user attempting to open the database receives a message indicating the user is waiting in a lock queue. If AUTOCHK successfully completes checking the database and finds no errors, it reports that no errors were found and sets the error variable to 0.

Checking continues in multi-user mode (even if a database lock cannot be obtained) if a database is connected by another user; however, row errors in File 2 can occur because of database activity.

Error Messages

AUTOCHK displays one of the following messages when it is unable to start checking or complete checking the database or when it finds an error in the database files. AUTOCHK returns *0 No errors found* if the database is okay. Some of these messages indicate that the database is damaged. Either switch to a backup copy of the database, or attempt repair of the database using R:SCOPE, R:BASE Technologies's database repair tool. If AUTOCHK is unable to open File 1 of the database, check that the path you specified to the database is correct; or, if you are trying a multi-user database, check that no other user has the database connected with **MULTI** set off.

Checking continues in multi-user mode (even if a database lock cannot be obtained) if a database is connected by another user; however, row errors in File 2 might occur because of database activity.

Any of these messages, except the first (code 0), indicates that the database is damaged. Either switch to a backup copy of the database, or attempt repair of the database using R:SCOPE, R:BASE Technologies's database repair tool.

AUTOCHK Error Messages

Number	Code Message
0	No errors found
1	This database is not of the correct version
2	The database filenames must all match
20	Out of memory
40	User Abort
50	Unable to open database file number 1
51	Unable to lock this database
52	Unable to open database number 2
53	Unable to open database number 3
54	Unable to open database number 4
55	Error reading the database information block
56	Error reading the timestamp information
57	Timestamp in file number 2 does not match file 1; run RBSYNC
58	Timestamp in file number 3 does not match file 1; run RBSYNC
59	Timestamp in file number 4 does not match file 1; run RBSYNC
60	Invalid number of tables
61	Invalid number of columns
62	Invalid number of indexes
63	File 1 is too small
70	Error in database structure block
80	Error reading the table list
81	Error reading the column list
82	Error reading the index list
100	Incorrect version flag

101	Error reading Case Folding and Collating tables
110	Error in DBinfo block offset
111	Error in DBinfo block length
120	Error in length of database file 2
121	Error in length of database file 3

Example

The following is an example of how to put AUTOCHK results in a file for viewing:

```
DISC
OUTPUT dbname.chk
AUTOCHK dbname FULL
OUTPUT SCREEN
```

You can view DBNAME.CHK in the Text Editor to view the results.

1.2.6 AUTONUM

Use the AUTONUM command to define, modify, or remove an autonumber formula from a column.

```
AUTONUM colname IN tblname { DELETE
                              USING startnum [ increment ] [ format ] ...
... [ ORDER BY clause ] NUM
      NONUM
```

Options

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter #c, where #c is the column number shown when the columns are listed with the [LIST TABLES](#) command.

DELETE

Removes a column's autonumber formula.

format

Defines the format in which values are displayed. This option is used only for columns with the TEXT data type.

You can use the following formatting characters:

Formatting Character	Result
9	Specify a numeric digit; leading zeros are suppressed.
0	Specify a numeric digit; leading zeros are displayed.
. (period)	Aligns digits along a decimal point.
[] (square brackets)	Encloses literal text.

For example, if the format is [MX]9999 and the numeric value is 123, the value entered will be MX123.

IN tblname

Specifies the table in which to autonumber the column.

increment

Specifies the value of the increment as each new row is added to the table. The default increment is 1.

NONUM

Leaves existing values unchanged and assigns autonumbered values to new rows as they are added to the table. NONUM is the default option.

NUM

Renumbers all the existing values in the column defined as an autonumbered column.

ORDER BY clause

Sorts rows of data. The [ORDER BY](#) clause is only used with the NUM option.

USING startnum

Defines or redefines the formula for an autonumber column. You must specify a starting value. Optionally, you can specify an increment, and for columns with the TEXT data type, a display format. For a column in a table that contains values, you can either renumber existing values or leave them as they are.

About the AUTONUM Command

An autonumbered column ensures that each row in that column has an incremental value. For example, use an autonumbered column to assign identification numbers, model numbers, or invoice numbers:

The following types of columns can be autonumbered:

- Columns that are not computed.
- Columns with DOUBLE, INTEGER, NUMERIC, REAL, or TEXT data types.

When you use the [LIST](#) command to list information about a column or table, autonumbered columns are described as AUTONUMBER in the *attributes* column.

Automatic Numbering

R:BASE automatically enters values in an autonumbered column when you add rows to a table using a form, the Data Editor, [INSERT](#) command, or [LOAD](#) command with the NUM option. When you import rows to a table that contains an autonumber column, you can either set autonumbering off and load imported values, or set autonumbering on and let R:BASE autonumber the values.

Capturing the Autonumbered Value

The next value for an autonumbered column can be captured for extended calculation or for display in a form. To capture the value, use the NEXT Function.

Changing Values

You can change the values in an autonumbered column by using a form, or the [UPDATE](#) or [EDIT](#) command. However, if you change a value in an autonumbered column, you could assign a duplicate number or disrupt the sequence of numbers. For more information about changing values in an autonumbered column, see the below guidelines in [Renumbering Columns Containing Data](#).

Renumbering Columns Containing Data

If you renumber a column that contains data, use the following guidelines to decide whether to change the column's existing values.

Autonumber?	Option to Use	Conditions
Yes	NUM	A column exists in only one table in the database. You can use the ORDER BY NUM clause to sort the rows in the order in which you want them

		renumbered. When you add new rows, values are numbered in the order in which the rows are added to the table.
No	NONUM (or do not specify)	A linking (or common) column exists in more than one table. You will destroy the common column values that link your tables if you renumber the values in a linking column. R:BASE adds autonumbered values to new rows as you add them to the table.

Redefining Formulas

You can redefine the formula for an autonumbered column. For example, use the AUTONUM command with the NUM option to change a column's display format from suppressing leading zeros to displaying them. For more information about redefining formulas of an autonumbered column, see "Renumbering Columns" earlier in this entry.

Removing Formulas

To remove an autonumber formula for a column, use the DELETE option. R:BASE removes only the formula, not the existing values in the autonumbered column. After you remove an autonumber formula, the user must enter values in the column as rows are added.

Autonumbering Tables Created with Relational Commands

When you create a table with one of the relational commands ([INTERSECT](#), [JOIN](#), [PROJECT](#), [SUBTRACT](#), or [UNION](#)), R:BASE transfers an autonumbered column as a regular column. You must define an autonumber formula for the column in the new table.

Database Access Rights with AUTONUM

When access rights for a table have been assigned using the [GRANT](#) command, AUTONUM requires either the database owner's user identifier, or the rights to alter a table.

Examples

The following command defines an autonumber formula for the *custid* column in the *customer* table. Existing values are renumbered starting at 100; assigned values increase by one for each row. Only use this command for a column that meets the renumbering guidelines in the section "Renumbering Columns Containing Data."

```
AUTONUM custid IN customer USING 100 1 NUM
```

The following command defines an autonumber formula for the *model* column in the *product* table. Existing values are not renumbered. Values in new rows are numbered starting with 100. Assigned values increase by one each time a row is added. The numbering format specifies that the letters MX always precede the numeric value. The 0000 provides space for a numeric value of up to four digits. When the value is less than four digits, R:BASE enters leading zeros.

```
AUTONUM model IN product USING 100 1 [MX]0000 NONUM
```

The following command assigns autonumbering to the *empid* column in the *employee* table. Existing values are renumbered starting at 100; assigned values increase by one for each row. The rows are renumbered by the employees' last and first names. Only use this command for a column that meets the renumbering guidelines in the section "Renumbering Columns Containing Data."

```
AUTONUM empid IN employee USING 100 1 ORDER BY emplname, empfname NUM
```

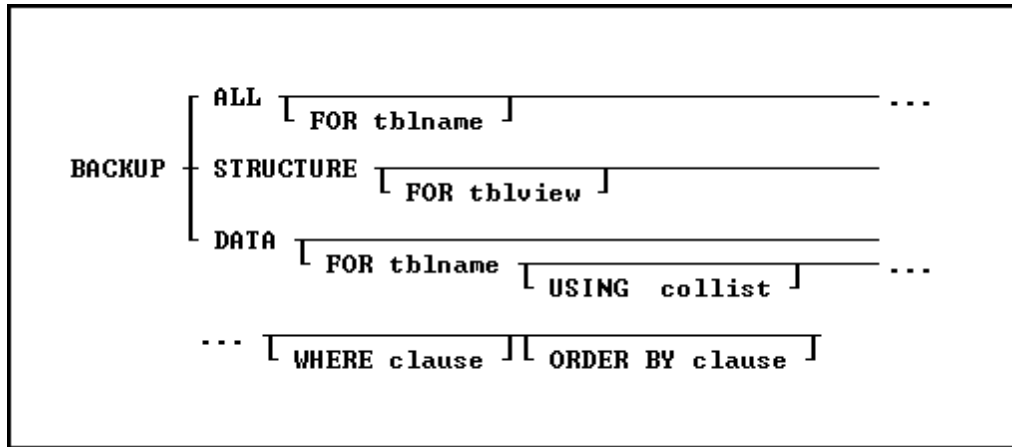
The command below deletes the autonumber formula from the *empid* column in the *employee* table.

```
AUTONUM empid IN employee DELETE
```

1.3 B

1.3.1 BACKUP

Use the BACKUP command to copy the data and/or structure of a database to floppy disks or another directory on your hard disk. The BACKUP command is very much like the [UNLOAD](#) command, only it supports the ability to span multiple diskettes when backing up databases to a floppy drive. With the many various backup software packages, or by simply performing file copies, the BACKUP command has become antiquated.



Options

ALL

Backs up the entire structure and data of the current database.

DATA

Backs up just the data of the current database.

FOR tblname

Specifies a single table to back up. If this clause is not used, all tables in the database are backed up.

FOR tblview

Specifies a table or view to back up. If this clause is not used, all tables and views in the database are backed up.

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

STRUCTURE

Backs up just the structure of the current database.

USING collist

Specifies the columns to back up and in what order to back them up. USING ALL backs up all columns in the order they are defined in the table.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the BACKUP Command

The BACKUP command backs up data and/or structure to a backup file in the ASCII delimited format, which you can edit with a text editor. For example, you can restore the structure or data from one database to a new database, then edit the backup file with a text editor to change the database name to a new one.

BACKUP does not back up computed-column values; the values will be computed when the database is

restored.

BACKUP does not change the data or the structure in the original database. Use the [RESTORE](#) command to restore your data.

If a BACKUP command is included in a transaction when transaction processing is on, the backup cannot be rolled back.

The BACKUP command creates a file with a .LOB extension for binary large objects, and a file for the data and/or structure.

SET Special Characters and Operating Conditions

To properly restore a database, BACKUP places the settings for the [SET](#) special characters and operating conditions used in the database at the beginning of the ASCII backup file. Within the backup file, object names, such as table names, will be enclosed within the IDQUOTES character. The current setting for DELIMIT is used to separate values. Text strings are enclosed in the current setting of QUOTES. And, the current value of WIDTH affects the width of data lines in the backup file. To view your database character and display settings, select "Settings" > "Configuration Settings" from the main Menu Bar.

The following are three tips that will ensure your database is restored from a backup properly:

- Set the null symbol to -0- (the R:BASE default) before backing up.
- Do not set a special character to the same setting as another special character.
- For best results backing up, keep all the default settings.

Backing Up to Disks

When you back up data to floppy disks, use blank formatted disks. The backup file can span multiple floppy disks. If BACKUP fills a disk, it prompts you to enter a new disk before continuing.

Note: So that you don't mix up the disks, label and number them as you remove them.

If you are backing up to a hard disk and the disk becomes full, press [Esc] to stop the backup. You can either free up hard disk space before backing up again, or back up to floppy disks.

It is worth noting that even when using R:BASE for Windows you will need to issue this command at the R> Prompt or via a command file.

Note: Backing Up to Floppy Disks is not recommended when using any 6.x, or higher, versions of R:BASE.

Using the BACKUP Command

When you back up your data, use the [OUTPUT](#) command to specify the filename of the backup. The BACKUP command does not erase files on a floppy disk, but overwrites a file if the filename you specify already exists on the disk. For examples of how to use the OUTPUT command, see the OUTPUT reference.

You can back up a database in one of the following ways:

- Back up both the structure and data.
- Back up only the structure.
- Back up only the data.

The following is information about each method.

Backing Up the Structure and Data

Use the BACKUP ALL command to back up the structure and data in the current database, or the structure and data for a specific table. You can backup a database structure and data to separate files if you prefer two smaller files to one large file. BACKUP writes all the commands necessary to define the database or table, starting with the [CREATE SCHEMA AUTHORIZATION dbname](#) command near the beginning of the file.

BACKUP uses the [LOAD](#) command to load the data.

If the database is protected by the database owner's user identifier, BACKUP ALL requires the owner's user identifier, which R:BASE places in the backup file to ensure that the restored database file continues to be protected. Keep the backup file from unauthorized users to ensure the database file remains protected.

Backing Up the Structure Only

Use the BACKUP STRUCTURE command to back up just the structure of the current database, or the structure of a specific table. BACKUP writes all the commands necessary to define the database or table, starting with the CREATE SCHEMA AUTHORIZATION dbname command near the beginning of the file. Any collating or case-folding table defined by the database owner is automatically preserved.

Keep the backup file from unauthorized users to ensure the database file remains protected. If the database is protected by an owner's user identifier, BACKUP STRUCTURE requires the owner's user identifier, which R:BASE places in the backup file to ensure that the restored database file continues to be protected.

Backing Up Data Only

Use the BACKUP DATA option to back up the data in the current database or the data in a specific table. BACKUP uses the LOAD command to load the data. Also, use the BACKUP DATA option to back up any custom information stored in R:BASE system tables such as Forms, Reports, Labels and Stored Procedures.

Examples

The following command lines back up the structure and data for every table in the open database to the MYDB.BUP file on drive A:, redirect the output to the screen, and close the file.

```
OUTPUT a:mydb.bup
BACKUP ALL
OUTPUT SCREEN
```

The command lines below back up the data and structure of the customer table to the CUSTOMER.BUP file, limit the rows to those that are dated after 10/1/89, and order the rows by custid.

```
OUTPUT customer.bup
BACKUP ALL FOR customer WHERE datecol > 10/1/89 ORDER BY custid
OUTPUT SCREEN
```

The following command lines back up the structure of the customer table to the CUSTSTRC.BUP file, redirect the output to the screen, and close the file.

```
OUTPUT custstrc.bup
BACKUP STRUCTURE FOR customer
OUTPUT SCREEN
```

The following command lines back up the data from the producttable to the PRODUCT.BUP file, redirect the output to the screen, and close the file.

```
OUTPUT product.bup
BACKUP DATA FOR product
OUTPUT SCREEN
```

The following command lines show how to transfer a form to a new database. The first five command lines open the olddb database, set the null value to -0-, create the TEMPFORM.DAT file on drive D:, back up the data from the sys_forms2 table that contains the definition of the oldform form, and redirect the output to the screen. Command lines six through eight open the newdb database in order to transfer the oldform form definition (the database you open must not have a form named oldform), set the null value to -0-, and load the form definition from the backup file into the sys_forms2table.

```
CONNECT olddb
SET NULL -0-
OUTPUT d:tempform.dat
BACKUP DATA FOR SYS_FORMS3 WHERE SYS_FORM_NAME = 'oldform'
OUTPUT SCREEN
CONNECT newdb
SET NULL -0-
RUN d:tempform.dat
```

1.3.2 BEEP

Use the BEEP command to sound the computer's bell.

BEEP ['FREQ <Frequency>, <Duration>' 'FILE Filename.wav']

Options

Duration

Time in milliseconds for the bell to sound.

FREQ

Allows for user specified bell properties. **Note:** This feature is only applicable to Windows 2000/NT/XP/Vista/2003 and will be ignored by all other operating systems.

Frequency

A numeric value ranging from 10-65635 which represents the frequency of the bell in Hertz.

FILE

Full path to a WAV sound file to play.

About the BEEP Command

BEEP is useful to signal when a process has finished, or to signal that an error has occurred in a command file.

Example

In the following example, the first command line displays a message to indicate that a process has finished and instructs you to press a key to continue. The second command line sounds the computer's bell. The third command line stops processing until you press any key.

```
WRITE 'Processing is complete. Press any key to continue.'
BEEP
PAUSE
```

1.3.3 BREAK

Use the BREAK command to force an early exit from a WHILE...ENDWHILE loop or a SWITCH...ENDSW structure.

BREAK

About the BREAK Command

The BREAK command is usually run in an [IF...ENDIF](#) structure contained within a WHILE...ENDWHILE loop or a CASE block within a SWITCH...ENDSW structure. The IF conditions indicate when to run the BREAK.

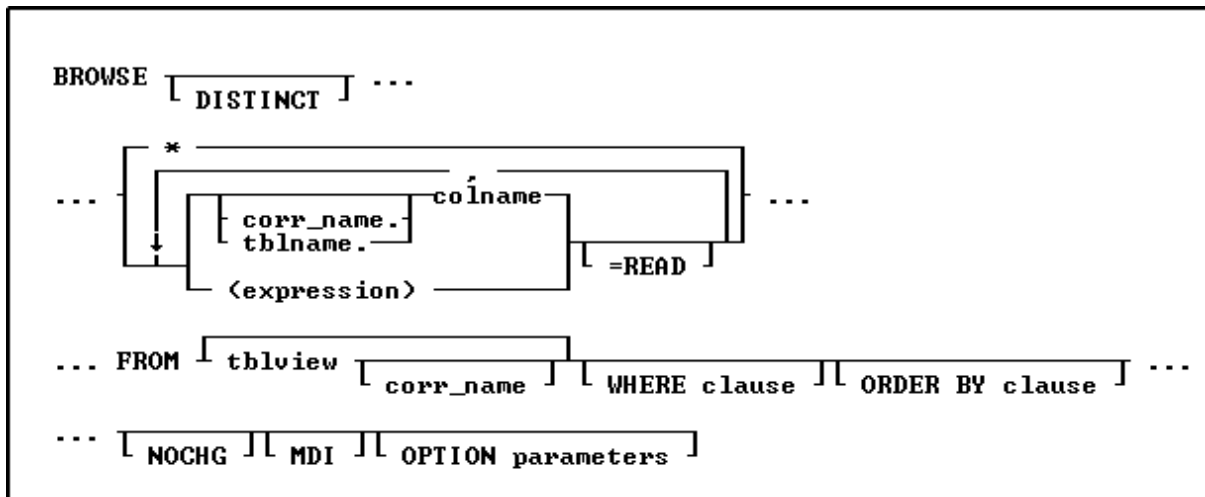
R:BASE exits the currently processing WHILE...ENDWHILE loop or SWITCH...ENDSW structure when a BREAK is encountered, and does not run any further commands in the WHILE loop or SWITCH structure. BREAK decreases the nesting level by one. BREAK passes control to the next line of the command file following the WHILE loop or SWITCH structure.

Examples

For an example of using BREAK with WHILE...ENDWHILE, see [WHILE...ENDWHILE](#). For an example of using BREAK with SWITCH...ENDSW, see [SWITCH...ENDSW](#).

1.3.4 BROWSE

Use the BROWSE command to display data.



DISTINCT

Suppresses the display of duplicate rows.

*

Specifies all columns.

,

Indicates that this part of the command is repeatable.

corr_name.

Correlation name. A nickname or alias for a table or view name. Use *corr_name* to refer to the same table twice within the command, or to explicitly specify a column in more than one table.

tblname.

Specifies a column using the table name. Use *tblname.** to specify all columns in the table.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter *#c*, where *#c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*). You can enter *tblname.** to specify all columns in the table.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

=READ

Specifies that the column is read-only, preventing changes to data.

FROM tblview

Specifies the table or view.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

NOCHG

Prevents changes to the issued BROWSE command; stops users from reaching additional data by adding columns to the command or changing the command's WHERE clause. When the NOCHG option appears in the command, users cannot toggle from BROWSE to EDIT by pressing [F4], nor by using the **Edit** menu in R:BASE for Windows. NOCHG is particularly useful when the BROWSE command is issued within an application or command file.

MDI

This option is used in R:BASE for Windows only; specifies a modeless window-a modeless window allows you to access other windows without closing the current window first.

OPTION parameters - provides the ability customize the Data Browser Window. These enhanced parameters are supported at the R> Prompt, in command files, EEPs and Stored Procedures.

WINDOW_STATE *value*

To setup Data Browser window state with the following supported values:

MAXIMIZED - show maximized
NORMAL - show normalized
MINIMIZED - show minimized

CAPTION *value*

To change Data Browser window caption.

LOCK *nnn*

To lock a number of columns in Data Browser window.

ROWVIEW

To open Data Browser window in "View as Row" mode.

READONLY *value*

To open Data Browser window in Read-Only mode. (Default is OFF)

- ON
- OFF

MODAL *value*

To open Data Browser as modal/non-modal window. (Default is ON)

- ON
- OFF

EXPORTRESULT *value*

To enable/disable the "Export Result to->" menu item. (Default is ON)

- ON
- OFF

SAVERESULT *value*

To enable/disable the "Save Data As->" menu item. (Default is ON)

- ON
- OFF

PRINTDATA *value*

To enable/disable the "Print Data" menu item. (Default is ON)

- ON
- OFF

About the BROWSE Command

The BROWSE command starts the Data Browser and displays the information you specify.

Examples

Example 01:

The following command displays all columns in the *transmastertable* and sorts the rows by the customer identification numbers contained in *custid* column.

```
BROWSE * FROM transmaster ORDER BY custid
```

Example 02:

The following command displays the *custid*, *company*, *custaddress*, *custcity*, *custstate*, and *custzip* columns in the *customer* table. Only the rows where the customer identification is equal to 100 are displayed.

```
BROWSE custid, company, custaddress, custcity, custstate, custzip FROM customer  
WHERE custid = 100
```

Example 03:

The following command displays the *custid* column and restricts editing. Do not place a delimiter, such as a comma or a space, between options prefaced with an equal sign when you are building a command.

```
BROWSE custid=read, company, custaddress, custcity, custstate, custzip FROM customer  
WHERE custid = 100
```

Example 04:

The following command open all records for the Customer table with a maximized Data Browser window and "Customer Data" as the caption.

```
BROWSE ALL FROM customer OPTION WINDOW_STATE MAXIMIZED|CAPTION Customer Data
```

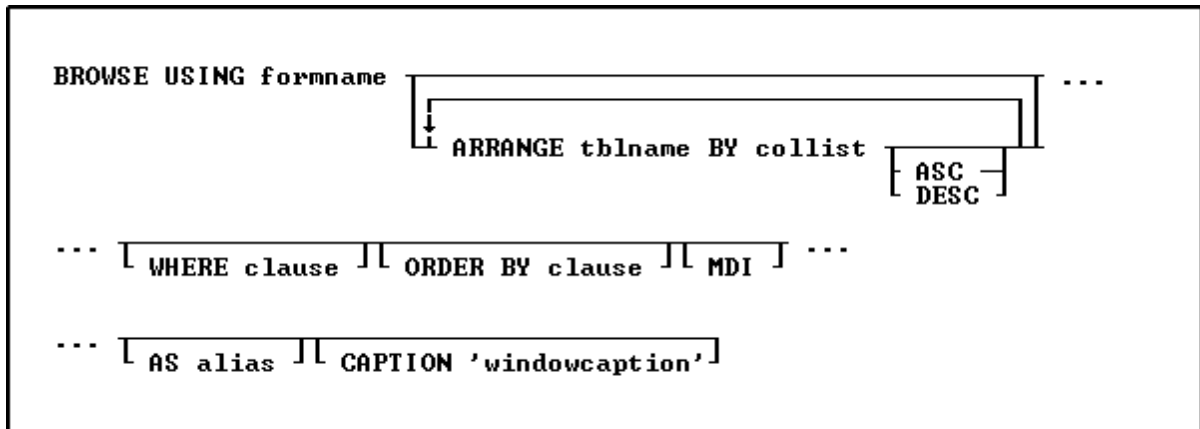
Example 05:

The following will start the Data Browser for *custid* 125, displaying the data as a single row in read only mode.

```
BROWSE ALL FROM customer WHERE custid = 125 OPTION WINDOW_STATE NORMAL|ROWVIEW|
      READONLY ON
```

1.3.5 BROWSE USING

The BROWSE USING FormName command will make all column and variable objects on the form READ ONLY.



Options

formname

Specifies a form name. In R:BASE eXtreme 9.0 (32), the form name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the form name is limited to 128 characters.

ARRANGE tblname BY collist

Sorts the rows displayed by a form's lower-level tables. Each lower-level table in the form that you want to sort requires its own ARRANGE clause. Now you can specify up to twenty-five tables to ARRANGE and up to five columns in each table on which R:BASE will sort.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

MDI

This option is used in R:BASE for Windows only; specifies a modeless form-a modeless form allows you to access other windows without closing the form first.

AS alias

This option is used in R:BASE for Windows only. Allows the user to specify a name for this instance of the form. This option is used in conjunction with the MDI option so that the user can programmatically specify the form with the [SETFOCUS](#) command.

CAPTION 'windowcaption'

This option is used in R:BASE for Windows only. Specifies a caption for the window title bar.

About BROWSE USING FormName

The BROWSE USING FormName command will make all column and variable objects on the form READ ONLY.

Example:

```
BROWSE USING CustomerEdit WHERE CustID = 100
```

Notes:

- The Form Settings must include the option of "Editing Data".
- Only Column and Variable Objects are supported for READ ONLY mode when using BROWSE USING FormName command.
- Same Form could be used for entering and editing data accordingly.
- Any code defined within Entry/Exit Procedures (EEPs) will execute regardless of the status of the form in BROWSE mode, however, no table updates will be allowed.

The database RRBYW15 contains a five-table form called *SalesTransactions*. The command below shows how the form's four subordinate tables, *InvoiceHeader*, *InvoiceDetail*, *Employee* and *Contact*, can be sorted using an ARRANGE clause. The ARRANGE clause sorts the second table, *InvoiceHeader*, by the *TransID* column, in ascending order, sorts the third table, *InvoiceDetail*, by the *DetailNum* column, in ascending order, sorts the fourth table, *Employee*, by *EmplName* and *EmpFName*, in ascending order and the fifth table, *Contact*, by the *ContLName* and *ContFName*, in ascending order.

```
CONNECT RRBYW15
```

```
BROWSE USING SalesTransactions +
ARRANGE InvoiceHeader BY TransID ASC, +
ARRANGE InvoiceDetail BY DetailNum ASC, +
ARRANGE Employee BY EmplName ASC, EmpFName ASC, +
ARRANGE Contact BY ContLName ASC, ContFName ASC +
ORDER BY CustID
```

1.4 C

1.4.1 CALL

Runs a Stored Procedure.

```
CALL procname( [ arglist ] )
```

Syntax

A) As function: SET VAR vVariable TYPE = (CALL procname(arglist))

B) As command: CALL procname(arglist)

About the CALL Command

The call command is used to invoke a Stored Procedure that was created by using the [SET PROCEDURE](#) command. It can be referenced by either a function notation or as a stand-alone command. In either case the argument list must be included. If you wish to include a blank argument list then use an empty pair of parenthesis.

Both methods of using CALL have their advantages. For example, using the function notation allows you to use a Stored Procedure in a computed view or to invoke a Stored Procedure via an SQL statement. On the other hand, used as a command, you will be able to reference the Stored Procedure by itself.

When using the Function notation the return value of the Stored Procedure is stored in the variable itself or displayed in the column (in the case of a computed column in a table or view). When using the Command notation the return value of the Stored Procedure will be placed into the system variable STP_RETURN. STP being an abbreviation of Stored Procedure.

The CALL command is supported in EEPs.

Examples

In the example below, a view is using a Stored Procedure to calculate values from another table.

```
CREATE VIEW MonthSum (CustomerID, CustomerSummary) +
AS SELECT T1.CustomerID, (CALL SumUpCust(T1.CustomerID)) FROM Customers T1
```

In the following example, an SQL select statement is used to invoke a maintenance routine from another application such as Visual Basic or R:WEB Suite. The use of WHERE LIMIT=1 causes the procedure to run once and only once. Without this clause the Stored Procedure would execute once for every matching row in the table. The *AnyTable* can be any table in the database in this case. The only requirement is that we must use a table in order to have a "healthy" [SELECT](#) clause.

```
SELECT (CALL DBCheck()) FROM AnyTable WHERE LIMIT=1
```

1.4.2 CASCADE

Use the CASCADE command to cascade all open windows.

CASCADE

1.4.3 CHDIR

Use the CHDIR command to display or change the current directory.

CHDIR [**d:**
pathname]

Options

d:

Specifies a valid drive letter.

pathname

Specifies one or more directory names that lead from the root directory to a specified directory.

About the CHDIR Command

R:BASE assumes you want to work in the current directory and drive. To display the current drive and directory, enter CHDIR without a path.

CHDIR is similar to the CHDIR command in the operating system; however, the R:BASE CHDIR command requires a space between the command and the directory. For more information about the CHDIR command, refer to your operating system manual.

CHDIR supports long (over 80 character) strings, and long strings with spaces.

Examples

Example 01:

The following command lines change the current directory to the SALEDEPT directory on drive C:.

```
CHDRV c:
CHDIR /saledept
```

Example 02:

The following command moves up one directory level and makes that directory the current directory.

```
CHDIR ..
```

Example 03:

At the R> Prompt or in a command file:

```
CD '\Documents and Settings\Administrator\Application Data\Microsoft\ +
    Internet Explorer\Quick Launch'
CHDIR
DIR
RETURN
```

Example 04:

At the R> Prompt or in a command file:

```
CD \FaxFiles\Archive\My_Very_Long_Corporation_Name\Service_Department\ +
    General_Pharmacy
CHDIR
DIR
RETURN
```

1.4.4 CHDRV

Use the CHDRV command to change the current drive.

CHDRV d:

Options

d:

Specifies a valid drive letter.

About the CHDRV Command

R:BASE assumes you want to use the current drive.

To display the current drive, use CHDRV without designating a drive.

Examples

The following command displays the current drive.

```
CHDRV
```

The following command makes drive A: the current drive.

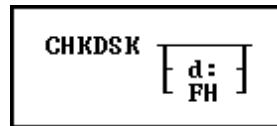
CHDRV a:

The following command makes drive C: the current drive.

C:

1.4.5 CHKDSK

Use the CHKDSK command to display information about an active local drive.



Options

d:

Specifies a drive other than the current drive.

FH

Displays the number of available file handles.

About the CHKDSK Command

CHKDSK lists the following:

- The total number of bytes of disk space.
- The number of bytes of disk space available for use.

To display information about another drive-not the current drive-specify the drive.

CHKDSK is similar to the CHKDSK command in the operating system; however, the operating system command can correct errors in the disk directory or file allocation table.

Use CHKDSK to check available file handles after opening a database, executing a command, running a file, or printing reports. In R:BASE, a maximum of 20 file handles can be open at one time. R:BASE uses four file handles for an open database, two file handles to sort information, two file handles for a temporary scratch file, one file handle for files, and one file handle for each of the following commands: [RUN](#), [DISPLAY](#), [CHOOSE](#), [OUTPUT](#) *file name*, and [TYPE](#).

Examples

The following command displays the total bytes available on the current disk.

```
CHKDSK
```

The display looks similar to the lines below.

```
10592256 bytes total disk space
774144 bytes available on disk
```

The following command checks the total disk space available for the disk in drive A.

```
CHKDSK a:
```

The following command displays the number of file handles currently available.

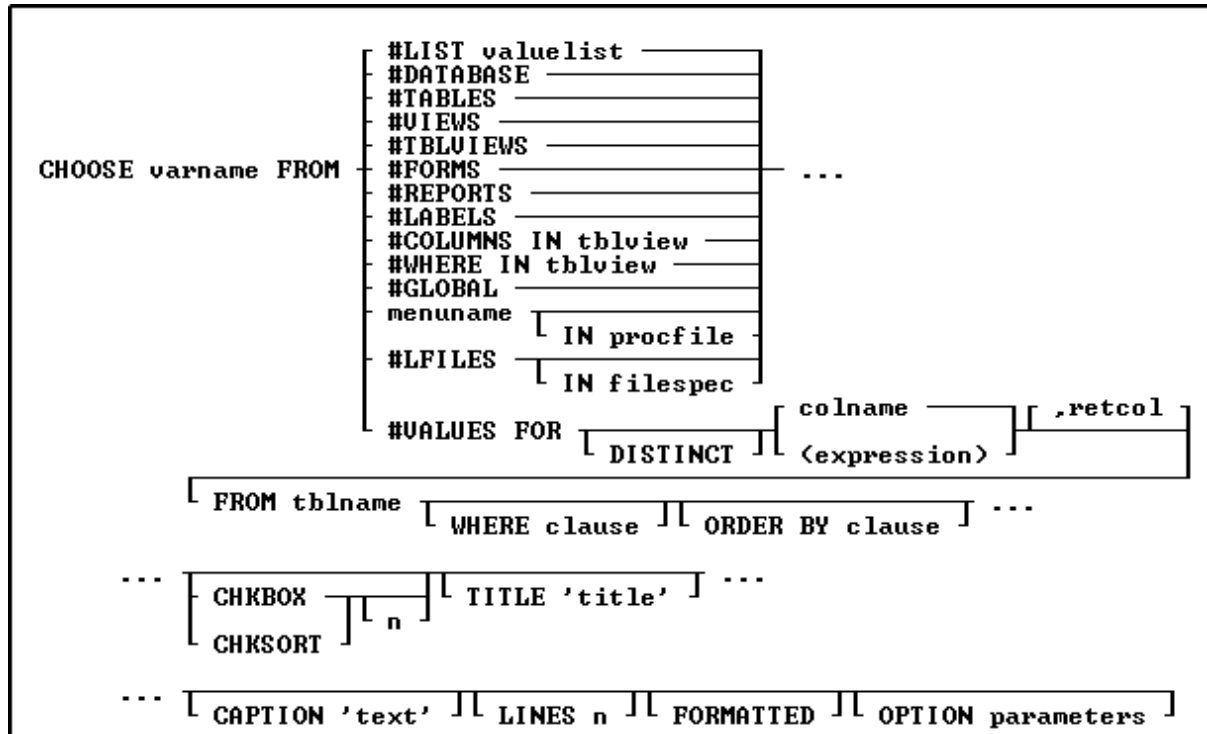
```
CHKDSK FH
```

The display looks similar to the lines below.

File handles: 9 10 11 12 13 14 15 16 17 18 19

1.4.6 CHOOSE

Use the CHOOSE command to display the following types of menus: bar with a pull-down, check-box, pop-up, sort check-box, vertical, or WHERE builder. For assistance with building your CHOOSE commands, refer to the [R:Choose Builder Plugin](#).



Options

#COLUMNS IN tblview

Displays a pop-up menu containing all columns in the specified table or view and stores the chosen column name in the variable varname.

#DATABASE

Displays a pop-up menu containing all databases in the current directory, and stores the chosen database name in the variable varname.

#FORMS

Displays a pop-up menu containing all forms in the open database and stores the chosen form name in the variable varname.

#GLOBAL

Displays a pop-up menu containing currently defined global variables and stores the chosen variable name in the variable varname.

#LABELS

Displays a pop-up menu containing all labels in the open database and stores the chosen label name in the variable varname.

#LFILES IN filespec

Displays all files in the current directory in a pop-up list, unless the IN filespec option is specified. In this latter case, the user may specify a filespec such as *.RTF, and R:BASE will display a pop-up list of all files in the specified directory with the .RTF extension. The chosen file name is stored in the variable varname.

#LIST valuelist

Allows you to specify a list of values in a comma delimited format. You can also use a variable that contains comma delimited values in the command. The group of values **MUST** be encapsulated in quotes **UNLESS** a variable is used.

#REPORTS

Displays a pop-up menu containing all reports in the open database and stores the chosen report name in the variable varname.

#TABLES

Displays a pop-up menu containing all tables in the open database and stores the chosen table name in the variable varname.

#TBLVIEWS

Displays a pop-up menu containing all tables and views in the open database and stores the chosen table or view name in the variable varname.

#VIEWS

Displays a pop-up menu containing all views in the open database and stores the chosen view name in the variable varname.

#WHERE IN tblview

Opens the WHERE builder for you to enter conditions for the columns in the specified table or view; stores the entire WHERE clause in the variable varname. Your [WHERE Clause](#) conditions can also pre-loaded into the WHERE Builder using the **OPTION WHERE_CLAUSE** parameter.

menuname

A user-defined menu can be stored either in an ASCII disk file or as a menu block in an R:BASE procedure file created by [CodeLock](#). If procfile is used, menuname refers to a procfile menu block. If procfile is not used, menuname specifies an ASCII disk file containing the menu to be displayed.

#VALUES FOR

Displays a pop-up menu containing values from the specified column or expression and table. Use the (expression) option to display more than one column from the table in the menu, such as (colname1 & colname2).

DISTINCT

Suppresses the display of duplicate rows.

colname

Specifies a column name. In a command, you can enter #c, where #c is the column number shown when the columns are listed with the LIST TABLES command. In an SQL command, a column name can be preceded by a table or correlation name and a period (tblname.colname).

<expression>

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as #DATE, #TIME, and #PI.

,retcol

Specifies the column whose value is returned into the variable varname. The column does not have to be one of the columns displayed in the menu.

FROM tblname

Specifies the table name.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

CHKBOX

Displays a menu that limits the number of selections the end user can make.

CHKSORT

Displays a menu that limits the number of selections the end user can make, with support to sort the selected results.

n

n is an optional positive integer specifying the maximum number of options on that menu that can be checked. If n is zero or is greater than the number of menu options, all options can be checked. If n is unspecified, the default value is zero. The maximum value of n is 9999.

TITLE 'title'

Displays a title in the dialog box

CAPTION 'text'

Displays text in the dialog box title bar

LINES n

Determines the number of lines, n, to display in the list box. The default is 10.

FORMATTED

Displays the CHOOSE box using a mono space font

Additional [OPTION parameters](#)

Additional parameters are available to increase the visual display of the CHOOSE window. To use the graphic CHOOSE Builder, choose "Utilities" > "Plugins" > "Internal Plugins" > "CHOOSE Builder" from the main Menu Bar. All OPTION parameters and values must be separated by the "|" (pipe) character.

Notes:

- CHOOSE will always be displayed at the center of your screen, unless specified using the TOP nn LEFT nn parameters.
- The resulting variable will always be TEXT
- The resulting variable will be left justified without leading spaces.

Example 1:

```
CLS
CHOOSE vCustIDTxt FROM #VALUES FOR +
(LJS(Company,40)&CustCity+', '&CustState),CustID +
FROM Customer ORDER BY Company +
TITLE 'Company Name and Location' +
CAPTION 'Colorful CHOOSE Example 01' LINES 20 FORMATTED
IF vCustIDTxt IS NULL OR vCustIDTxt = '[Esc]' THEN
    GOTO Done
ELSE
    SET VAR vCustID = .vCustIDTxt
ENDIF
__ Do what you have to do here with the variable CustID
LABEL Done
CLEAR VAR vCustIDTxt, vCustID
RETURN
```

Example 2:

```
CLS
CHOOSE vCustIDTxt FROM #VALUES FOR +
(LJS(Company,40)&CustCity+', '&CustState),CustID +
FROM Customer ORDER BY Company +
CHKBX 1 TITLE 'Company Name and Location' +
CAPTION 'Colorful CHOOSE Example 02' LINES 20 FORMATTED
```

```

IF vCustIDTxt IS NULL OR vCustIDTxt = '[Esc]' THEN
    GOTO Done
ELSE
    SET VAR vCustID = .vCustIDTxt
ENDIF
__ Do what you have to do here with the variable CustID
LABEL Done
CLEAR VAR vCustIDTxt, vCustID
RETURN

```

Example 3:

```

CLS
CHOOSE vCustIDTxt FROM #VALUES FOR +
(LJS(Company,40)&CustCity+', '&CustState),CustID +
FROM Customer ORDER BY Company +
TITLE 'Company Name and Location' +
CAPTION 'Colorful CHOOSE Example 03' LINES 20 FORMATTED +
OPTION List_Font_Color WHITE|List_Back_Color TEAL +
|Title_Font_Color TEAL|Title_Back_Color WHITE +
|Window_Back_Color WHITE|Title_Font_Size 24 +
|Title_Font_Name TIMES NEW ROMAN|Window_Caption
IF vCustIDTxt IS NULL OR vCustIDTxt = '[Esc]' THEN
    GOTO Done
ELSE
    SET VAR vCustID = .vCustIDTxt
ENDIF
__ Do what you have to do here with the variable CustID
LABEL Done
CLEAR VAR vCustIDTxt, vCustID
RETURN

```

Example 4:

```

CLS
CHOOSE vCustIDTxt FROM #VALUES FOR +
(LJS(Company,40)&CustCity+', '&CustState),CustID +
FROM Customer ORDER BY Company +
TITLE 'Company Name and Location' +
CAPTION 'Colorful CHOOSE Example 04' LINES 20 FORMATTED +
OPTION List_Font_Color WHITE|List_Back_Color TEAL +
|Title_Font_Color TEAL|Title_Back_Color WHITE +
|Window_Back_Color WHITE|Title_Font_Size 24 +
|Title_Font_Name TIMES NEW ROMAN|Window_Caption
IF vCustIDTxt IS NULL OR vCustIDTxt = '[Esc]' THEN
    GOTO Done
ELSE
    SET VAR vCustID = .vCustIDTxt
ENDIF
__ Do what you have to do here with the variable CustID
LABEL Done
CLEAR VAR vCustIDTxt, vCustID
RETURN

```

Example 05:

```

CLS
CHOOSE vCustIDTxt FROM #VALUES FOR +
(LJS(Company,40)&CustCity+', '&CustState),CustID +
FROM Customer ORDER BY Company +
CHKBOX 1 TITLE 'Company Name and Location' +
CAPTION 'Colorful CHOOSE Example 05' LINES 20 FORMATTED +
OPTION List_Font_Color WHITE|List_Back_Color TEAL +
|Title_Font_Color TEAL|Title_Back_Color WHITE +
|Window_Back_Color WHITE|Title_Font_Size 24 +
|Title_Font_Name TIMES NEW ROMAN|Window_Caption +
|Buttons_Back_Color WHITE
IF vCustIDTxt IS NULL OR vCustIDTxt = '[Esc]' THEN
    GOTO Done
ELSE
    SET VAR vCustID = .vCustIDTxt
ENDIF
__ Do what you have to do here with the variable CustID
LABEL Done
CLEAR VAR vCustIDTxt, vCustID
RETURN

```

Example 6:

```

CLS
CHOOSE vCustIDTxt FROM #VALUES FOR +
(LJS(Company,40)&CustCity+', '&CustState),CustID +
FROM Customer ORDER BY Company +
CHKBOX 1 TITLE 'Company Name and Location' +
CAPTION 'Colorful CHOOSE Example 06' LINES 20 FORMATTED +
OPTION List_Font_Color WHITE|List_Back_Color TEAL +
|Title_Font_Color TEAL|Title_Back_Color WHITE +
|Window_Back_Color WHITE|Title_Font_Size 24 +
|Title_Font_Name TIMES NEW ROMAN|Window_Caption +
|Buttons_Show_Glyph ON|Buttons_Back_Color WHITE
IF vCustIDTxt IS NULL OR vCustIDTxt = '[Esc]' THEN
    GOTO Done
ELSE
    SET VAR vCustID = .vCustIDTxt
ENDIF
__ Do what you have to do here with the variable CustID
LABEL Done
CLEAR VAR vCustIDTxt, vCustID
RETURN

```

Example 7:

```

CLS
CHOOSE vWhere FOR #WHERE IN Customer +
OPTION THEMENAME R:BASE Rocks! +
|WHERE_CLAUSE WHERE CustState = 'CA' ORDER BY COMPANY
RETURN

```

1.4.6.1 OPTION parameters

These parameters can be used to change the "look and feel" of a CHOOSE window. In addition to the parameters below, you can also alter the CHOOSE [Title](#), [List](#), and [Buttons](#).

WINDOW_CAPTION *value*

Customizes the window caption. Supported values:

- OFF - makes the window caption invisible.
- SMALL - makes the window caption small (tool window).

WINDOW_BACK_COLOR *value*

Changes the color of the WINDOW BACKGROUND area. User can specify the integer value or type one of the predefined values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

TOP *value*

Moves the CHOOSE box location, in pixels, from the top of the screen down.

LEFT *value*

Moves the CHOOSE box location, in pixels, from the left of the screen right.

WHERE_CLAUSE *value*

Allows you to pre-load a WHERE Clause into the WHERE Builder window when using "**#WHERE IN tblview**".

SHOW_ALL_BUTTON *value*

Displays a "Select All" button when using the [CHKBOX](#) option. Values:

- ON
- OFF

SINGLE_CLICK *value*

Limits the user to select a single option from the displayed list, and then immediately close. This option works only when user can select single item from a list. For multiple item selection this feature will be ignored. Values:

- ON
- OFF

THEMENAME *value*

Specifies one of 86 pre-defined Themes, or a custom Theme loaded into R:BASE. See Themes.

NOTE: All previously defined CHOOSE commands which are redefined to use Themes should be thoroughly checked prior to putting into production as objects and text may look substantially different.

1.4.6.1.1 Title

These parameters are specific to changing how the CHOOSE "title" is displayed.

TITLE_BACK_COLOR *value*

Changes the background color of the TITLE area. User can specify the integer value or use one of the predefined values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

TITLE_FONT_NAME *value*

Specifies a font name for the title area. Values would consist of the fonts available on the computer.

TITLE_FONT_SIZE *value*

Specifies a font size in title area

TITLE_FONT_COLOR *value*

Changes the font color of TITLE area. User can specify the integer value or type one of the predefined names:

Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

TITLE_BOLD *value*

Makes font in title area bold style. Values:

- ON
- OFF

TITLE_ITALIC *value*

Changes the font in title area italic style. Values:

- ON
- OFF

TITLE_UNDERLINE *value*

Changes the font in title area underlined. Values:

- ON
- OFF

TITLE_STRIKEOUT *value*

Makes the font in title area strikeout. Values:

- ON
- OFF

1.4.6.1.2 List

These parameters are specific to changing how the CHOOSE "list" is displayed.

LIST_BACK_COLOR *value*

Changes the background color of LIST area. User can specify the integer value or type one of the predefined values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

LIST_FONT_NAME *value*

Specifies a font name in LIST area. Values would consist of the fonts available on the computer.

LIST_FONT_SIZE *value*

Specifies a font size in LIST area

LIST_FONT_COLOR *value*

Changes font color of LIST area. User can specify the integer value or type one of the predefined values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

LIST_BOLD *value*

Makes the font in LIST area bold style. Values:

- ON
- OFF

LIST_ITALIC *value*

Makes the font in title area italic style. Values:

- ON
- OFF

LIST_UNDERLINE *value*

Makes the font in title area underlined. Values:

- ON
- OFF

LIST_STRIKEOUT *value*

Makes the font in title area strikeout. Values:

- ON
- OFF

1.4.6.1.3 Buttons

These parameters are specific to changing how the CHOOSE "buttons" is displayed.

BUTTON_OK_CAPTION *value*

Changes the caption for the "OK" button.

BUTTON_CANCEL_CAPTION *value*

Changes the caption for the "Cancel" button.

BUTTONS_SHOW_GLYPH *value*

Places images on the OK and Cancel buttons. Values:

- ON
- OFF

BUTTONS_BACK_COLOR *value*

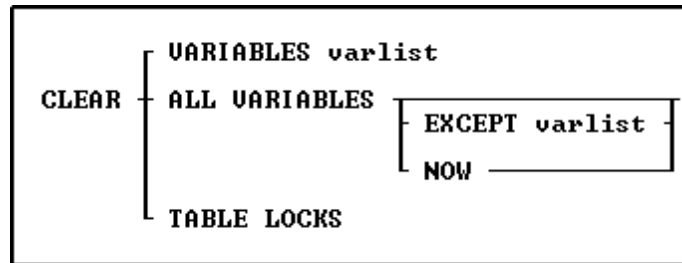
Changes the color of the BUTTON area. User can specify the integer value or type one of the predefined values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY

- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

1.4.7 CLEAR

Use the CLEAR command to remove global variables from memory or table locks.



Options

ALL VARIABLES

Removes all global variables from memory.

dbspec

Specifies the database name.

EXCEPT varlist

Specifies variables that the CLEAR command will not remove. You can use wildcards in variable names.

NOW

CLEAR ALL VAR NOW, the fourth argument to the CLEAR ALL VAR command clears all the variable storage blocks (VSBs), including all printer control code variables and other non-permanent system variables (NPSVs). It re-allocates the blocks with just the permanent system variables (PSVs), #DATE, #TIME, #PI and SQLCODE.

Without the fourth parameter, printer control code variables and other # variables are NOT cleared. Also, it does not completely re-initialize the memory blocks as does the CLEAR ALL VAR NOW.

TABLE LOCKS

Removes all locks on tables. This command parameter must be used with [MULTI](#) set to OFF and while connected to the database.

VARIABLES varlist

Removes a list of one or more variables. Use this option at the end of a complete set of procedures to clear variables that are no longer needed. You can use wildcards in variable names.

About the CLEAR Command

When R:BASE is first loaded into memory, only system variables are defined; they are not affected by the CLEAR command. Variables defined in reports are cleared automatically after the report is printed. Other variables you define remain in memory until you exit from R:BASE or use the CLEAR command.

Examples

The following command removes the global variables *vcounter* and *vname* from memory.

```
CLEAR VARIABLES vcounter, vname
```

The following command removes all global variables from memory.

```
CLEAR ALL VARIABLES
```

The following command removes all global variables except *var1*.

```
CLEAR ALL VARIABLES EXCEPT var1
```

The following command clears all variables beginning with the letter *v*.

```
CLEAR VARIABLES v%
```

1.4.8 CLOSE

Use the CLOSE command to close an open cursor.

CLOSE cursor

Options

cursor

Closes a row pointer in a table that was defined with the [DECLARE CURSOR](#) command.

About the CLOSE Command

Cursors are pointers to rows in a table, and are defined using the DECLARE CURSOR command. When you no longer want to use the cursor but want to retain it for later use, use the CLOSE command to close the cursor. When you open the cursor again, it is positioned at the beginning of the set of rows defined by the DECLARE CURSOR command.

Use the [LIST CURSOR](#) command to list all of the currently defined cursors and whether the cursor is opened or closed.

When you close a cursor, most of the memory taken by the cursor definition is returned. If the DECLARE CURSOR command used any file handles, they are released.

Example

The following command makes the rows defined by the DECLARE CURSOR command as *cursor1* unavailable. To use the information defined by *cursor1* again, you need to reopen the cursor with the [OPEN](#) command.

```
CLOSE cursor1
```

1.4.9 CLOSEWINDOW

Use the CLOSEWINDOW command to close a form from within an application or entry/exit procedure (EEP). Without the alias it closes the current form.

CLOSEWINDOW [alias]

Options

alias

Specifies a window name, which is the name given to the instance of an MDI form started with the "AS *alias*" option when using the [ENTER](#), [EDIT USING](#), or [BROWSE USING](#) commands.

1.4.10 CLS

Use the CLS command to clear the R> Prompt window or [PAUSE](#) message box that is currently displayed. The additional options below pertain only to DOS versions of R:BASE.

```
CLS [ FROM scrnrow, scrncol [ TO scrnrow, scrncol ] [ backcolor ]
```

Options

backcolor

Specifies the background color of the cleared area. Valid R:BASE background colors are black, blue, brown, cyan, green, gray, magenta, and red. In Windows you can also use yellow and white.

FROM scrnrow, scrncol

Specifies the first screen row to clear, starting at the column specified by *scrncol*.

TO scrnrow, scrncol

Specifies the last screen row to clear, ending at the column specified by *scrncol*.

About the CLS Command

CLS does not issue a form feed. (The [NEWPAGE](#) command does issue a form feed.)

Example

The following example clears the window from row 7 column 5 to row 9 column 60, and sets the background color in the cleared region to red.

```
CLS FROM 7,5 TO 9,60 RED
```

1.4.11 CODELOCK

Use the CODELOCK command to start CodeLock, the ASCII-to-binary conversion program that locks command and procedure files.

```
CODELOCK [ 1 cmdfile backfile binfile ] [ 2 cmdfile apxfile procname ] [ 3 scrfile apxfile screenname ] [ 4 menufile apxfile menuname ] [ 5 appfile apxfile ] [ -p ]
```

Options

1

Corresponds to the first menu choice in CodeLock: "Convert an ASCII command file to a binary command file."

2

Corresponds to the second menu choice in CodeLock: "Add an ASCII command file to a procedure file."

3

Corresponds to the third menu choice in CodeLock: "Add an ASCII screen file to a procedure file."

4

Corresponds to the fourth menu choice in CodeLock: "Add an ASCII menu file to a procedure file."

5

Corresponds to the fifth menu choice in CodeLock: "Convert an ASCII application file to a binary procedure file."

appfile

Specifies an ASCII application file.

apxfile

Specifies the name for the binary procedure file that is produced.

backfile

Specifies the name for the backup file.

binfile

Specifies the name for the binary command file that is produced.

cmdfile

Specifies an ASCII command file.

menufile

Specifies an ASCII menu file.

menuname

Specifies the name for the inserted menu block.

-p

Forces CodeLock to use the older encryption method (used previous to R:BASE version 4.5 Plus!). This option is necessary if the application is run using R:BASE version 4.5 or earlier.

procname

Specifies the name for the inserted command block.

screenname

Specifies the name for the inserted screen block.

scrfile

Specifies an ASCII screen file.

About the CODELOCK Command

CodeLock is an ASCII-to-binary conversion program that encodes command and procedure files, and creates an .APX file from the source file. CodeLock creates files that can only be run-not viewed or edited.

The advantages to using files encoded by CodeLock are:

- The encoded file that R:BASE creates protects the source file from being altered because the binary files produced by CodeLock cannot be changed with a text editor.
- The run time for the encoded file might be reduced because the commands are already parsed or interpreted by CodeLock; some of the work the computer does to run the ASCII form of the command file is not necessary. Also, calls between binary blocks in procedure files can be done without opening and closing a different disk file for every call.
- A developer can create a complex application that can be distributed in one file instead of several stand-alone files.

In R:BASE for Windows, when [MESSAGES](#) is set off, R:BASE does not display the file that is being encoded or any system messages while the file is being encoded. When MESSAGES is set on, the application file is displayed on screen as it is encoded.

When you make additions or changes to an .APX file, the additions and changes are appended to the existing .APX file. If you make regular changes to an .APX file, you should make a new .APX file to avoid the .APX file from getting too large.

Because CodeLock is only run from within R:BASE, R:BASE uses the current QUOTES setting while encoding an application. If R:BASE encounters a [SET QUOTES](#) command while encoding an application, R:BASE will use that QUOTES setting until it encounters another SET QUOTES command, or is finished encoding the application. When R:BASE finishes encoding the application, it resets the QUOTES setting to the setting that was in effect before CodeLock began.

1.4.12 COMMENT

Use comments in command or application files to provide internal program documentation.

There are a few comment designators: "--", "{ }", and "()".

1. A "--" comment can be used only on a single line either by itself or following a command.

To comment an individual line, add two hyphen characters "--" to the beginning of the line. A carriage return at the end of the line indicates the end of the text for a comment that begins with two hyphens. In R:BASE Editor, the syntax highlighting will alter the display and change the font color to pink and the style to italicized. In the following example;

```
CLEAR VAR vResult
```

the command will become:

```
--CLEAR VAR vResult
```

2. A "{ }" comment may share a command line with a command, occupy a line itself, or extend over multiple command lines.

This designator is the recommended option with the latest releases of R:BASE, as it helps in avoiding any confusion when using parentheses with your R:BASE expressions. The set of squiggly brackets "{ }", with the desired commented text or commands enclosed within the squiggly brackets will comment the text. In the following example;

```
PLUGIN RPDFMerge.RBL 'vResult +
|ACTION MERGE +
|DOC_LIST_FILE PDFFilesToMerge.LST +
|SHOW_SETUP_DIALOG ON +
|OUTPUT_FILE OneBigMergedFile.PDF '
```

the command(s) will become:

```
{
PLUGIN RPDFMerge.RBL 'vResult +
|ACTION MERGE +
|DOC_LIST_FILE PDFFilesToMerge.LST +
|SHOW_SETUP_DIALOG ON +
|OUTPUT_FILE OneBigMergedFile.PDF '
}
```

Keep in mind that any command(s) that is embedded within a multiple-line comment will not be executed.

3. A **"*()"** comment may share a command line with a command, occupy a line itself, or extend over multiple command lines.

Another use of characters that will comment your code is the asterisk character preceding a set of parentheses **"*()"**, with the desired commented text or commands enclosed within the parentheses. R:BASE interprets text following an asterisk and left parenthesis as a comment until a closing right parenthesis is reached. If the right parenthesis is not entered, R:BASE responds with a continuation prompt (**+>**). Enter a closing parentheses until you are returned to the R> Prompt, or other processing. In the following example;

```
PLUGIN RPDFMerge.RBL 'vResult +
|ACTION MERGE +
|DOC_LIST_FILE PDFFilesToMerge.LST +
|SHOW_SETUP_DIALOG ON +
|OUTPUT_FILE OneBigMergedFile.PDF '
```

the commands will become:

```
*(
PLUGIN RPDFMerge.RBL 'vResult +
|ACTION MERGE +
|DOC_LIST_FILE PDFFilesToMerge.LST +
|SHOW_SETUP_DIALOG ON +
|OUTPUT_FILE OneBigMergedFile.PDF '
)
```

Although this option is still supported in R:BASE, it is now recommended that you use the squiggly bracket **"{}"** method above. Keep in mind that any command(s) that is embedded within a multiple-line comment will not be executed.

Using Comment Designators

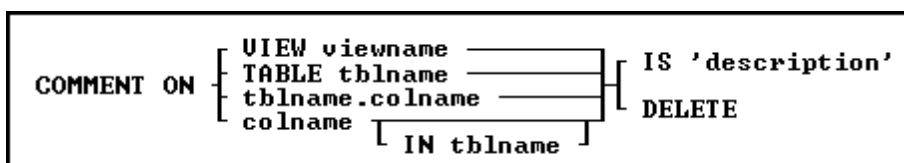
- Although used primarily in command files, you can enter a comment at the R> Prompt.
- If you place a comment on the same line as a command, leave at least one space between the comment and the command so the comment is not interpreted as part of the command.

Restrictions on Using the Comments

- Do not include comments within the text of an ASCII menu file or a menu block because the comment will be read as part of the file.
- Do not embed comments within multi-line commands between continuation characters.

1.4.13 COMMENT ON

Use the COMMENT ON command to add a description to a table or column.



Options

colname

Adds a description for a column in all tables in which it appears.

DELETE

Removes a description for a table or for a column in either the specified table or in all tables.

IN tblname

Adds a description for a column only in the specified table.

IS 'description'

Defines a description for a table or for a column in either the specified table or in all tables. The text must be enclosed in quotes using the current [QUOTES](#) setting.

TABLE tblname

Adds a description for the specified table.

VIEW viewname

Adds a description for the specified view.

tblname.colname

Specifies a column name. In a command, you can enter *#c*, where *#c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*).

About the COMMENT ON Command

A COMMENT ON description can be a maximum of 80 characters. If the description is longer, R:BASE displays only the first 80 characters. Descriptions must be enclosed in single quotation marks ('), or the current delimiter character for QUOTES.

When you add a description to a column that appears in multiple tables, R:BASE adds the description to the column in every table. If you add a new table containing the column, you must add the description for that column to the new table.

Use the [LIST](#) command to list the descriptions of the tables and columns. R:BASE also displays the description when you modify a table using the Database Designer. Comments are stored in the *sys_comments* system table. When a column or table is renamed or removed, R:BASE automatically updates the *sys_comments* table to reflect the change.

When access rights for a table have been assigned using the [GRANT](#) command, COMMENT ON requires the database owner's user identifier to describe tables and columns.

Examples

The following command adds a description to the *employeetable*.

```
COMMENT ON TABLE employee IS 'employee information'
```

The following command adds a description to the *empid* column in all tables in the database.

```
COMMENT ON empid IS 'employee identification number'
```

The following commands show two ways to add a description to the *empid* column in only the *employee* table.

```
COMMENT ON employee.empid IS 'employee identification number'  
COMMENT ON empid IN employee IS 'employee identification number'
```

The following command removes the description from the *employeetable*.

```
COMMENT ON TABLE employee DELETE
```


The following command removes the description from the *empid* column in every table in which the column occurs.

```
COMMENT ON empid DELETE
```

The following command removes the description from the *empid* column in the *employee* table.

```
COMMENT ON empid IN employee DELETE
```

1.4.14 COMMIT

Use the COMMIT command to make permanent the changes to the data or database structure made since the beginning of the transaction. COMMIT is used during transaction processing.

```
COMMIT [ WORK ]
```

Options

WORK

WORK is optional and has no effect in R:BASE.

About the COMMIT Command

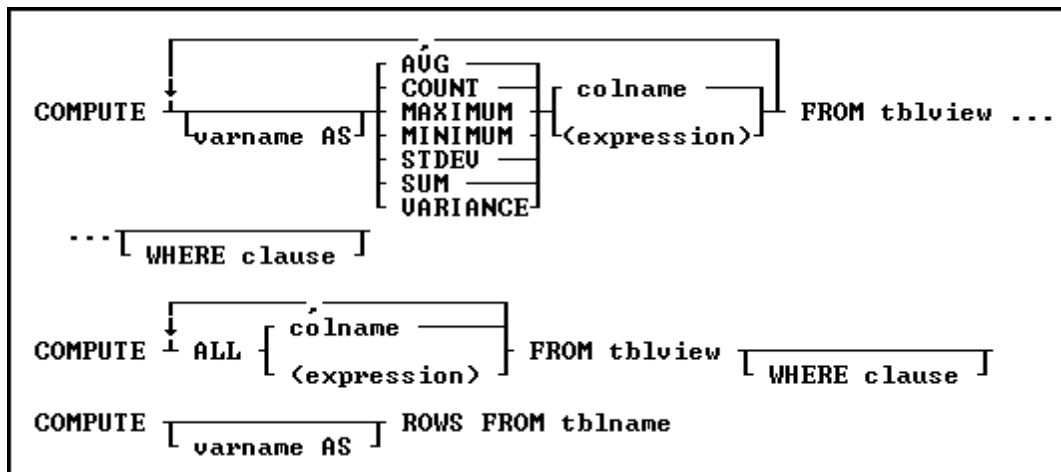
A transaction consists of all the commands executed since the last COMMIT or [ROLLBACK](#) command, or since you connected to the database if a COMMIT command has not been executed.

If you want to be able to process a group of commands, or transaction, without permanently affecting your data until you accept the changes, turn the [AUTOCOMMIT](#) setting off. R:BASE stores the information needed to reverse each modification made to the database in a Before Image file.

To accept a transaction and make the changes permanent, enter the COMMIT command after you have entered the transaction. The changes made by the commands in the transaction are made permanent and R:BASE clears the Before Image file to prepare it for the next transaction. The COMMIT command also releases all automatic table and database locks and executes any [SET LOCK OFF](#) commands included in the transaction. To reverse the transaction and undo the changes, enter the ROLLBACK command.

1.4.15 COMPUTE

Use the COMPUTE command to calculate expressions or perform mathematical operations on columns.



Options

'
Indicates that this part of the command is repeatable.

ALL

Executes all of the functions relevant to the data type of the selected column.

AVG

Computes the numeric average of CURRENCY, DOUBLE, NUMERIC, REAL, INTEGER, DATE, or TIME data types. R:BASE rounds averages of integer values to the nearest integer value and currency values to their nearest unit.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter *#c*, where *#c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*).

COUNT

Determines how many non-null entries there are for a particular column item.

(expression)

Determines the value for the column using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

FROM tblview

Specifies the table or view.

MAXIMUM

Selects the maximum value in a column. For TEXT and CHARACTER data types, R:BASE evaluates the first 30 characters. This option accepts columns with any data type except BIT, VARBIT, VARCHAR, and BITNOTE.

MINIMUM

Selects the minimum value in a column. For TEXT and CHARACTER data types, R:BASE evaluates the first 30 characters. This option accepts columns with any data type except BIT, VARBIT, VARCHAR, and BITNOTE.

ROWS FROM tblname

Counts rows in a specified table.

STDEV

Computes standard deviation for CURRENCY, DOUBLE, REAL, NUMERIC, or INTEGER data types. The standard deviation is a measure of how widely values are dispersed from the average value.

SUM

Computes the numeric sum of CURRENCY, DOUBLE, REAL, NUMERIC, or INTEGER data types.

VARIANCE

Determines variance for CURRENCY, DECIMAL, DOUBLE, FLOAT, REAL, NUMERIC, INTEGER, or SMALLINT data types.

varname AS

Specifies a global variable name in which to store the result of a COMPUTE command. If you use the arguments COUNT or ROWS, the variable must be an INTEGER data type.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the COMPUTE Command

R:BASE does not consider null values when it calculates averages, minimums, maximums, counts, sums, standard deviation, and variance.

The COMPUTE command can define a global variable with the data type that matches the results of the computation of a column's values.

You can also choose **Calculate** on the **Data Browser** menu in R:BASE for Windows to choose a column, then perform the mathematical operations that apply to the data type of the column.

In general we do recommend using the [SELECT](#) Function(Column) notation over the COMPUTE notation as this provides greater compliance to the SQL standard and greater portability to new versions of R:BASE.

Examples

The following command computes, then stores the minimum on-hand quantity in the *vminvar* variable found in the *prodlocation* table.

```
COMPUTE vminvar AS MINIMUM onhand FROM prodlocation
```

The following command computes the total number of rows in the *transmaster* table and places the result in the *vrowcount* variable.

```
COMPUTE vrowcounter AS ROWS FROM transmaster
```

The following command computes the average of an INTEGER data type (*onhand*) and returns the result as a DOUBLE data type.

```
COMPUTE AVG (onhand * 1.0) FROM prodlocation
```

The following command calculates your multiple computations on different columns. The command counts the number of entries in the *custid* column and averages the entries in the *netamount* column. The *custid* column and the *netamount* column are both from the *transmaster* table; all the specified columns must be in the same table or view.

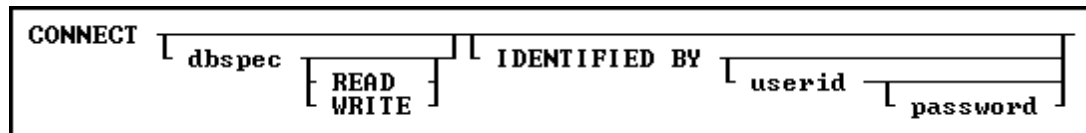
```
COMPUTE COUNT custid, AVG netamount FROM transmaster
```

The command below shows you how to use variables to hold the results of computations.

```
COMPUTE v1 AS COUNT custid,v2 AS AVG netamount FROM transmaster
```

1.4.16 CONNECT

Use the CONNECT command to open a database.



Options

dbspec

Opens a specified database.

IDENTIFIED BY

Specifies the user identifier. If left blank, R:BASE prompts you for the user identifier. R:BASE does not display it as you enter the text.

password

Optional; verifies the correct user identifier of the current user of the database.

READ

Opens a database as read-only. When you open a database as read-only, you cannot modify the data or database structure. All users must connect to the database in read mode if it is used.

userid

Specifies the user identifier. When access rights have been assigned using the [GRANT](#) command, you must enter an appropriate user identifier with the CONNECT command unless permission has been granted to PUBLIC.

WRITE

Opens a database in write/read mode. When you open a database in write/read mode, you can view and modify the database. This mode is the default when you use the CONNECT command without options.

About the CONNECT Command

R:BASE opens a database according to the current settings for the following SET commands: [FASTFK](#), [FASTLOCK](#), [MULTI](#), [ROWLOCKS](#), [TRANSACTIONS](#), and [STATICDB](#). Only one database can be open at a time.

You must open a database to use it; however, a database cannot be open when you copy, rename, or erase the database files.

When you use the CONNECT command without any of its options, R:BASE displays a list of the databases in the current directory. To open a database, highlight the database name and press [Enter]. This differs from older versions of R:BASE which required you to list a database in the connection command.

To open a database that is not in the current directory, add the drive and directory before you enter the database name, or use the [CHDIR](#) command before using CONNECT.

Database Access Rights with CONNECT

When you use the IDENTIFIED BY option, R:BASE identifies your level of access to the database. Only the last user identifier specified is in effect. If your user identifier contains spaces, enclose it in quotation marks when you enter it.

Closing a Database

R:BASE closes a database when you:

- Open another database.
- Issue a DISCONNECT command.
- Issue an EXIT command.
- Define a new database.

Updating Database Timestamps

When a database is created or opened, a timestamp is encoded in each of the four database files, enabling the database files to be stored in separate directories apart from one another. To ensure that the correct database files are accessed, R:BASE evaluates the timestamp values of the files as it searches for them on the path statement. R:BASE tries to connect to File 1 first, then it searches the path for the other database files.

Updating the encoded timestamp values in the four database files so that they all have the same timestamp values might be necessary if R:BASE is halted or if your computer or operating system crashes. When you try to connect to a database that is out of synchronization, R:BASE displays an error message. The [RBSYNC](#) command must be run.

Examples

The following command opens the *letters* database located on drive C: in the TOOLS/WP directory. R:BASE opens *letters* for reading and writing.

```
CONNECT c:\tools\wp\letters
```

The following command opens the *mydata* database with *user two* as the user identifier. The user

identifier is enclosed in quotation marks because it contains a space.

```
CONNECT mydata IDENTIFIED BY 'user two'
```

The following example displays all databases in the current directory and allows you to select the database you wish to connect to.

```
CONNECT
```

1.4.17 CONTINUE

Use the CONTINUE command to move to the next occurrence of the [WHILE](#) loop and run the code.

CONTINUE

Example

In the following example, when the code is run, processing returns to line 3 after it completes the CONTINUE command on line 6. The while-block commands in line 8 are not run.

```
SET VARIABLE v1=0
SET VARIABLE v2=1
WHILE v1 = 0 THEN
  *(while-block commands)
  IF v2 <> 0 THEN
    CONTINUE
  ENDIF
  *(while-block commands)
ENDWHILE
```

1.4.18 CONVERT

The CONVERT command was used to convert a 4.5 database to R:BASE 7.6. This command is currently not supported in R:BASE eXtreme 9.0.

```
CONVERT dbname [ IDENTIFIED BY ownername [ ownerpassword ] ]
```

Options

dbname

Specifies the database to be converted.

IDENTIFIED BY ownername

Specifies the ownername; required if one has been defined.

ownerpassword

Optional; verifies the correct ownername for the database.

About the CONVERT Command

R:BASE requires the conversion of your existing 5.5 or lower R:BASE database. Once the database is converted, it CANNOT be accessed by any previous version of R:BASE. Be sure and backup your database before you convert it.

1.4.19 COPY

Use the COPY command to copy files.

COPY filespec1 filespec2

Options

filespec1

Specifies the file to be copied-the source file.

filespec2

Specifies the file to receive the copy-the target file.

About the COPY Command

This command is similar to the operating system COPY command.

When you copy files from one drive to another, include the drive letter of the drive you want to copy to.

When you copy files from one directory to another directory, the name of the directory you want to copy to must be different from the current directory.

Naming Copied Files

When you copy a file from one drive or directory to another drive or directory, you can keep the same name for the new file, or you can give the file a new name. If you want to make a copy of a file on the same directory, you must give the file a different name. If the new name is the name of an existing file, COPY replaces the existing file with the file you are copying.

Using Wildcards with the COPY Command

When copying a group of files, you can use the operating-system wildcards (? or *) in the file specification. The settings for the R:BASE special characters SINGLE and MANY do not affect the operating system wildcards.

When the target name is the same length as the source name, the command succeeds. However, if the target name is shorter or longer than the source name, the results are unpredictable.

Backing Up with the COPY Command

You can use the COPY command to back up your database, command files, and programs. Copy the four .RX* files as a set.

If you change the .RX* extension when you use COPY to back up your database files, you must change the extension back to .RX* to use the database in R:BASE. The COPY command does not copy files onto multiple disks. If your database is too large to fit on one disk, use the [BACKUP](#) or [UNLOAD](#) commands.

Suggestion

On a workstation with multiple drives (local or mapped), especially when the files are on the different drive, it is always the best practice to define a drive letter when copying, deleting, renaming or running files, unless the specified files are located in the working directory. You will not need to specify the drive letter if all of the files are located in the default directory when using the copy, delete, rename or run commands.

Examples

The following command copies the file THISFILE to THATFILE on the current drive and directory.

```
COPY thisfile thatfile
```

The following command copies the MYDATA file on drive C: to the current directory on drive B: using the same file name.

```
COPY c:mydata b:mydata
```

The following command copies any files in the DBS directory that begin with MYDATA and have the extension .RX? to the ADMIN directory on the current drive using the same file names. The files with the .RX? extension could be the four disk files that contain an R:BASE database: MYDATA.RX1, MYDATA.RX2, MYDATA.RX3, and MYDATA.RX4.

```
COPY /dbs/mydata.rx? /admin
```

The following command copies the MYDATA database files using the new name SAVDAT.

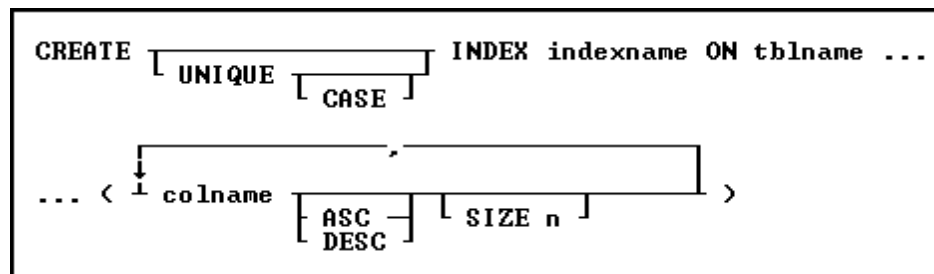
```
COPY mydata savdat
```

The following command copies the MYDATA database files with an .RX? extension and changes the extension to .SV?

```
COPY mydata.rx? mydata.sv?
```

1.4.20 CREATE INDEX

Use the CREATE INDEX command to speed up data retrieval by creating pointers that locate rows in a table easily.



Options

,
Indicates that this part of the command is repeatable.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

CASE

Specifies that the data values will be case sensitive.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters. In a command, you can enter #c, where #c is the column number shown when the columns are listed with the [LIST TABLES](#) command.

INDEX indexname

Specifies an index, which is displayed with the [LIST INDEX](#) command. An *indexname* is required.

ON tblname

Specifies the table in which to create an index for a column.

SIZE n

Sets the minimum number of characters to preserve to determine uniqueness during hashing. This

number can be a maximum of 196 characters. The index is created with the first n characters preserved and the rest of the value stored as a 4-byte hashed representation.

UNIQUE

Requires the values in a column to be unique.

About the CREATE INDEX Command

An index creates pointers to rows in columns, which allows R:BASE to find the rows using pointers much faster than searching the rows of data sequentially. You can index a column of any data type. An indexed column improves the performance of the following commands, clauses, and operations.

Commands, Clauses, and Operations to Use with Indexes	
DELETE DUPLICATES	RULES
INTERSECT	SELECT (when it includes a WHERE or an ORDER BY clause)
JOIN	SUBTRACT
Look-up expressions in Forms or Reports	UNION
ORDER BY	WHERE
PROJECT	

Although indexes speed up processing, they might slow down data entry because building an index for each value as it is entered takes time. Creating indexes for columns that contain many duplicate values does not always speed up processing. Indexes also occupy space on a disk.

Null Values

An indexed column can contain null values, but R:BASE uses an index most efficiently if each row in the indexed column contains a value. Primary keys, unique keys, or unique indexes explicitly restrict the insertion of null values. For other indexes, you can define a rule to ensure that a column always contains a value.

UPDATE Permission

When access rights for a table have been assigned using the [GRANT](#) command, you must have UPDATE permission for the column you want to index.

Indexing Criteria

Some columns are better candidates than others for indexing. To receive the greatest benefit from indexes, use the following criteria to help you decide which type of column is the best choice for indexing your table(s):

Primary Key

R:BASE automatically indexes the column(s) that is defined as the table's primary key.

Foreign Key

R:BASE automatically indexes the column(s) that is defined as the table's foreign key.

Columns Used in Queries

Columns that are not primary or foreign keys but are frequently used in queries should be indexed. Create a unique key constraint for columns that are not primary or foreign keys, but which uniquely identify a row in the table.

Columns Frequently Using ORDER BY or GROUP BY

Include a column in an ascending-order index when the column is not a primary or foreign key but is frequently referenced in an ascending-column ORDER BY or [GROUP BY](#) clause. Similarly, include a column in a descending-order index when the column is frequently referenced in a descending-column ORDER BY clause.

Full- and Partial-Text Indexes

Text columns can make effective indexed columns. If the size of the column that has a TEXT data type is 200 bytes or less, R:BASE creates a full-text index. A full-text index is an index that stores the entire contents of a column as an index in File 3, which is the file that contains indexes to columns. If the size of the column is greater than 200 bytes, R:BASE creates a partial-text index.

If you specify the SIZE option to be less than the defined length of a column, R:BASE creates a partial-text index, and any text column that has a defined length over 200 bytes must be a partial-text index. For columns that have a TEXT data type and exceed 200 bytes, you can specify the SIZE option to be between 0 and 196 to create a partial-text index. Specifying the size allows you to base your index on a specified number of characters at the beginning of the columns and to hash the remaining characters. For example, you can index a 225-character column with a TEXT data type by specifying the SIZE option to be any number less than 197 bytes. R:BASE will create an index with the first *n* characters and the rest of the value will be stored as a four-byte hashed representation of the text.

Partial-text indexes minimize storage space. However, partial-text indexes might not be as efficient as a full-text index, for example:

```
CREATE TABLE cities (cityname TEXT(40), state TEXT(2), country +
TEXT(20))
CREATE INDEX cityindex ON cities (cityname, state)
INSERT INTO cities VALUES('Bellevue', 'WA', 'USA')
INSERT INTO cities VALUES('Belltown', 'PA', 'USA')
SELECT cityname, state from cities WHERE cityname = 'Bellevue'
```

In the above example, because the query reads data only from the index named *cityindex*, there is no need to read the actual data stored in File 2-which is the data file-so the query is done quickly. The query is an index-only retrieval and produces fast results.

If a partial-text index was used in the same query as above, the partial-text index could also only use the index named *cityindex*. Because the partial-text index only preserves the first four characters, it is impossible to return the correct answer to the query from the index. The query, as shown below, would slow processing because R:BASE must read data from the R:BASE data file.

```
CREATE INDEX cityindex ON cities (cityname SIZE 4, state)
SELECT cityname, state from cities WHERE cityname = 'Bellevue'
```

When creating text indexes, be aware of the following:

- If you omit the SIZE option and the text field in the column is greater than 200 bytes, R:BASE creates a partial text index by storing the first 32 bytes of each field and hashing the remaining bytes in each field into a four-byte numeric representation of the text. For example, if the text is 280 bytes and you do not specify a size, R:BASE stores the first 32 bytes of each field and hashes the remaining 248 bytes into a four-byte integer.
- If you specify the SIZE option to be 16 bytes for a 60-byte column with a TEXT data type, R:BASE stores the first 16 bytes of each 60-byte text field and hashes the remaining bytes in each field into a four-byte numeric representation of the text. The total length of each index entry will be 20 bytes (16 + 4).
- If you specify the SIZE option to be 30-bytes for a 250-byte column with a TEXT data type, R:BASE stores the first 30 bytes of each 250-byte field and hashes the remaining bytes in each field into a four-byte numeric representation of the text. The total length of each index entry will be 34 bytes.
- If you specify the SIZE option to be 250 bytes for a column with a TEXT data type, you have made an illegal request because the maximum value for the SIZE option is 196 bytes when the length of the text field is greater than 200 bytes. If you specified the SIZE option to be 196 bytes for a 250-byte column, R:BASE would hash the remaining 54 bytes into a four-byte numeric representation of the text.
- If you omit the SIZE option and the text field in the column is 200 bytes or less, R:BASE creates a full-text index. For example, if the text is 80 bytes and you do not specify a size, R:BASE builds a full-text index of 80 bytes.

MICRORIM_INDEXLOCK

The system variable, `MICRORIM_INDEXLOCK`, is available to control concurrency locks for the `CREATE INDEX` command.

This variable prevents `CREATE INDEX` from holding a permanent database lock. It locks only as necessary, allowing users access to the database. This results in longer index creation time but greater concurrency. `MICRORIM_INDEXLOCK` is set to any integer value.

Examples

The following command creates an index for the *custid* column in the *transmaster* table.

```
CREATE INDEX trancust ON transmaster (custid)
```

The following example creates a multi-column index for the *company*, *custaddress*, and *custstate* columns in the *customer* table.

```
CREATE INDEX custaddr ON customer (company ASC, custaddress ASC, custstate ASC)
```

1.4.21 CREATE SCHEMA

Use the `CREATE SCHEMA` command to name a database and assign a user identifier for the database owner.

```
CREATE SCHEMA AUTHORIZATION dbname [ownername]
```

Options

AUTHORIZATION dbname

Specifies the name of the database.

ownername

Allows you to assign a unique identifier for the owner of the database.

About the CREATE SCHEMA Command

In R:BASE eXtreme 9.0 (32), a database name is limited to 8 characters. In R:BASE eXtreme 9.0 (64), a database name is limited to 128 characters. The database name must begin with a letter, and can contain letters, numbers, and the following symbols: number or pound sign (#), dollar sign (\$), underscore (_), or percent sign (%). A database name cannot contain blanks or have a file extension, and must follow the naming conventions for R:BASE and the operating system.

R:BASE eXtreme 9.0 (32) automatically creates four database files with extensions: `.RB1`, `.RB2`, `.RB3`, and `.RB4`. R:BASE eXtreme 9.0 (64) automatically creates four database files with extensions: `.RX1`, `.RX2`, `.RX3`, and `.RX4`.

After you name a database, you need to use other commands to define the tables, views, rules, and access rights for the database.

Assigning A Database Owner's User Identifier

A database owner's user identifier can be a maximum of 36 characters and must begin with a letter and can contain letters, numbers, and the following symbols: number or pound sign (#), dollar sign (\$), underscore (_), or percent sign (%). If user identifiers are assigned to users, the database owner's user identifier must be unique among all user identifiers in the database.

If you do not specify a user identifier, R:BASE assigns the default user identifier, `PUBLIC`. Until a user identifier is assigned, anyone can modify the database structure, read, enter, change, or delete data. When an owner's user identifier is assigned to a database, the database is accessible only by the owner. To give other users access rights to the database, use the [GRANT](#) command.

You do not have to assign a user identifier when the database is created. To assign a user identifier after the database has been created, use the [RENAME OWNER](#) command, the **Utilities: Access Rights...**

menu option in R:BASE for Windows or the **Info: Create: Access Rights: Change Owner** in R:BASE for DOS.

CREATE SCHEMA stores the owner's case folding and collating tables from the RBENGINE9.CFG file in the database. Be sure to keep a record of the owner's user identifier in a safe place away from your computer. If you lose the owner's identifier, you cannot search the database to find it.

Building a Database

You can use CREATE SCHEMA and other [CREATE](#) commands as an alternative to creating a database using the **File: New Database** and **Tools: Database Designer** menu options in R:BASE for Windows or **Databases: Create/modify** in R:BASE for DOS.

When you run the CREATE SCHEMA command, R:BASE closes the currently open database (if one exists), then defines and opens a new database.

Transaction Processing and the CREATE SCHEMA Command

If transaction processing is on when you enter a CREATE SCHEMA command, R:BASE first commits your current transaction (if any), then creates and connects you to the database. Transaction processing is on in the database, but you cannot reverse the CREATE SCHEMA command.

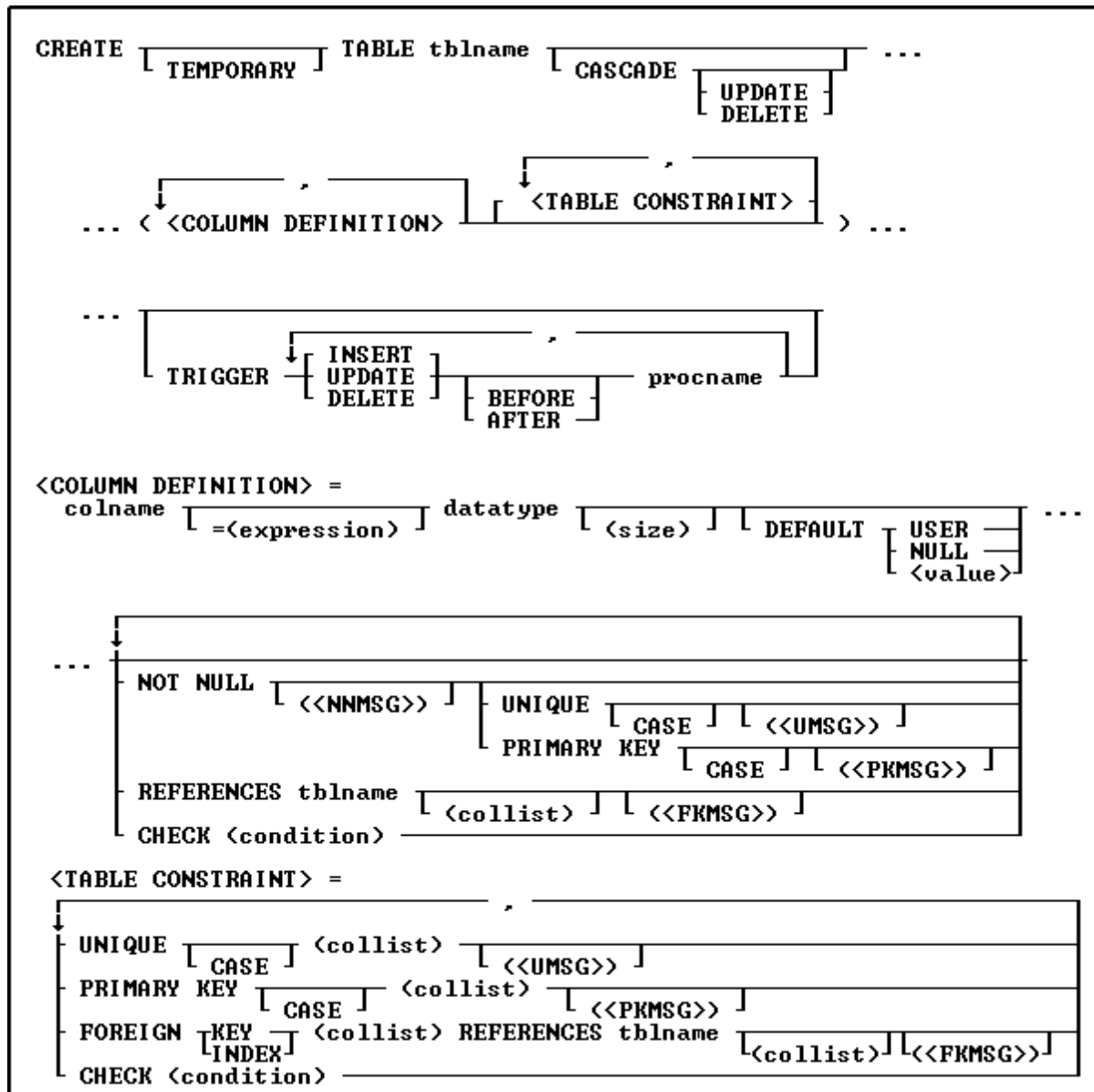
Example

The following command names the *finance* database and assigns *jane* as the database owner's user identifier.

```
CREATE SCHEMA AUTHORIZATION finance jane
```

1.4.22 CREATE TABLE

Use the CREATE TABLE command to define a new table in an existing database.



Options

,
Indicates that this part of the command is repeatable.

AFTER

Sets the trigger to activate after the INSERT, UPDATE or DELETE action.

BEFORE

Sets the trigger to activate before the INSERT, UPDATE or DELETE action. This is the default setting if the BEFORE/AFTER parameter is unused.

CASCADE

Maintains primary/foreign key relationships automatically. For example, if you either UPDATE or DELETE

a primary key value from a table, the corresponding foreign key values are updated or deleted automatically. A CASCADE can applied to UPDATE, DELETE or BOTH to specific primary keys. By not specifying either UPDATE or DELETE, both CASCADE restrictions will be enforced upon the primary/foreign key tables. Separate UPDATE and DELETE data restrictions can allow a CASCADE to be enforced for records that are updated, but not enforced when records are deleted, in order to avoid an accidental or undesired record delete. CASCADE can only be added to tables with primary keys.

CASE

Specifies that the data values will be case sensitive.

CHECK (condition)

Sets a condition to be satisfied before an update or insertion of a row can occur, which creates an R:BASE rule.

(collist)

Specifies a list of one or more column names, separated by a comma (or the current delimiter), used in the unique key specification. This option is only used when referencing a unique key.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

datatype

Specifies an R:BASE data type.

DEFAULT

Specifies a default value for the column if no value is provided by the user.

= (expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

(<FKMSG>)

Creates a constraint violation message to appear whenever a foreign-key data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a foreign-key constraint violation. You can define two messages: one for inserting and one for updating. A constraint must be dropped, then recreated in order to modify the violation message.

FOREIGN INDEX

With the FASTFK setting on, creates a foreign key that has an index using row pointers for data retrieval on selected columns.

FOREIGN KEY

Specifies a column or set of columns required to match values in a particular primary key or unique key defined in a table.

(<NNMSG>)

Creates a constraint violation message to appear whenever a not-null data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a not-null constraint violation. A constraint must be dropped, then recreated in order to modify the violation message.

NOT NULL

Prevents a column from accepting null values, but permits it to accept duplicate values. If this option is specified without a setting for a default value, you cannot insert rows without specifying values for the given column.

(<PKMSG>)

Creates a constraint violation message to appear whenever a primary-key data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a primary-key constraint violation. You can define three messages: one for uniqueness, one for deleting, and one for updating. A constraint must be dropped, then recreated in order to modify the violation message.

PRIMARY KEY

Specifies the column(s) to designate as a primary key.

procname

The procedure name. If a procedure by this name already exists in the database, an error is generated.

REFERENCES tablename

Identifies the primary key or unique key table to which the foreign key refers.

(size)

Defines the length of a column of either the CHARACTER data type (if not the default 1) or TEXT data type (if not the default 8). Defines the precision and scale of a column of the DECIMAL or NUMERIC data type, if not the default of precision 9 and scale 0 (9,0). VARBIT, VARCHAR, and BIT either require or can have a size.

tblname

Specifies a table name. In R:BASE eXtreme 9.0 (32), the table name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the table name is limited to 128 characters.

TEMPORARY

Creates a temporary table that disappears when the database is disconnected.

TRIGGER

Runs a Stored Procedure when an UPDATE, DELETE, or INSERT command is run on the table.

(<UMSG>)

Creates a constraint violation message to appear whenever a unique-key data integrity violation occurs. The message can suit the meaning of your data, such as "You must enter a valid number" for a unique-key constraint violation. You can define three messages: one for uniqueness, one for deleting, and one for updating. A constraint must be dropped, then recreated in order to modify the violation message.

UNIQUE

Requires the values in a column to be unique by defining a unique key constraint.

USER**NULL****(value)**

Default USER: Specifies the default value to be the user identifier.

Default NULL: Specifies the default value to be null.

Default (*value*): Specifies the default to be the indicated value.

About the CREATE TABLE Command

To define a table, you need to specify column definitions. Table and column names must begin with an upper- or lowercase letter. Names can contain letters, numbers, and the following special characters: #, \$, _, and %. R:BASE verifies that a table or column name is unique by reading all characters.

When you define a table, you can also add table constraints.

To define more than one column in a command, use commas (or the current delimiter character) to separate the column definitions.

Computed Columns

A computed column is a column containing a value that R:BASE calculates from an expression defined for the column. In the expression, you can use other columns from the table, constant values, functions, and the system variables *#date*, *#time*, and *#pi*. Global variables are not allowed in an expression.

You must assign a data type that is compatible with the result of the computation. The columns used for calculating the computed column must precede the computed column in the table.

Indexing Columns

If you want to assign an index to a column, use the [CREATE INDEX](#) command.

Database Access Rights with CREATE TABLE

CREATE TABLE requires either the CREATE access right or the owner's user identifier when access rights have been assigned with the [GRANT](#) command.

Examples

The command below defines a table named *employee* with the following columns and data types: *empid* (INTEGER), *emptitle*(TEXT 30), *empfname* (TEXT 10), *emplname* (TEXT 16), *empaddress* (TEXT 30), *empcity* (TEXT 20), *empstate* (TEXT 2), *empzip* (TEXT 10), *empphone* (TEXT 12), *empext*(INTEGER), *hiredate* (DATE), and *entrydate* (DATE). In addition, the NOT NULL option specifies that the columns *empfname*, *emplname*, and *hiredate* must contain a value. The NOT NULL UNIQUE option specifies that the *empid* and *empext* columns must contain unique values.

```
CREATE TABLE employee (empid INTEGER NOT NULL UNIQUE, +
emptitle TEXT 30, empfname TEXT 10 NOT NULL, emplname TEXT +
16 NOT NULL, empaddress TEXT 30, empcity TEXT 20, empstate +
TEXT 2, empzip TEXT 10, empphone TEXT 12, empext INTEGER +
NOT NULL UNIQUE, hiredate DATE NOT NULL, entrydate DATE)
```

The following command creates a table using the column constraint CHECK on the *empid* column.

```
CREATE TABLE employee (empid INTEGER CHECK (empid > 0), +
empname TEXT 40, empage INTEGER)
```

The example below creates a table using the column constraint CHECK on the *empid* and *empage* columns.

```
CREATE TABLE employee (empid INTEGER CHECK (empid > 0), +
empname TEXT 40, empage INTEGER CHECK (empage > 0 and empage < 100))
```

Table constraints are defined if it is necessary to reference multiple columns within the same expression. The UNIQUE (*collist*) option is entered at the end of the following command so that the values in the *empid*, *empfname*, *emplname* columns are unique as a group in a row. Because this option follows a column definition, precede the option with a comma.

```
CREATE TABLE employee (empid INTEGER NOT NULL, +
emptitle TEXT 30, empfname TEXT 10 NOT NULL, emplname TEXT +
16 NOT NULL, empaddress TEXT 30, empcity TEXT 20, empstate +
TEXT 2, empzip TEXT 10, empphone TEXT 12, empext INTEGER +
NOT NULL UNIQUE, hiredate DATE NOT NULL, entrydate DATE, +
UNIQUE (empid, empfname, emplname))
```

The following command creates a table that would contain an employee's total years of employment. The command places a column constraint on the *empid* and *empage* columns, and a table constraint on the *yrshere* and *yranywhere* columns. The value entered for *yrshere* must be less than or equal to the value entered for *yranywhere*.

```
CREATE TABLE employee (empid INTEGER CHECK (empid > 0), +
empname TEXT 40, empage INTEGER CHECK +
(empage > 0 and empage < 100), yrshere INTEGER, +
yranywhere INTEGER, CHECK (yrshere <= yranywhere))
```

1.4.23 CREATE VIEW

Use the CREATE VIEW command to define a view that combines columns from existing tables or views.

```
CREATE [ TEMPORARY ] VIEW viewname [ <collist> ] ...
      ... AS SELECT clause [ WITH CHECK OPTION ]
```

Options

AS SELECT clause

Specifies the columns and rows to include in the view. As a rule anything that is acceptable in an ordinary select clause will work here. If you are planning on using expressions you should considering using the [SELECT AS](#) notation to give each column meaningful names.

(collist)

Specifies a list of one or more column names or aliases, separated by a comma (or the current delimiter). These names will be the column headings displayed in the result of a [SELECT](#) command or the Data Browser.

TEMPORARY

Creates a temporary view that disappears when the database is disconnected.

viewname

Specifies a view name.

WITH CHECK OPTION

Specifies that a row cannot be added or updated unless it meets the conditions included in the [WHERE](#) clause (which is part of the SELECT clause). R:BASE uses this option only on a view that can be updated.

About the CREATE VIEW Command

CREATE VIEW defines a view to store in the sys_views table. You can use a stored view whenever necessary. Unlike a table, stored views contain no data. R:BASE collects data for the view from the source tables or views when a command calling the view is run.

A view is the most efficient way to gather data from separate tables or views into one location. A view that can be updated allows you to enter, change, and delete data from the source table. The number of tables in a view is dependent on available memory.

You can define a view containing a maximum of 400 columns from as many tables or views as memory allows. However, a view is still limited to the character row-size limit, 32,786 in R:BASE eXtreme 9.0 (64) and 4,096 in R:BASE eXtreme 9.0 (32). You can use the SQL symbol asterisk (*) to include all columns from all tables or views, or you can specify the columns you want to include. You can combine these two methods to include all columns from one table or view and specify columns from another table or view. You must separate column, table, and view names with commas (or the current delimiter character).

Avoiding Multiple Occurrences of Columns

When you use only an asterisk (*) in the [SELECT](#) clause, the view will contain all columns from all tables or views. If the tables or views contain common columns, the view will contain multiple occurrences of those columns.

To avoid multiple occurrences of common columns, specify which columns to include in the view. For example, to include all columns from one table but only certain columns from another table, use an * for the first table, then list the column names to be included from the second table. You can specify the columns for a view as *t1.**, *t2.col2*, *t2.col3*, where *t1.** specifies all columns from table *t1* and *t2.col2*, *t2.col3* specifies two columns from table *t2*. Be sure that the list does not include the common columns contained in the second table. When you use a combination of * and column names, you must specify the table with which * is associated. However, you can omit the table or correlation name for the

columns listed individually if those columns occur in only one table in the view.

Duplicate Rows

If the tables forming a view contain duplicate rows, either individually or in combination with other tables in the view, multiple duplicate rows will be displayed. Usually, the presence of duplicate rows in a view indicates a database design problem. Check your database structure for design flaws such as redundant data storage.

Linking Columns

When you build a view from two or more tables or views, define the relationship between the source tables and views by identifying linking columns in a WHERE clause. Linking columns are columns that contain the same values; their names can be the same or different. For example, the following WHERE clause specifies that a view displays only those rows where the values in *t1.col1* are equal to the values in the common column *t2.col1*.

```
WHERE t1.col1 = t2.col1
```

UNION Operator

Use views to create a subset of information. The conditions that link the source tables or views in the WHERE clause will limit the rows used in the view. However, you can create a view that uses all the rows from two or more tables by using the [UNION](#) operator. The UNION operator allows you to specify different selection criteria for each table in the view, as shown in the **Examples** below.

Updating Views

You can update the data for columns in a view when the view does not contain a UNION operator, and its [SELECT](#) clause meets the following requirements:

- The clause does not specify [DISTINCT](#).
- The clause does not include a [sub-SELECT](#) command in the WHERE clause.
- The clause does not include a [GROUP BY](#) or [HAVING](#) clause.
- The clause does not include an [ORDER BY](#) clause.

When you add, change, or delete rows by updating a single table view, you also modify the data in the source table. In multi-table views you cannot add, edit, or delete rows. Any additions or changes to data made through a view are subject to all the user-defined rules specified for the table when it was constructed. In addition, if you specify the WITH CHECK OPTION for the view, you can only add or modify rows that meet the conditions defined in the WHERE clause.

You can only use the [DELETE](#), [EDIT](#), [INSERT](#), [LOAD](#), and [UPDATE](#) commands with views that can be updated. If a view cannot be updated, you can use the view only to display data or as the basis for reports.

Other Commands to be Used with CREATE VIEW

Use the following commands with any view created with CREATE VIEW.

Available Commands for Views Created with CREATE VIEW			
BROWSE	EDIT USING *	PRINT	REVOKE
COMPUTE	FORMS	PROJECT	SELECT
CREATE VIEW	GRANT	QUERY	TALLY
CROSSTAB	LBLPRINT	RBLABELS	UPDATE **
DROP	LIST	RENAME	
EDIT *	LIST ACCESS	REPORTS	

* These commands will display data when using a multi-table View (a View based on multiple tables), but the data will not be editable. A View with a GROUP BY parameter is also not editable.

** The UPDATE command will not update data when using a multi-table View (a View based on

multiple tables), as the data is not editable. A View with a GROUP BY parameter is also not editable.

Views Compared with Look-up Tables

If the data used in a report is stored in more than one table, using a view is more efficient than a driving table and several look-up tables because it takes less time to print the report using a view. Using a view is more efficient because R:BASE gathers the data for a view before, rather than during, printing.

Database Access Rights with CREATE VIEW

The access rights that can be assigned with the [GRANT](#) command depend on whether or not the view can be updated. The ALL PRIVILEGES, DELETE, INSERT, SELECT, and UPDATE access rights can be granted on a view that can be updated. You must have ALL PRIVILEGES or SELECT access rights on a table or view to include it in a view.

If you have been assigned the SELECT access right and the WITH GRANT OPTION has been assigned on all the source tables or views used in a view, you can grant both SELECT and the WITH GRANT OPTION to other users.

If you are the database owner or you have the WITH GRANT OPTION on a view, you can also assign access rights on stored views.

Changing Views

You cannot change a view at the R> Prompt. To change a view from the R> Prompt, you must delete the view by using the [DROP](#) command, then define a new view. However, you can use the **View Designer** option from the **Tools** menu in R:BASE for Windows or **Views: Create/modify: Manage views** in R:BASE for DOS, to change a view that meets the following requirements:

- The view does not include a [UNION](#) operator.
- The view does not include a [GROUP BY](#) or [HAVING](#) clause.
- The view does not include a [sub-SELECT](#) command in the [WHERE](#) clause.

Examples

The following command defines a view that can be updated and specifies a subset of columns "custid, company, custaddress, custcity, custstate, and custzip" from one table, *customer*. The column list must match the number of columns in the SELECT clause; the names, however, can be different. The WHERE clause restricts the rows to those with zip codes ranging from 40001 through 49999. The WITH CHECK OPTION specifies that only rows that meet the condition included in the WHERE clause can be added or changed in the database.

```
CREATE VIEW cust_addr (custid, custcompany, custaddress, custcity, custstate, +
custzip) AS SELECT custid, company, custaddress, +
custcity, custstate, custzip FROM customer WHERE custzip +
BETWEEN 40001 AND 49999 WITH CHECK OPTION
```

The following command defines a view that will display only those rows from the *customer* and *transmaster* tables that have matching values in the common column *custid*. Therefore, only the rows that have customers who have had a transaction will be included in the view. The command line ORDER BY *custid* tells R:BASE to sort the rows and display them by the customer identification number.

```
CREATE VIEW cust_trans AS SELECT t1.custid, company, netamount +
FROM customer t1, transmaster t2 WHERE t1.custid = t2.custid +
ORDER BY custid
```

The following command defines a view that will display only those rows from the *customer* table where the values in the *custid* column do not exist in the *transmaster* table. Therefore, only the rows that have customers who have not had a transaction will be included in the view.

```
CREATE VIEW cust_notrans AS SELECT custid, company FROM +
customer WHERE custid NOT IN (SELECT custid FROM transmaster) +
```

ORDER BY custid

The following command combines the commands in the two preceding examples, creating a view that will display all rows from both the *customer* and the *transmaster* tables. The UNION operator joins the two SELECT clauses, allowing you to display rows for all customers whether or not they have had a transaction.

The first SELECT clause instructs R:BASE to include the rows from both tables where the values in *custid* match. The second SELECT clause instructs R:BASE to include rows from the *customer* table where there are no values for *custid* in the *transmaster* table.

When you use the UNION operator, the number of columns specified in the SELECT clauses must be the same and the data types of the columns must be compatible. If there is no column in one table that matches a column listed in the other table's SELECT clause, you must substitute a value (or null value). Because the *netamount* column does not exist in the *customer* table used in the second SELECT statement, the value \$0.00 was entered in place of *netamount*.

```
CREATE VIEW all_cust_trans AS SELECT t1.custid, company, netamount +
FROM customer t1, transmaster t2 WHERE t1.custid = t2.custid +
UNION SELECT custid, company, $0.00 FROM customer +
WHERE custid NOT IN (SELECT custid FROM transmaster) +
ORDER BY custid
```

1.4.24 CREATEOBJECT

Use the CREATEOBJECT command to create dynamic form objects on demand.

```
CREATEOBJECT <ControlType> <ComponentID> <ParentControl> <Left Top Width Height>
```

Options

ComponentID

The unique identifier that you assign to the component when you insert the object.

Height

The Height coordinate for the control

Left

The Left coordinate for the control in relation to the left side of its parent

ControlType

The form control, or object type. The actual name of the Control Type that must be used in the CREATEOBJECT syntax are listed as follows by Form Control category:

Standard Controls

- StaticText
- Button
- EnhancedPanel
- EnhancedGroupBox
- Wallpaper
- EnhancedWallpaper

Additional Controls

- BitBtn
- SpeedButton
- Image
- Shape
- Bevel
- Animate

Splitter
TabControl
TabSheet
LedTextLabel
Clock
TreeView
ListView
TimeLedLabel
OfficeButton
DropDownButton
GIFImage
LedDisplay
ShapeButton
Meter
EnhancedSpeedButton
PDFViewer
TabSet
GroupBar
Group
EnhancedTabControl
EnhancedTabSheet
Separator
Line
StatusBar
 ClockStatus
 StatusPane
 KeyStatus
 GlyphStatus
 MarqueeStatus
 DBStatus
 DBStateStatus
PDFViewer
TabSet
LinkLabel

Variable Controls

VarMemo
VarEdit
VarDateTimePicker
VarCalendar
VarCheckBox
VarLabel
VarRadioButton
VarRadioButtonGroup
VarUserComboBox
VarUserListBox
VarLookUpListBox
VarLookUpComboBox
VarSpinEdit
VarEmailLabel
VarWebBrowser
VarImage
VarRichEdit
VarLookUpListView
SystemVariable

Database Controls

DBGrid
DBNavigator
DBText
DBEdit
DBDateTimePicker
DBCalendar
DBMemo
DBImage

DBCheckBox
DBRadioGroup
DBLookupListBox
DBLookupComboBox
DBRichEdit
DBSpinEdit
DBUserComboBox
DBUserListBox
DBRadioButton
DBTreeView
DBWebBrowser
DBMailLabel
DBLookupListView
ScrollingRegion
EnhancedDBNavigator
EnhancedDBGrid

Legacy Controls

VerticalLine
HorizontalLine
3DBox
Panel
GroupBox

Internet Controls

WebBrowser
MailLabel

File System Controls

FileBox
DirTree

ParentControl

The form object in which other controls are contained within. A parent control can be an actual form or any of the following form controls: Panel, EnhancedPanel, GroupBox, EnhancedGroupBox, Tab Control, Enhanced Tab Control, Status Bar, RadioGroup, DB RadioGroup, Variable RadioGroup, Scrolling Region, etc.

Top

The Top coordinate for the control in relation to the top of its parent

Width

The Width coordinate for the control

About the CREATEOBJECT Command

Any form object can be created at runtime, simply by knowing the exact Control Type, Component ID, parent (form or any control having the option of container such as Panel, GroupBox, Tab Control, Enhanced Tab Control, Status Bar, DB RadioGroup, Scrolling Region, Variable RadioGroup), and coordinates (Left, Top, Width, Height).

To place DB Objects, you'll need to provide additional parameters, such as, Table Name, Field Name. For Variable Objects, you'll need to provide associated variable name. And, once the object is defined, you need to initialize it to be place on the form. All of these additional definitions for the newly created object are performed using the PROPERTY Command.

Use the Data Dictionary to browse the list of Form Control Type and to easily insert the values into your command files/applications.

Order of Processing To Create an Object

The order of processing for the CREATEOBJECT command is as follows

1. Use CREATEOBJECT to create the object
2. For Column(DB) or Variable fields, use the [PROPERTY](#) Command to assign the Column(DB) or Variable field
3. When inserting Custom EEPs, use the PROPERTY Command to assign the Custom EEP command syntax
4. Use the **PROPERTY ComponentID INIT 'TRUE'** Command to initialize the object
5. Use the PROPERTY Command to set any additional characteristics, like color, font, background, etc., that would be specific to the type of object you are creating

List of Controls and EEPs:

Standard Controls						
Button	OnClickEEP					
Database Controls						
DB Grid	OnEntryEEP	OnExitEEP	OnColumnEntryEEP	OnColumnExitEEP	OnColumnMoveEEP	OnCellClickEEP
DB Edit	OnEntryEEP	OnExitEEP	OnRightClickEEP			
DB Memo	OnEntryEEP	OnExitEEP	OnRightClickEEP			
DB Rich Edit	OnEntryEEP	OnExitEEP	OnRightClickEEP			
DB Date Time Picker	OnClickEEP					
DB Calendar	OnClickEEP					
DB Image	OnClickEEP	OnRightClickEEP	OnMouseEnterEEP	OnMouseMoveEEP	OnMouseLeaveEEP	
DB Check Box	OnClickEEP					
DB Radio Button	OnClickEEP					
DB Radio Group	OnEntryEEP	OnExitEEP	OnClickEEP			
DB Lookup Combo Box	OnEntryEEP	OnExitEEP	OnClickEEP			
DB Lookup List Box	OnEntryEEP	OnExitEEP	OnClickEEP			
DB User Defined Combo Box	OnEntryEEP	OnExitEEP	OnClickEEP			
DB User Defined List Box	OnEntryEEP	OnExitEEP	OnClickEEP			
DB Tree View	OnEntryEEP	OnExitEEP	OnClickEEP			
DB Lookup List View	OnEntryEEP	OnExitEEP	OnClickEEP	OnDbClickEEP		
Variable Controls						
Variable Edit	OnEntryEEP	OnExitEEP	OnRightClickEEP			
Variable Memo	OnEntryEEP	OnExitEEP	OnRightClickEEP			
Variable Rich Edit	OnEntryEEP	OnExitEEP	OnRightClickEEP			
Variable Date Time Picker	OnClickEEP					
Variable Calendar	OnClickEEP					
Variable Image	OnClickEEP	OnRightClickEEP	OnMouseEnterEEP	OnMouseMoveEEP	OnMouseLeaveEEP	

Variable Check Box	OnClickEEP					
Variable Radio Button	OnClickEEP					
Variable Radio Group	OnEntryEEP	OnExitEEP	OnClickEEP			
Variable Lookup Combo Box	OnEntryEEP	OnExitEEP	OnClickEEP			

Notes:

- A form must include the associated Master table
- Any additional form related tables **MUST** be defined using the form designer
- To delete an object on the fly, like **CREATEOBJECT**, refer to the [DELETEOBJECT](#) command

Examples

```
-- Place a text caption for the field (ObjectType: StaticText)
CREATEOBJECT StaticText 'StaticText1' 'RBASE_FORM' 10 10 400 20
PROPERTY StaticText1 INIT TRUE
PROPERTY StaticText1 CAPTION 'Dynamically Created DB Controls'

-- Place a Panel on the Form (ObjectType: Panel)
CREATEOBJECT Panel 'Panel' 'RBASE_FORM' 10 40 400 120
PROPERTY Panel INIT TRUE

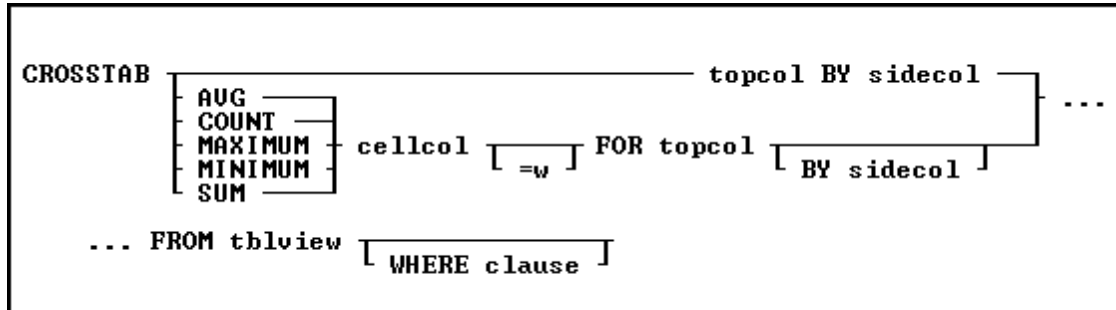
-- DB Edit Control Based on Table: Customer, Column: CustAddress
CREATEOBJECT DBEdit 'DBEdit2' 'Panel' 100 35 220 20
PROPERTY DBEdit2 TblName 'Customer'
PROPERTY DBEdit2 FldName 'CustAddress'
PROPERTY DBEdit2 INIT TRUE
PROPERTY DBEdit2 FRAMEVISIBLE TRUE

-- Place a DB Navigator Control Based on Table: Customer
CREATEOBJECT DBNavigator 'Navigator' 'Panel' 100 90 200 20
PROPERTY Navigator TblName 'Customer'
PROPERTY Navigator INIT TRUE
PROPERTY Navigator IMAGESTYLE 'Razzmatazz'
PROPERTY Navigator FLAT TRUE

-- To insert Custom EEP command syntax into the "DBEdit2" Control defined
above
-- PROPERTY ComponentID 'Custom_EEP->YourCustomEEP' 'YourCustomEEPText'
CLEAR VAR vCommand1
SET VAR vQuote = (CVAL('QUOTES'))
SET VAR vCommand1 TEXT = ('PAUSE 2 USING'&vQuote&'This is the Address'
&vQuote&';RETURN')
PROPERTY DBEdit2 'Custom_EEP->OnClickEEP' .vCommand1
RETURN
```

1.4.25 CROSSTAB

Use the CROSSTAB command to count the occurrences of each unique pair of values from two columns in a table or view.



Options

AVG

Computes the numeric average of CURRENCY, DOUBLE, NUMERIC, REAL, INTEGER, DATE, or TIME data types. R:BASE rounds averages of integer values to the nearest integer value and currency values to their nearest unit.

BY sidecol

Specifies the column whose values are displayed down the side of the crosstab.

cellcol

Specifies the column whose values are computed and displayed in the cells portion of the crosstab.

COUNT

Determines how many non-null entries there are for a particular column item.

FOR topcol

Specifies the column whose values are displayed across the top of the crosstab. If the *topcol* option is omitted, the *sidecol* and *cellcol* total or computation is displayed.

FROM tblview

Specifies the table or view.

MAXIMUM

Selects the maximum value in a column. For TEXT and CHARACTER data types, R:BASE evaluates the first eight characters. This option accepts columns with any data type except BIT, LONG VARBIT, LONG VARCHAR, VARBIT, VARCHAR, and BITNOTE.

MINIMUM

Selects the minimum value in a column. For TEXT and CHARACTER data types, R:BASE evaluates the first eight characters. This option accepts columns with any data type except BIT, LONG VARBIT, LONG VARCHAR, VARBIT, VARCHAR, and BITNOTE.

SUM

Computes the numeric sum of CURRENCY, DOUBLE, REAL, NUMERIC, or INTEGER data types.

topcol

Specifies the column whose values are displayed across the top of the crosstab.

=w

Sets the width, in number of characters, of the column displaying the results for *cellcol*.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the CROSSTAB Command

You can use CROSSTAB from the R> Prompt from the **Calculate** menu in the "Data Browser" window in R:BASE for Windows or **Calculate** from the **Info/views** menu in R:BASE for DOS. R:BASE sorts the information in ascending order and displays it in tabular form. You can also display the results of numeric calculations performed on the values in the two columns in a third column.

You can now do CROSSTAB commands on NOTE fields. It will only use the first 20 characters, but this permits text expressions from views to be used.

The CROSSTAB command does not consider null values in its calculations.

CROSSTAB Command Arguments	
Argument	Description
AVG	Computes the numeric average of CURRENCY, DOUBLE, NUMERIC, REAL, INTEGER, DATE, or TIME data types. R:BASE rounds averages of integer values to the nearest integer value and currency values to their nearest unit.
COUNT	Determines how many entries there are for a particular column item.
MAXIMUM	Selects the value in a column with the greatest numeric, time, date, or alphabetic value. For a TEXT data type, R:BASE evaluates the first eight characters.
MINIMUM	Selects the value in a column with the least numeric, time, date, or alphabetic value. For a TEXT data type, R:BASE evaluates the first eight characters.
SUM	Computes the numeric sum of CURRENCY, DOUBLE, REAL, NUMERIC, INTEGER, or data types.

Example

The following command shows the net sum of the transactions for the sales representative whose employee identification is 102, ordered by transaction date.

```
CROSSTAB SUM netamount FOR empid BY transdate FROM transmaster WHERE empid = 102
```

The results look similar to the following:

```
transdat | 102 (Total)
----- | -----
01/11/94 | $176,000.00 $176,000.00
02/26/94 | $87,500.00 $87,500.00
02/27/94 | $22,500.00 $22,500.00
02/28/94 | $40,500.00 $40,500.00
03/07/94 | $57,500.00 $57,500.00
09/26/94 | $3,060.00 $3,060.00
----- | -----
          | $387,060.00 $387,060.00
```

1.5 D

1.5.1 DEBUG

Precede any R:BASE command with the DEBUG command to control the running of the command. In forms runtime, [Ctrl] + [Shift] + [F11] will toggle the DEBUG flag.

DEBUG command

Option

command

Specifies a valid command name.

About the DEBUG Command

If [DEBUG](#) is set on, the DEBUG command runs; if it is set off, the DEBUG command is ignored. By allowing DEBUG to be set on or off, DEBUG statements in command files or applications can be activated or deactivated.

Examples

The following command lines show all variables and issue a [LIST](#) command.

```
SET DEBUG ON
DEBUG SHOW VAR
LIST
SET DEBUG OFF
```

Because DEBUG is set off, the following command sequence only issues the LIST command.

```
SET DEBUG OFF
DEBUG SHOW VAR
LIST
```

1.5.2 DECLARE CURSOR

Use the DECLARE CURSOR command to create a cursor that points to a row in a table or view.

DECLARE cursor [SCROLL] CURSOR FOR SELECT clause

Options

cursor

Specifies a 1 to 18 character cursor name.

CURSOR FOR SELECT clause

Specifies the columns and rows from the table whose values you want to use. You may include the [DISTINCT](#) modifier as well as [WHERE](#) clauses and [ORDER BY](#) clauses.

SCROLL

Defines a cursor that moves forwards and backwards through a table. If this option is omitted, the cursor can only move forward.

About the DECLARE CURSOR Command

In the [SELECT](#) clause, specify the columns that contain the values you want to use from the row. Specifying the columns makes the column values accessible to the [FETCH](#) and [SET VARIABLE](#) commands. Once a cursor is declared, use the [OPEN](#) command to initialize the cursor and position it before the first row specified by DECLARE CURSOR.

Use DECLARE CURSOR to define a path through a table or view. You can move through the defined rows using the FETCH command by using either multiple FETCH commands or embedded FETCH commands within a WHILE loop. You only need to point to specific columns with DECLARE CURSOR, then FETCH can retrieve those columns by placing their values into variables. You can define a scrollable cursor, which is a cursor that moves backwards and forwards through a table.

DECLARE CURSOR defines a temporary view in memory; R:BASE does not store the view definition in the sys_views table. The SELECT clause defines columns, tables, rows, sort order, and potential grouping for the rows. When DECLARE CURSOR executes, it validates the syntax and names of columns and tables. The [OPEN](#) command can evaluate variables, create a copy of the cursor based on those values, then position the cursor before the first row.

Listing Cursors

Use [LIST CURSOR](#) to list all currently defined cursors and their status, open or closed.

Using Cursor Names in Commands

You can use the cursor name instead of a table name in commands. The following table provides examples of using the cursor name instead of a table name in commands.

To do this...	Use the cursor name like this...
Set a variable to a column value	FETCH <i>cursor</i> INTO <i>varlist</i> SET VARIABLE <i>varname</i> = <i>colname</i> WHERE CURRENT OF <i>cursor</i>
Change a column value to a constant	UPDATE <i>tblname</i> SET <i>colname</i> + = <i>value</i> WHERE CURRENT OF <i>cursor</i>
Change a column value to a variable value	UPDATE <i>tblname</i> SET <i>colname</i> + = <i>.varname</i> WHERE CURRENT OF <i>cursor</i>
Change a column value to an expression	UPDATE <i>tblname</i> SET <i>colname</i> + = (<i>expression</i>) WHERE CURRENT OF <i>cursor</i>
Delete the pointed-to row	DELETE FROM <i>tblname</i> + WHERE CURRENT OF <i>cursor</i>

Modifying Data Using a Cursor

If you use a cursor in commands that modify data (the [UPDATE](#) and [DELETE](#) commands), only the current row is modified. To modify all referenced rows, include [FETCH](#) in a [WHILE](#) loop to move the cursor through the rows.

Checking for End-of-Data Conditions

End-of-data conditions determine whether you have reached the end of the data declared with the DECLARE CURSOR command. The three ways to check for end-of-data conditions are:

- Use an error variable defined with the [SET ERROR VARIABLE](#) command
- Use the *sqlcode* variable
- Include a [WHENEVER NOT FOUND](#) command

Closing Cursors

The following commands close cursors.

Command Name	Description
CLOSE	Closes the open cursor but does not remove the cursor definition. However using CLOSE frees most of the memory used when a cursor is opened. CLOSE also frees any file handles used by DECLARE CURSOR.
COMMIT	Closes any open cursors
CONNECT	Removes any cursor definitions from memory
DISCONNECT	Removes any cursor definitions from memory
DROP CURSOR	Entirely removes the cursor definition. Dropping a cursor definition frees all memory used by the definition.
ROLLBACK	Closes any open cursors

Examples

The following example uses the SCROLL option with DECLARE CURSOR.

```
DECLARE c1 SCROLL CURSOR FOR SELECT empid, transid, transdate, custid, netamount FROM transmaster
```

Checking End-of-Data Conditions Using sqlcode

The following example shows the same type of processing as shown in the first example but uses *sqlcode* instead of an error variable to check end-of-data conditions. Using *sqlcode* is the recommended program structure for DECLARE CURSOR.

The two types of error processing differ. An error variable defined by [SET ERROR VARIABLE](#) holds the

error value for any type of error; however, the *sqlcode* system variable holds values only for specific types of status.

Type of Error	SQLCODE
Data found	0
Data not found	100

In the following example, the [WHILE](#) statement checks the value of *sqlcode*.

- 1) DECLARE cursor1 CURSOR FOR SELECT custid, netamount +
FROM transmaster ORDER BY netamount
- 2) OPEN cursor1
- 3) FETCH cursor1 INTO vcustid INDICATOR vi1, vnetamt +
INDICATOR vi2
- 4) WHILE sqlcode <>100 THEN
SHOW VARIABLE vcustid
SHOW VARIABLE vnetamt
- 5) FETCH cursor1 INTO vcustid INDICATOR vi1, vnetamt +
INDICATOR vi2
ENDWHILE
- 6) DROP CURSOR cursor1

1. DECLARE CURSOR defines the cursor path.
2. [OPEN](#) opens the cursor, evaluates variables, and positions the cursor before the first row.
3. The first [FETCH](#) command retrieves the first set of values. The indicator variables *vi1* and *vi2* capture the status values, -1 for null and 0 for a value. If you omit indicator variables in [FETCH](#) commands, R:BASE displays a message if it encounters a null value, but continues processing rows.
4. The [WHILE](#) loop processes the rows until there are no more rows. At that point, *sqlcode* is set to 100, and the WHILE loop ends. Control passes to the command after ENDWHILE. If the first [FETCH](#) retrieved no data, the WHILE loop is not entered.
5. [FETCH](#) retrieves all succeeding rows and sets *sqlcode* each time. When it does not find any more data, *sqlcode* is set to 100 and the WHILE loop ends.
6. [DROP CURSOR](#) removes the cursor definition from memory.

Using the WHENEVER Command with DECLARE CURSOR

The following example shows the use of the [WHENEVER](#) command, which checks the value of *sqlcode*. A single WHENEVER command can start a status-checking cycle that remains in operation until a command or procedure file finishes running. As in the first two examples, an indicator variable is included with each variable in [FETCH](#). Without the indicator variables, R:BASE displays a message if it encounters a null value, but continues processing rows.

- 1) WHENEVER NOT FOUND GOTO skiploop
- 2) DECLARE cursor1 CURSOR FOR SELECT custid, netamount +
FROM transmaster ORDER BY netamount
- 3) OPEN cursor1
- 4) FETCH cursor1 INTO vcustid INDICATOR vi1, vnetamt +
INDICATOR vi2
- 5) WHILE #DATE IS NOT NULL THEN
SHOW VARIABLE vcustid
SHOW VARIABLE vnetamt
FETCH cursor1 INTO vcustid INDICATOR vi1, vnetamt +
INDICATOR vi2
ENDWHILE
- 6) LABEL skiploop
- 7) DROP CURSOR cursor1

1. WHENEVER NOT FOUND tells R:BASE to execute [GOTO](#) if a command that searches for data, such as [FETCH](#), cannot find more rows. If the first [FETCH](#) command does not find any rows, control passes to the command following LABEL *skiploop*. WHENEVER automatically checks any command that

searches for data. If a data-not-found condition occurs, control passes to the command following the specified label.

2. **DECLARE CURSOR** defines the cursor path.
3. **OPEN** opens the cursor, evaluates the variables, and positions the cursor before the first row.
4. The first **FETCH** command retrieves the first set of values. If no rows match, control passes to **LABEL skiploop**. Indicator variables *vi1* and *vi2* capture the status values (-1 for null and 0 for a value). If you omit indicator variables in **FETCH** commands, R:BASE displays a message if it encounters a null value, but continues processing rows. (**WHENEVER** instructs R:BASE to exit the **WHILE** loop only when *sqlcode* is 100.)
5. The **WHILE** loop processes rows until **WHENEVER** stops execution.
6. This label defines where to pass control if a data-not-found condition occurs before the **WHILE** loop begins executing. **WHENEVER** includes this label name.
7. **DROP CURSOR** removes the cursor definition from memory.

1.5.3 DEFF

Use the **DEFF** command to open the External Form File Designer, or to open the designer with aspecific file.

```
DEFF [ filename.rff ]
```

1.5.4 DELETE

Use the **DELETE** command to remove selected rows from a table.

```
DELETE [ ROWS ] FROM tblview [ WHERE clause  
WHERE CURRENT OF cursor ]
```

Options

FROM tblview

Specifies the table or view.

ROWS

This word is optional.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

WHERE CURRENT OF cursor

Specifies a cursor pointing to the row the [DELETE](#) command will remove. This option can replace a standard **WHERE** clause.

Use the [DECLARE CURSOR](#) command to define the cursor.

filespec

Specifies the file to be deleted.

On a workstation with multiple drives (local or mapped), especially when the files are on the different drive, it is always the best practice to define a drive letter when copying, deleting, renaming or running files, unless the specified files are located in the working directory. You will not need to specify the drive letter if all of the files are located in the default directory when using the copy, delete, rename or run commands.

About the **DELETE** Command

DELETE removes rows from a table or view. Without a **WHERE** or **WHERE CURRENT OF** clause, R:BASE deletes all rows from the specified table or view. R:BASE displays a confirmation message before deleting the rows. R:BASE does not display a confirmation message when you execute a **DELETE**

command from a command file. Views must be updatable to delete rows from it; for more information about updatable views, see [CREATE VIEW](#).

Before you use a WHERE clause with the DELETE command, test the clause by using it with a [SELECT](#) command, which allows you to view the rows before deleting them.

The WHERE CURRENT OF clause specifies a cursor pointing to a row that the DELETE command will remove. Once you define a cursor with DECLARE CURSOR and open a route with the OPEN command, you can use the cursor in a WHERE CURRENT OF clause to delete only the current row. Use the [FETCH](#) command to move the cursor to the next available row.

You must restore deleted rows from a backed up database or table. To recover disk space after deleting rows, use the [PACK](#) or [RELOAD](#) commands.

DELETE removes rows from a table or single-table view. If you have set transaction processing on, you can restore rows with [ROLLBACK](#). If not, you must restore them from a backup database or table. If you prefer not to use transaction processing, you can first use a relational command, such as [PROJECT](#), to make a backup copy of the table from which you are deleting rows. Then you can delete rows from the original table and remove the backup copy later.

Examples

The following command deletes all rows from the *transmastertable*. When you omit a [WHERE](#) clause, be sure that you want to delete all rows from the table.

```
DELETE FROM transmaster
```

The following command deletes rows from the *transmaster* table where the *custid* value is 100.

```
DELETE FROM transmaster WHERE custid = 100
```

1.5.5 DELETE DUPLICATES

Use the DELETE DUPLICATES command to remove duplicate rows from a table.

```
DELETE DUPLICATES FROM tblname [ USING collist ]
```

Options

FROM tblname

Specifies the table name.

USING collist

Deletes rows based on duplicate values in the specified list of columns.

About the DELETE DUPLICATES Command

Use DELETE DUPLICATES to delete duplicate rows from a table. A duplicate row is a row where the values for each column are exactly the same as those in another row in the table. This command deletes all but the first row for each set of duplicate rows.

DELETE DUPLICATES processes faster when the table contains an indexed column and the USING *collist* option is used.

Rules for Column Deletion

You can specify which rows to delete in a list of columns. The following rules apply:

- The first row is retained in the table.
- Any row with duplicate values in a specified column list is deleted, regardless of the values in

any of its other columns.

Case Sensitivity

DELETE DUPLICATES is case sensitive when [CASE](#) is set on. For example, if CASE is set on and one row included the name SMITH and another row included the name Smith, R:BASE would not delete either row. However, if CASE was set off, R:BASE would delete the second row. (The default setting for CASE is off.)

You must restore deleted rows from a backed up database or table. To recover the data's disk space after rows are deleted, use [PACK](#) or [RELOAD](#).

If you have set transaction processing on, you can restore rows with [ROLLBACK](#). If not, you must restore them from a backup database or table. If you prefer not to use transaction processing, you can first use a relational command, such as [PROJECT](#) to make a backup copy of the table from which you are deleting rows. Then you can delete rows from the original table and remove the backup copy later.

Example

The following command deletes duplicate rows from the *transmaster* table, but retains the first of the duplicate rows.

```
DELETE DUPLICATES FROM transmaster
```

The following example deletes duplicate rows based on the *transid*, *empid*, and *custid* columns in the *transmaster* table. Only the designated columns will be used to determine whether the rows are duplicates.

```
DELETE DUPLICATES FROM transmaster USING transid, empid, custid
```

1.5.6 DELETEOBJECT

Use the DELETEOBJECT command to delete form objects on demand.

```
DELETEOBJECT { <ComponentID> }
               { <ParentControl> }
```

Options

ComponentID

The unique identifier that is assigned to the component, or control .

ParentControl

The form object in which other controls are contained within. A parent control can be an actual form or any of the following form controls: Panel, EnhancedPanel, GroupBox, EnhancedGroupBox, Tab Control, Enhanced Tab Control, Status Bar, RadioGroup, DB RadioGroup, Variable RadioGroup, Scrolling Region, etc.

This command can be directly associated to the [CREATEOBJECT](#) command.

1.5.7 DELROW

Use the DELROW command to delete the current row of data currently being edited or entered in a form.

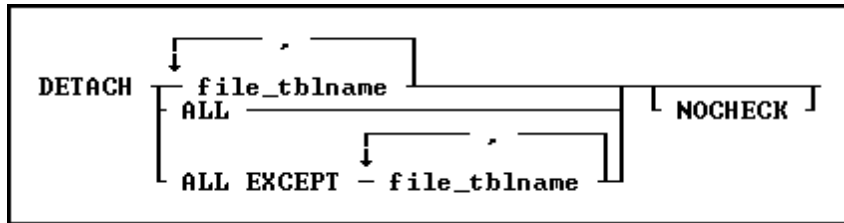
```
DELROW
```

About the DELROW Command

Use the DELROW command to delete the current row of data currently being edited or entered in a form.

1.5.8 DETACH

Use the DETACH command to remove a dBASE file table and its associated dBASE index files from the open R:BASE database.



Options

' Indicates that this part of the command is repeatable.

ALL

Removes all dBASE tables and associated dBASE index files from the open R:BASE database.

ALL EXCEPT file_tblname

Removes all dBASE tables and associated dBASE index files from the open R:BASE database, except the specified table.

file_tblname

Removes the specified dBASE table and associated dBASE index files from the open R:BASE database.

NOCHECK

Eliminates the confirmation message.

About the DETACH Command

Before you can detach a dBASE file table, an R:BASE database must be open. When you detach a dBASE file table, do not include the .DBF extension. DETACH requires the database owner's user identifier if one has been defined.

If you remove a dBASE file that is used in a form, report, label, or application, make the necessary changes to reflect detaching the dBASE file from the database. The rules, descriptions, and any access rights are deleted when you detach the dBASE file.

Use [ATTACH](#) to reattach a detached file to the same or a different database.

When [STATICDB](#) is set on-which activates a read-only schema mode-DETACH is unavailable.

Example

In the following example, the first command line opens the *concomp* database. The second command line detaches the dBASE file table *sampgate* from the *concomp* database.

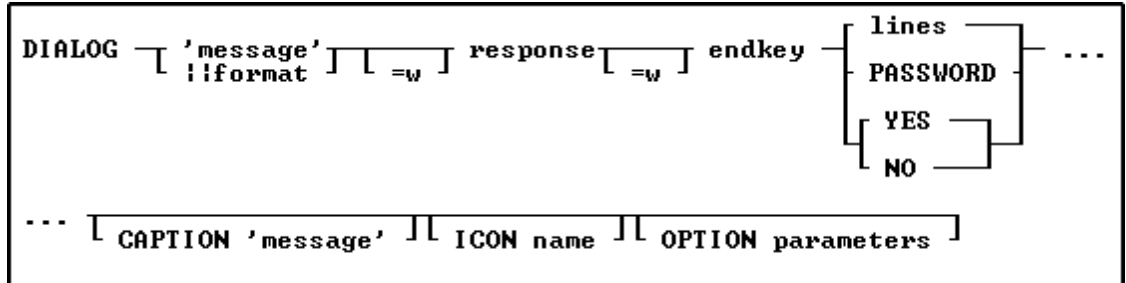
```

CONNECT concomp
DETACH sampgate

```


1.5.9 DIALOG

Use the DIALOG command to display a dialog box on the screen to accept text entry from a user. For assistance with building your DIALOG commands, refer to the [R:Dialog Builder Plugin](#), or the [R:Dialog with Buttons Builder Plugin](#).



Options

'message'

Specifies the text of the message to display in the dialog box. The value can also be passed as a variable.

||format

Also known as EditMask. You can apply the mask specified by the EditMask parameter to the text string specified by the Value parameter. See the "[||format](#)" subchapter for more.

=w

Specifies the wrap width for the dialog box message.

response

Specifies the variable that will contain the dialog box entry.

=w

Specifies the width for the dialog box entry.

endkey

Specifies the variable containing the final keystroke ([Enter] or [Esc]) in a dialog box.

lines

Specifies the number of lines to display for the text entry in a dialog box.

password

Displays the dialog box entry as asterisks.

YES

Creates a Yes/No dialog box and sets the default response to Yes.

NO

Creates a Yes/No dialog box and sets the default response to No.

CAPTION 'message'

Specifies the text of the message to display in the dialog box caption. The value can also be passed as a variable.

ICON *value*

Icon ' <i>value</i> ' Parameter	Icon
APPS	
ATTENTION	

CONFIRM	
ERROR	
HELP	
INFO	
QUESTION	
SERIOUS	
STOP	
WARNING	
WINDOWS	

Additional [OPTION parameters](#)

Additional parameters are available to increase the visual display of the DIALOG window. To use the graphic DIALOG Builder, choose "Utilities" > "Plugins" > "Internal Plugins" > "DIALOG Builder" from the main Menu Bar. All OPTION parameters and values must be separated by the "|" (pipe) character.

Examples

Example 01: (Dialog with custom button text)

```
CLS
DIALOG 'Enter Last Name' vLastName=26 vEndKey 1 +
CAPTION 'Search Employee by Last Name' +
ICON APP +
OPTION TITLE_FONT_COLOR BLACK +
|TITLE_BACK_COLOR WHITE +
|TRANSPARENCY 255 +
|WINDOW_BACK_COLOR WHITE +
|BUTTON_OK_CAPTION &Search +
|BUTTON_CANCEL_CAPTION &Cancel
```



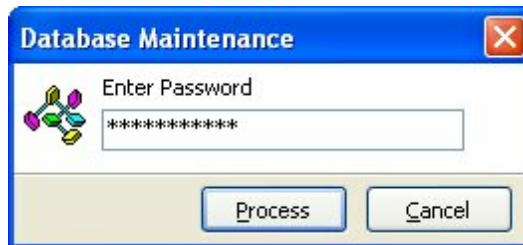
Example 02: (Dialog with PASSWORD option)

```
CLS
DIALOG 'Enter Password' vPassword=26 vEndKey PASSWORD +
CAPTION 'Database Maintenance' +
ICON APP +
```

```

OPTION TITLE_FONT_COLOR BLACK +
|TITLE_BACK_COLOR WHITE +
|TRANSPARENCY 255 +
|WINDOW_BACK_COLOR WHITE +
|BUTTON_OK_CAPTION &Process +
|BUTTON_CANCEL_CAPTION &Cancel

```



Example 03: (Dialog with Multi-Line Message with the TOP LEFT parameters)

```

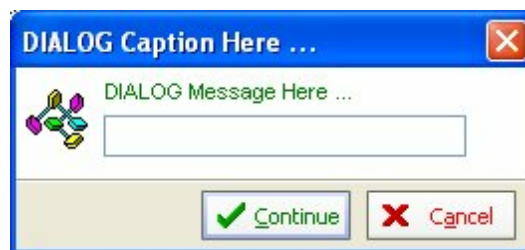
-- (CHAR(009)) = Tab Key (Indent)
-- (CHAR(013)) = Carriage Return
CLS
SET VAR vMsg = +
('Line 1: '+(CHAR(009))+(CHAR(009))&'Contents of Line 1'+(CHAR(013))++
'Line 2: '+(CHAR(009))+(CHAR(009))&'Contents of Line 2'+(CHAR(013))++
'Line 3: '+(CHAR(009))+(CHAR(009))&'Contents of Line 3'+(CHAR(013))++
'Line 4: '+(CHAR(009))+(CHAR(009))&'Contents of Line 4'+(CHAR(013))++
'Line 5: '+(CHAR(009))+(CHAR(009))&'Contents of Line 5'+(CHAR(013))++
'Line 6: '+(CHAR(009))+(CHAR(009))&'Contents of Line 6'+(CHAR(013))++
'Line 7: '+(CHAR(009))+(CHAR(009))&'Contents of Line 7'+(CHAR(013))++
'Line 8: '+(CHAR(009))+(CHAR(009))&'Contents of Line 8'+(CHAR(013))++
'Line 9: '+(CHAR(009))+(CHAR(009))&'Contents of Line 9'++(CHAR(013)))
DIALOG .vMsg vYesNo vEndKey YES +
CAPTION ' Your Dialog Caption Here ...' +
ICON APP +
OPTION TITLE_FONT_COLOR BLACK +
|TITLE_BACK_COLOR WHITE +
|TRANSPARENCY 255 +
|WINDOW_BACK_COLOR WHITE +
|BUTTON_YES_CAPTION &Start +
|BUTTON_NO_CAPTION &Cancel +
|BUTTON_YES_COLOR GREEN +
|BUTTON_NO_COLOR RED +
|BUTTON_YES_FONT_COLOR WHITE +
|BUTTON_NO_FONT_COLOR WHITE +
|TOP 50 +
|LEFT 50

```



Example 04: (Dialog with button images)

```
CLS
SET VAR vResponse TEXT = NULL
SET VAR vEndKey TEXT = NULL
DIALOG 'DIALOG Message Here ...' vResponse=26 vEndKey 1 +
CAPTION 'DIALOG Caption Here ...' ICON APPS +
OPTION TITLE_FONT_COLOR GREEN +
|TITLE_BACK_COLOR WHITE +
|TITLE_FONT_NAME ARIAL +
|TITLE_BOLD OFF +
|WINDOW_BACK_COLOR WHITE +
|BUTTON_OK_CAPTION &Continue +
|BUTTON_CANCEL_CAPTION C&ancel +
|BUTTON_YES_COLOR WHITE +
|BUTTON_NO_COLOR WHITE +
|BUTTON_YES_FONT_COLOR GREEN +
|BUTTON_NO_FONT_COLOR RED +
|BUTTONS_SHOW_GLYPH ON
```



Example 05: (No Caption Window)

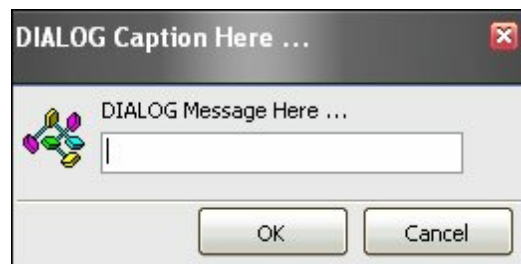
```
CLS
DIALOG 'DIALOG Message Here ...' vResponse=26 vEndKey 1 +
CAPTION 'DIALOG Caption Here ...' ICON APPS +
OPTION TITLE_FONT_COLOR GREEN +
|WINDOW_CAPTION OFF +
|TITLE_BACK_COLOR WHITE +
|TITLE_FONT_NAME ARIAL +
|TITLE_BOLD OFF +
|WINDOW_BACK_COLOR WHITE +
```

```
|BUTTON_OK_CAPTION &Continue +
|BUTTON_CANCEL_CAPTION C&ancel +
|BUTTONS_BACK_COLOR WHITE +
|BUTTON_FONT_COLOR GREEN
```



Example 06: (Dialog window using Themes)

```
DIALOG 'DIALOG Message Here ...' vResponse=26 vEndKey 1 +
CAPTION 'DIALOG Caption Here ...' ICON APPS +
OPTION themename longhorn
```



1.5.9.1 ||format

To use EditMask feature with the DIALOG command, you must start the text string with || (double pipe character). The edit mask string consists of three fields with semicolons separating them. The **first part** of the mask is the mask itself. The **second part** is the character that determines whether the literal characters of the mask are matched to characters in the Value parameter or are inserted into the Value string. The **third part** of the mask is the character used to represent missing characters in the mask.

These are the special characters used in the **first part** of the mask:

Character	Meaning in mask
!	If a ! character appears in the mask, optional characters are represented in the returned string as leading blanks. If a ! character is not present, optional characters are represented in the returned string as trailing blanks.
>	If a > character appears in the mask, all characters that follow are in uppercase until the end of the mask or until a < character is encountered.
<	If a < character appears in the mask, all characters that follow are in lowercase until the end of the mask or until a > character is encountered.
<>	If these two characters appear together in a mask, no case checking is done and the data is formatted with the case present in the Value parameter.
\	The character that follows a \ character is a literal character. Use this character to use any of the mask special characters as a literal.
L	The L character requires an alphabetic character only in this position. For the US, this is A_Z, a_z.
	The I character permits only an alphabetic character in this position, but doesn't require it.
A	The A character requires an alphanumeric character only in this position.

	For the US, this is A_Z, a_z, 0_9.
a	The a character permits an alphanumeric character in this position, but doesn't require it.
C	The C character requires an arbitrary character in this position.
c	The c character permits an arbitrary character in this position, but doesn't require it.
0	The 0 character requires a numeric character only in this position.
9	The 9 character permits a numeric character in this position, but doesn't require it.
#	The # character permits a numeric character or a plus or minus sign in this position, but doesn't require it.
:	The : character is used to separate hours, minutes, and seconds in times. If the character that separates hours, minutes, and seconds is different in the regional settings of the Control Panel, that character is substituted in the returned string.
/	The / character is used to separate months, days, and years in dates. If the character that separates months, days, and years is different in the regional settings of the Control Panel, that character is substituted in the returned string.
;	The ; character is used to separate the three fields of the mask.
_	The _ character automatically inserts spaces into the returned string.

Any character that does not appear in the preceding table can appear in the first part of the mask as a literal character. Literal characters are inserted automatically if the second field of the mask is 0, or matched to characters in the Value parameter if the second field is any other value. The special mask characters can also appear as literal characters if preceded by a backslash character (\).

The **second part** of the mask is a single character that indicates whether literal characters from the mask are included in the Value parameter. For example, the mask for a telephone number with area code could be the following string:

(000)_000_0000;0;*

The 0 in the second field indicates that the Value parameter should consist of the 10 digits of the phone number, rather than the 14 characters that make up the final formatted string.

A 0 in the second field indicates that literals are inserted into the Value string, any other character indicates that they should be included.

The **third part** of the mask is the character that appears in the returned string for blanks (characters that do not appear in Value). By default, this is the same as the character that stands for literal spaces. The two characters appear the same in the DIALOG. However, when a user edits the text in a masked edit control, the cursor selects each blank character in turn, and skips over the space character.

Note:

When working with multibyte character sets, such as Japanese Shift_JIS, each special mask character represents a single byte. To specify double_byte characters using the L,l,A,a,C, or c specifiers, the mask characters must be doubled as well. For example, LL would represent two single_byte alphabetic characters or a one double_byte character. Only single_byte literal characters are supported.

Typical Field Formats by Country:

United States	
Phone	4155551212 !\ (999)\ 000 0000;1;
Extension	15450 !99999;1;
Social Security No	555555555 000\ 00\ 0000;1;
Short Zip Code	90504 00000;1;
Long Zip Code	905040000 00000\ 9999;1;
Date	062794 !99/99/00;1;

Long Time	090515PM !90:00:00>LL;1;
Short Time	1345 !90:00;1;

United Kingdom	
Phone	1714561234 !\00009 0000099;1;
Extension	15450 !99999;1;
Postcode	WT3114TA >AAAa aaaa;1;
Short Postcode	EC12 >AAAa;1;
National Insurance	GM134152A !>AA000000A;1;
Date	270195 !90/90/00;1;
Long Time	090515PM !90:00:00>LL;1;
Short Time	1345 !90:00;1;

Taiwan	
Zip Code	107 000;1;
Organization No	86692741 00000000;1;
Social Security No	A324534422 L000000000;1;
ROC Date	00850930 !9990.00.00;1;
Date	100196 00/00/00;1;
Phone	03532455768 \ (999)9000 0000;1;
Mobile Phone	090789879 000 000000;1;
Pager	060345678 000 000000;1;

Sweden	
Social Security No	6011012110 000000\ 0000;1;
Organization Number	6011012110 000000\ 0000;1;
Zip Code	64700 000 00;1;
Date	950223 !00/00/00;1;
Long Date	19950223 !0000/00/00;1;
Long Time	210515 !90:00:00;1;
Short Time	1345 !90:00;1;

Spain	
Phone	4155551212 !\ (999)0000000;1;
Social Security No	123456789123456 0000000000000;1;
Zip Code	90504 00000;1;
Date	250895 !99/99/00;1;
Long Time	210515 !90:00:00;1;
Short Time	1345 !90:00;1;

Norway	
Phone	22831310 00 09 99 99;1;
Social Security No	01016512345 0000000 0000;1;
Organization Number	01016512345 0000000 0000;1;
Zip Code	0250 0000;1;
Date	170595 00\00\ 00;1;
Date	170595 00/00/00;1;
Long Time	210515 !90:00:00;1;
Short Time	1345 !90:00;1;
Banking	12341212345 0000 00 00000;1;

Netherland	
Phone	205405400 !\00099\ 0000999;1;
Social Security No	123456789 000000000;1;
Zip Code	1234AB 0000 >LL;1;
Date	270695 !90/90/00;1;
Long Time	214515 !90:00:00;1;
Short Time	1345 !90:00;1;

Korea	
Phone	025381890 !\ (9999)000 0000;1;

Handphone(Beeper)	0005381895 !000 000 0000;1;
Social Security No	9507271234567 000000 0000000;1;
Zip Code	135080 000 000;1;
Date 1	950727 00 !90 90;1;
Date 2	19950727 !0000 !90 90;1;
Date 3	950727 00!?!90-!90L;1;
Date 4	19950727 !0000!?!90-!90L;1;
Long Time	042520PM !90:00:00 >LL;1;
Long Korean time	042520PM !90 +00! 00+ >LL;1;
Short Time	1625 !90:00;1;
Short Korean Time	1625 !90 +00! ;1;

Japan	
Phone	0353508393 !099999 0999 0000;1;
Zip Code	1510073 000\ 0000;1;
Date	062794 !99/99/00;1;
Time	090515PM !90:00:00>LL;1;

Italy	
Phone	0212345678 !\ (9999\) 0000 0000;1;
Phone with Extension	02123456781234 !\ (9999\) 0000 0000\ .9999;1;
Local Phone	12345678 !0000 0000;1;
Company Tax #	00010010023 000000000000;1;
Tax Payer #	TRMDNC55P27B242Z >LLLLLL00L00L000L;1;
Zip Code	00100 00000;1;
Short Date	130295 !99/99/00;1;
Long Date 1	130295 !99/99/\1\900;1;
Long Date 2	13Gen95 !99 >L<LL \1\900;1;
Long Time	210515 !90:00:00;1;
Short Time	1345 !90:00;1;

Germany	
Phone	06103979 !\ (999 99\) 99999;1;
Extension	2010 ! 9999;1;
Banking	50850150 000\ 000\ 00;1;
BRD ZIP Code	63225 D 00000;1;
Intl ZIP Code	CH50000 >cc 99999;1;
Date	301193 !90/90/00;1;
Short Time	1345 !90:00;1;
Long Time	193059 !90:00:00;1;

France	
Telephone	0299887766 !00\ 00\ 00\ 00\ 00;1;
Bank Card	5555444433332222 !0000 0000 0000 0000;1;
Social Security No	166107811122233 !0000000000000\ 99;1;
Zip Code	92130 !00000;1;
Short Date	2712 !99/99;1;
Long Date 1	271296 !99/99/00;1;
Long Date 2	27Déc1996 !99 >L<LL 0000;1;
Long Time	134515 !90:00:00;1;
Short Time	1345 !90:00;1;

Denmark	
Phone	48140001 00 09 99 99;1;
Phone with Country	48140001 \+45 00 00 00 00;1;
Social Security No	1234567890 000000\ 0000;1;
Zip Code	3450 0000;1;
Zip Code with Country	DK3450 !>LL\ 0000;1;
Date	260195 90\90\.\1\900;1;
Date	260195 90/90/00;1;
Long Time	210515 !90:00:00;1;
Short Time	1345 !90:00;1;

Example 01:(Capturing Social Security Number)

```
SET VAR vSSN TEXT = NULL
CLS
DIALOG 'Enter Social Security Number||000_00_0000;0;_' +
vSSN=20 vEndKey 1 CAPTION 'Example 01' ICON APPS
```

**Example 01a:** (Capturing Social Security Number)

```
SET VAR vSSN TEXT = NULL
SET VAR vMessage TEXT = 'Enter Social Security Number||000_00_0000;0;_'
SET VAR vCaption TEXT = 'Example 01a'
CLS
DIALOG .vMessage vSSN=20 vEndKey 1 CAPTION .vCaption ICON APPS
```



vSSN = 432222986 TEXT

Example 02: (Capturing Phone Number)

```
SET VAR vPhone TEXT = NULL
CLS
DIALOG 'Enter Phone Number||(000) 000_0000;0;_' +
vPhone=20 vEndKey 1 CAPTION 'Example 02' ICON HELP
```

**Example 02a:** (Capturing Phone Number)

```

SET VAR vPhone TEXT = NULL
SET VAR vMessage TEXT = 'Enter Phone Number||(000) 000_0000;0;_'
SET VAR vCaption TEXT = 'Example 02a'
CLS
DIALOG .vMessage vPhone=20 vEndKey 1 CAPTION .vCaption ICON HELP

```



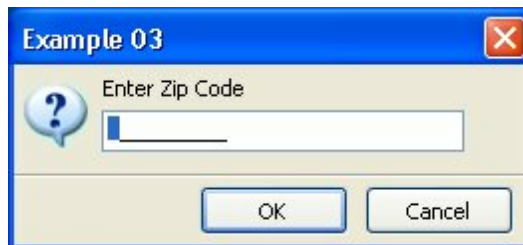
vPhone = 7247240053 TEXT

Example 03: (Capturing Zip Code)

```

SET VAR vZipCode TEXT = NULL
CLS
DIALOG 'Enter Zip Code||00000_0000;0;_' +
vZipCode=20 vEndKey 1 CAPTION 'Example 03' ICON QUESTION

```

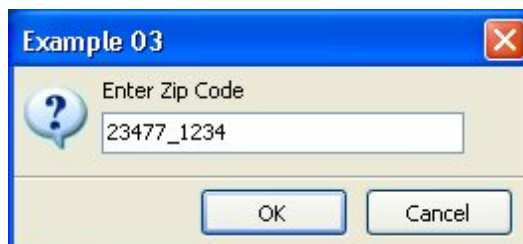


Example 03a: (Capturing Zip Code)

```

SET VAR vZipCode TEXT = NULL
SET VAR vMessage TEXT = 'Enter Zip Code||00000_0000;0;_'
SET VAR vCaption TEXT = 'Example 03a'
CLS
DIALOG .vMessage vZipCode=20 vEndKey 1 CAPTION .vCaption ICON QUESTION

```



vZipCode = 234771234 TEXT

Example 04: (Capturing Credit Card Number)

```

SET VAR vCCNumber TEXT = NULL

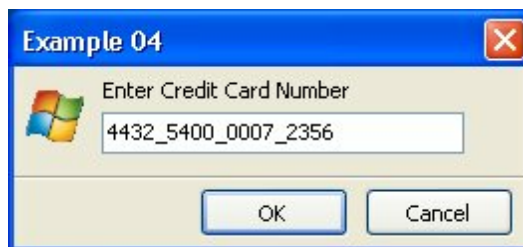
```

```
CLS
DIALOG 'Enter Credit Card Number||0000_0000_0000_0000;0;_' +
vCCNumber=20 vEndKey 1 CAPTION 'Example 04' ICON WINDOWS
```



Example 04a: (Capturing Credit Card Number)

```
SET VAR vCCNumber TEXT = NULL
SET VAR vMessage TEXT = 'Enter Credit Card Number||0000_0000_0000_0000;0;_'
SET VAR vCaption TEXT = 'Example 04a'
CLS
DIALOG .vMessage vCCNumber=20 vEndKey 1 CAPTION .vCaption ICON WINDOWS
```



vCCNumber = 4432540000072356 TEXT

1.5.9.2 OPTION parameters

These parameters can be used to change the "look and feel" of DIALOG window. All parameters and values must be separated by the "|" (pipe) symbol.

1.5.9.2.1 Message

These parameters are specific to changing how the DIALOG "message" is displayed.

MESSAGE_FONT_COLOR *value*

Changes the font color of message area. User can specify the integer value or type one of the predefined color names:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN

- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

MESSAGE_BACK_COLOR *value*

Changes the background color of message area. User can specify the integer value or type one of the predefined color names:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

MESSAGE_BOLD *value*

Makes font in the message area bold style. Values:

- ON
- OFF

MESSAGE_ITALIC *value*

Makes font in the message area italic style. Values:

- ON
- OFF

MESSAGE_UNDERLINE *value*

Makes font in the message area underlined. Values:

- ON
- OFF

MESSAGE_STRIKEOUT *value*

Makes font in the message area strikeout. Values:

- ON
- OFF

MESSAGE_FONT_NAME *value*

Specifies a font name in the message area.

MESSAGE_FONT_SIZE *value*

Specifies a font size (Integer value) in the message area.

1.5.9.2.2 Buttons

These parameters are specific to changing how the DIALOG "buttons" are displayed.

Since you can use a DIALOG window with an edit field using the [OK and CANCEL](#) buttons, as well as only [YES and NO](#) buttons without the edit field, there are several "Button" parameters available.

The follow parameters can be used to alter both buttons at the same time.

BUTTONS_FONT_SIZE *value*

Specifies the font size for buttons

BUTTONS_SHOW_GLYPH *value*

If ON, images are displayed on the "OK and Cancel" or "Yes and No" buttons

BUTTONS_BACK_COLOR *value*

Specifies the background color for buttons

Additional parameters are available within the subchapters of this topic to alter the button properties individually.

1.5.9.2.2.1 OK and CANCEL

These parameters are specific to changing how the DIALOG **OK and CANCEL** buttons are displayed.

BUTTON_OK_CAPTION *value*

Specifies the text for the OK button

BUTTON_OK_FONT_COLOR *value*

Specifies the font color for OK button (color values listed below).

BUTTON_OK_BACK_COLOR *value*

Specifies the background color for OK button (color values listed below).

BUTTON_OK_COLOR *value*

Specifies the color for OK button (color values listed below).

BUTTON_OK_FONT_BOLD *value*

Specifies the bold font for OK button. Values:

- ON
- OFF

BUTTON_OK_FONT_ITALIC *value*

Specifies the italic font for OK button. Values:

- ON
- OFF

BUTTON_OK_FONT_UNDERLINE *value*

Specifies the underline font for OK button. Values:

- ON
- OFF

BUTTON_OK_FONT_STRIKEOUT *value*

Specifies the strike out font for OK button. Values:

- ON
- OFF

BUTTON_OK_FONT_NAME *value*

Specifies the font name for OK button

BUTTON_OK_FONT_SIZE *value*

Specifies the font size for OK button

BUTTON_OK_WIDTH *value*

Specifies the OK button width in pixels

BUTTON_CANCEL_CAPTION *value*

Specifies the text for the Cancel button.

BUTTON_CANCEL_FONT_COLOR *value*

Specifies the font color for the Cancel button (color values listed below).

BUTTON_CANCEL_BACK_COLOR *value*

Specifies the background color for Cancel button (color values listed below).

BUTTON_CANCEL_COLOR *value*

Specifies the color for Cancel button (color values listed below).

BUTTON_CANCEL_FONT_BOLD *value*

Specifies the bold font for Cancel button. Values:

- ON
- OFF

BUTTON_CANCEL_FONT_ITALIC *value*

Specifies the italic font for Cancel button. Values:

- ON
- OFF

BUTTON_CANCEL_FONT_UNDERLINE *value*

Specifies the underline font for Cancel button. Values:

- ON
- OFF

BUTTON_CANCEL_FONT_STRIKEOUT *value*

Specifies the strike out font for Cancel button. Values:

- ON
- OFF

BUTTON_CANCEL_FONT_NAME *value*

Specifies the font name for Cancel button.

BUTTON_CANCEL_FONT_SIZE *value*

Specifies the font size for Cancel button.

BUTTON_CANCEL_WIDTH *value*

Specifies the Cancel button width in pixels

Available Colors:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE

- YELLOW

1.5.9.2.2.2 YES and NO

These parameters are specific to changing how the DIALOG **YES and NO** buttons are displayed.

BUTTON_YES_BACK_COLOR *value*

Specifies the background color for YES button (color values listed below).

BUTTON_YES_CAPTION *value*

Specifies the text for the Yes button.

BUTTON_YES_FONT_SIZE *value*

Specifies the font size for Yes button.

BUTTON_YES_FONT_COLOR *value*

Specifies the font color for Yes button (color values listed below).

BUTTON_YES_COLOR *value*

Specifies the color for Yes button (color values listed below).

BUTTON_NO_CAPTION *value*

Specifies the text for the No button.

BUTTON_NO_FONT_SIZE *value*

Specifies the font size for No button.

BUTTON_NO_FONT_COLOR *value*

Specifies the font color for No button (color values listed below).

BUTTON_NO_COLOR *value*

Specifies the color for No button (color values listed below).

BUTTON_NO_BACK_COLOR *value*

Specifies the background color for NO button (color values listed below).

Available Colors:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN

- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

1.5.9.2.3 Window

These parameters are specific to changing how the DIALOG "window" is displayed.

ICON_FILE *value*

Specifies the external path and image name for the icon (file size must be 16x16 pixels).

WINDOW_CAPTION *value*

To customize the window caption. Supported values:

- OFF - makes window caption invisible
- SMALL - makes window caption small (tool window)

WINDOW_BACK_COLOR *value*

To change the background color of window area. User can specify the integer value or type one of the predefined color names:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

TRANSPARENCY *value*

Set the transparency to specify a layered window with a translucent color. The transparency value specifies the degree of translucency. Values:

0 - 255

THEMENAME *value*

Specifies one of 86 pre-defined Themes, or a custom Theme loaded into R:BASE. See Themes.

NOTE: All previously defined DIALOG commands which are redefined to use Themes should be thoroughly checked prior to putting into production as objects and text may look substantially different.

1.5.9.2.4 Popup Menu

These parameters are specific to adding a "custom popup menu" to the the DIALOG window.

POPUP_ENABLED *value*

Enables the option of bringing up the custom pop-up menus when using the double-click. Values:

- ON
- OFF (default)

POPUP_DIALOG_TYPE *value*

Specifies the Pop-up type when double-clicking on the DIALOG box. Values:

- [LOOKUP](#)
- [DATE](#)
- [TIME](#)
- [DATETIME](#)
- [CALCULATOR](#)

DIALOG_EDIT_HINT *value*

Specifies the hint when you hover your mouse cursor over the dialog edit box, the hint is displayed. Very useful feature to provide hint, such as "Double-Click Here for a List of Customers ..."

AUTOSELECT *value*

Enables the option to select the entire value within the field. Values:

- TRUE
- FALSE

1.5.9.2.4.1 LOOKUP

These parameters are specific to adding a "LOOKUP" custom popup menu to the the DIALOG window.

POPUP_TABLE *value*

Specifies the Table/View Name for Pop-up Menus

POPUP_SELECT *value*

Specifies the SELECT clause for Pop-up Menu Values. *i.e. Company,CustState,CustZip FROM Customer*

POPUP_WHERE *value*

Specifies the WHERE clause for Pop-up Menus Values. *i.e. WHERE CustState = 'PA' ORDER BY Company ASC*

POPUP_DISTINCT *value*

Specifies the option to display Distinct Values for selected columns for Pop-up Menus. Values:

- TRUE
- FALSE

POPUP_RETURN_COLUMN *value*

Specifies the Column Returned for Pop-up Menus. The POPUP_RETURN_COLUMN parameter must be located after your WHERE Clause parameter, if exists, otherwise R:BASE will provide all values specified in your POPUP_SELECT portion of the command.

POPUP_RETURN_TYPE *value*

Specifies the option to return either the selected item in Pop-up Menu or the value of Popup Return Column. Values:

- VALUE
- COLUMN

POPUP_CAPTION *value*

Specifies the text for custom pop-up windows caption.

POPUP_TITLE_FONT_NAME *value*

Specifies the font name for custom pop-up title.

POPUP_TITLE_FONT_SIZE *value*

Specifies the font size for custom pop-up title.

POPUP_TITLE_FONT_COLOR *value*

Specifies the font color for custom pop-up title.

POPUP_TITLE_FONT_BOLD *value*

Specifies the bold font for custom pop-up title. Values:

- ON
- OFF

POPUP_TITLE_FONT_ITALIC *value*

Specifies the italic font for custom pop-up title. Values:

- ON
- OFF

POPUP_TITLE_FONT_UNDERLINE *value*

Specifies the underline font for custom pop-up title. Values:

- ON
- OFF

POPUP_TITLE_FONT_STRIKEOUT *value*

Specifies the strike out font for custom pop-up title. Values:

- ON
- OFF

POPUP_TITLE_TEXT *value*

Specifies the text for custom pop-up title.

POPUP_TITLE_BACK_COLOR *value*

Specifies the background color for custom pop-up title.

POPUP_LIST_FONT_NAME *value*

Specifies the font name for custom pop-up list items.

POPUP_LIST_FONT_SIZE *value*

Specifies the font size for custom pop-up list items.

POPUP_LIST_FONT_COLOR *value*

Specifies the font color for custom pop-up list items.

POPUP_LIST_FONT_BOLD *value*

Specifies the bold font for custom pop-up list items. Values:

- ON
- OFF

POPUP_LIST_FONT_ITALIC *value*

Specifies the italic font for custom pop-up list items. Values:

- ON
- OFF

POPUP_LIST_FONT_UNDERLINE *value*

Specifies the underline font for custom pop-up list items. Values:

- ON
- OFF

POPUP_LIST_FONT_STRIKEOUT *value*

Specifies the strike out font for custom pop-up list items. Values:

- ON
- OFF

POPUP_LIST_BACK_COLOR *value*

Specifies the background color for custom pop-up list items.

MULTISELECT *value*

Enables the option to select the multiple values within a POPUP. Values:

- TRUE
- FALSE

POPUP_LINES *value*

Specifies the number of visible lines to be displayed for list items

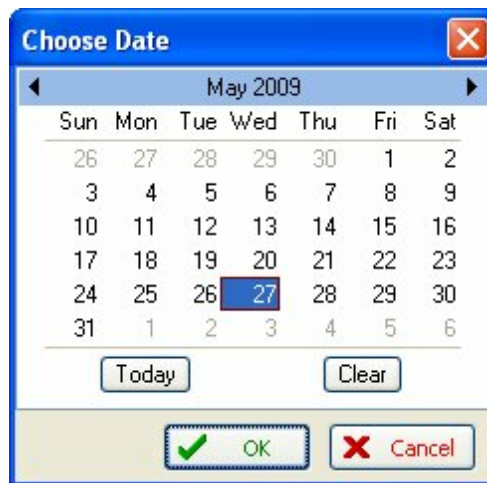
POPUP_SHOW_LINES *value*

Specifies the option to Show Lines between selected columns for Pop-up Menus. Values:

- TRUE
- FALSE

1.5.9.2.4.2 DATE

These parameters are specific to adding a "DATE" custom popup menu to the the DIALOG window.



POPUP_CALENDAR_BACK_COLOR *value*

Specifies the background color of the calendar when POPUP_DIALOG_TYPE is specified as DATE or DATETIME.

POPUP_CALENDAR_DAYS_COLOR *value*

Specifies the color of the calendar days when POPUP_DIALOG_TYPE is specified as DATE or DATETIME.

POPUP_CALENDAR_FILLDAYS_COLOR *value*

Specifies the color of the calendar fill days when POPUP_DIALOG_TYPE is specified as DATE or DATETIME.

POPUP_CALENDAR_DAYSOFWEEK_COLOR *value*

Specifies the color of the calendar days of week when POPUP_DIALOG_TYPE is specified as DATE or DATETIME.

POPUP_CALENDAR_LINES_COLOR *value*

Specifies the color of the calendar lines when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_SELBACK_COLOR *value*

Specifies the color of the selected date background when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_SELFORE_COLOR *value*

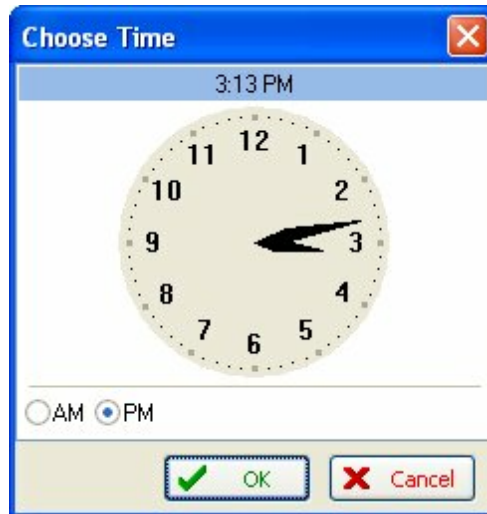
Specifies the color of the selected date foreground when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_TODAYFRAME_COLOR *value*

Specifies the color of today's frame when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

1.5.9.2.4.3 TIME

These parameters are specific to adding a "TIME" custom popup menu to the the `DIALOG` window.



POPUP_CLOCK_BACK_COLOR *value*

Specifies the background color of the clock when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_FACE_COLOR *value*

Specifies the face color of the clock when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_HANDS_COLOR *value*

Specifies the color of the clock's hands when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_NUMBERS_COLOR *value*

Specifies the color of the clock's numbers when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_HOURTICKS_COLOR *value*

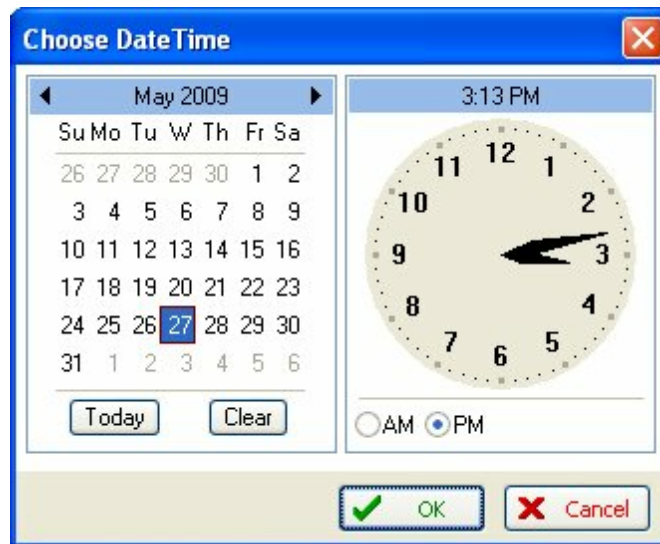
Specifies the color of the clock's hour ticks when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_MINUTETICKS_COLOR *value*

Specifies the color of the clock's minute ticks when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

1.5.9.2.4.4 DATETIME

These parameters are specific to adding a "DATETIME" custom popup menu to the the `DIALOG` window.



POPUP_CALENDAR_BACK_COLOR *value*

Specifies the background color of the calendar when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_DAYS_COLOR *value*

Specifies the color of the calendar days when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_FILLDAYS_COLOR *value*

Specifies the color of the calendar fill days when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_DAYSOFWEEK_COLOR *value*

Specifies the color of the calendar days of week when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_LINES_COLOR *value*

Specifies the color of the calendar lines when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_SELBACK_COLOR *value*

Specifies the color of the selected date background when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_SELFORE_COLOR *value*

Specifies the color of the selected date foreground when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CALENDAR_TODAYFRAME_COLOR *value*

Specifies the color of today's frame when `POPUP_DIALOG_TYPE` is specified as `DATE` or `DATETIME`.

POPUP_CLOCK_BACK_COLOR *value*

Specifies the background color of the clock when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_FACE_COLOR *value*

Specifies the face color of the clock when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_HANDS_COLOR *value*

Specifies the color of the clock's hands when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_NUMBERS_COLOR *value*

Specifies the color of the clock's numbers when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_HOURTICKS_COLOR *value*

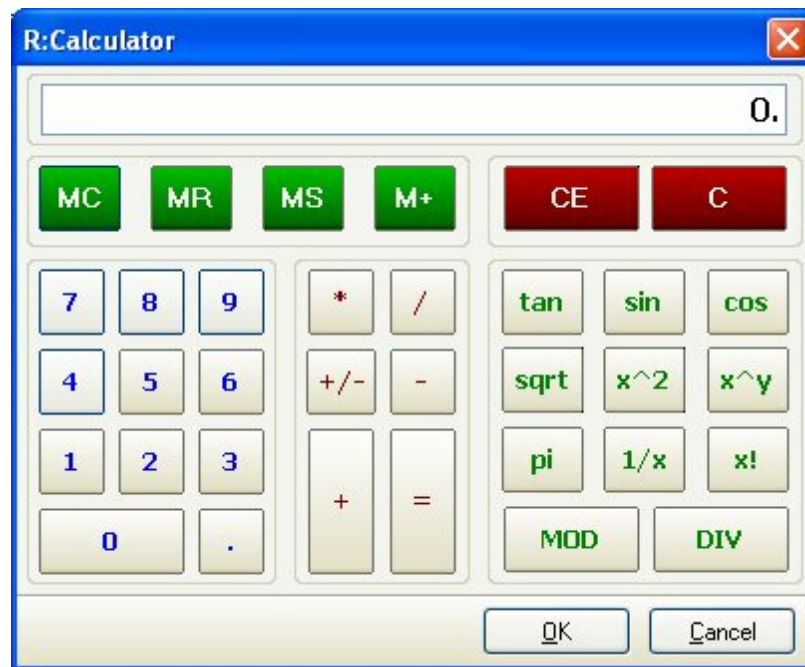
Specifies the color of the clock's hour ticks when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

POPUP_CLOCK_MINUTETICKS_COLOR *value*

Specifies the color of the clock's minute ticks when `POPUP_DIALOG_TYPE` is specified as `TIME` or `DATETIME`.

1.5.9.2.4.5 CALCULATOR

When this parameter is specific a "CALCULATOR" popup will appear for the the DIALOG window.



1.5.10 DIR

Use the DIR command to list the files in a directory.

```
DIR [filespec]
```

Option

filespec

Specifies a filename, including the drive or directory. Wildcards can be used.

About the DIR Command

This command is similar to the operating system DIR command. The R:BASE DIR command pauses at the end of each full screen of information. This command does not support the wide-display (/W) or pause (/P) switches of the operating system DIR command.

If you specify a file name, R:BASE displays information for that file only. You can use an operating system wildcard character (question mark (?) or asterisk (*)) to display a group of files.

DIR lists the disk's volume label and serial number, followed followed by a list of directories and files on the disk. File information includes the file date, time, size in bytes, and name. DIR also displays the number of files (with their total bytes) and directories listed and the total bytes available on the disk. Subdirectory names, if any, are indicated with <DIR>.

Use DIR to test a file specification that uses a wildcard before using the [COPY](#), [RENAME](#), or [ERASE](#) commands. If the directory or disk is empty, or if the specified file(s) is not in the directory, DIR displays:

File not found.

The DIR command supports long (over 80 character) strings, and long strings with spaces.

SET SHORTNAME

If you prefer the traditional format of displaying the directory contents, where the file names are listed first with the short names displayed, you can do so with the SET [SHORTNAME](#) parameter. With SHORTNAME SET ON, DIR lists the file name, extension, size in bytes, and the date and time files were last modified. DIR also displays the number of files listed and bytes available on disk. Subdirectory names, if any, are indicated with <DIR>.

Examples

Example 1:

The following command could be used to review the Help files you have placed in your R:BASE program directory (your results may vary). The display uses SHORTNAME set to OFF.

```
R>DIR *.CHM
```

```
Volume in drive C is ACER
```

```
Volume Serial Number is F43A-E9A3
```

```
Directory of C:\RBTI\RBG9\
```

```
05/14/2008  12:02 AM           2,544,918 Applications.chm
05/14/2008  12:01 AM           149,699 Codelock.chm
06/13/2008  04:34 PM           655,880 DataBrowseEdit.chm
05/13/2008  11:57 PM           960,100 DBExplorer.chm
06/19/2008  06:37 AM           3,602,275 EForms.chm
06/19/2008  06:35 AM           7,860,989 Forms.chm
05/14/2008  12:03 AM           5,099,517 Gateway.chm
06/19/2008  06:35 AM           698,008 Labels.chm
12/07/2007  06:02 PM           829,028 PluginPowerPack.chm
05/14/2008  12:05 AM           1,159,139 QBE.chm
02/19/2008  04:27 PM           289,839 RArchive.chm
05/14/2008  12:00 AM           470,249 RBDefine.chm
06/13/2008  04:34 PM           560,594 REdit.chm
06/19/2008  04:44 PM           5,875,651 RBG9.chm
05/26/2008  02:42 PM           618,075 RBZip.chm
05/16/2008  08:52 AM           54,558 RDWFViewer.chm
06/19/2008  06:38 AM           956,389 Reports.chm
11/06/2007  09:53 AM           86,373 RFTPClient.chm
04/11/2008  10:13 AM           207,788 RMail30.chm
05/14/2008  12:05 AM           120,572 RPrompt.chm
11/20/2007  03:51 PM           583,616 RSpellChecker.chm
05/16/2008  09:46 AM           178,341 RZip.chm
05/30/2008  04:28 PM           190,983 Trace.chm
06/13/2008  04:36 PM           3,498,551 Tutorial.chm
          24 File(s)          37,251,132 bytes
          0 Dir(s)          54,854,406,144 bytes free
```

Example 2:

The following command could be used to review the Help files you have placed in your R:BASE program directory (your results may vary). The display uses SHORTNAME set to ON.

```
R>dir *.CHM
```

```
Volume in drive C is ACER
```

```
Directory of C:\RBTI\RBG9\
```

```

APPLIC~1 CHM  2544918  5-14-08  12:02a Applications.chm
Codelock chm  149699  5-14-08  12:01a Codelock.chm
DATABR~1 CHM  655880  6-13-08  4:34p DataBrowseEdit.chm
DBEXPL~1 CHM  960100  5-13-08  11:57p DBExplorer.chm
EForms  chm  3602275  6-19-08  6:37a EForms.chm
Forms  chm  7860989  6-19-08  6:35a Forms.chm
Gateway chm  5099517  5-14-08  12:03a Gateway.chm
Labels  chm  698008  6-19-08  6:35a Labels.chm
PLUGIN~1 CHM  829028  12-07-07  6:02p PluginPowerPack.chm
QBE  chm  1159139  5-14-08  12:05a QBE.chm
RArchive chm  289839  2-19-08  4:27p RArchive.chm
RBDefine chm  470249  5-14-08  12:00a RBDefine.chm
RBEEdit  chm  560594  6-13-08  4:34p RBEEdit.chm
RBG9  chm  5875651  6-19-08  4:44p RBG9.chm
RBZip  chm  618075  5-26-08  2:42p RBZip.chm
RDWFVI~1 CHM  54558  5-16-08  8:52a RDWFViewer.chm
Reports chm  956389  6-19-08  6:38a Reports.chm
RFTPCL~1 CHM  86373  11-06-07  9:53a RFTPClient.chm
RMail30 chm  207788  4-11-08  10:13a RMail30.chm
RPrompt chm  120572  5-14-08  12:05a RPrompt.chm
RSPELL~1 CHM  583616  11-20-07  3:51p RSpellChecker.chm
RZip  chm  178341  5-16-08  9:46a RZip.chm
Trace  chm  190983  5-30-08  4:28p Trace.chm
Tutorial chm  3498551  6-13-08  4:36p Tutorial.chm
    24 File(s)      37251132 bytes
    24 Dir(s)      54854406144 bytes free

```

1.5.11 DISCONNECT

Use the DISCONNECT command to write any modifications of the currently open database to disk, then close the database.

DISCONNECT

About the DISCONNECT Command

Use the DISCONNECT command before copying database files, or when you want to leave the open database without opening another database. If you keep your database on a floppy disk, use DISCONNECT to close the database before you remove the floppy disk.

The [CONNECT](#) and [EXIT](#) commands also close an open database.

If transaction processing is on when you execute DISCONNECT, R:BASE [COMMITs](#) (saves) any open transactions.

Example

The following command closes an open database:

```
DISCONNECT
```

1.5.12 DISPLAY

Use the DISPLAY command to display up to 24 lines of text that is stored in a screen file. The DISPLAY command is ONLY supported in R:BASE for DOS.

```
DISPLAY scrnname [ IN procfile ] [ AT scrnrow [ ,scrncol ] ]
```

Options

AT scrnrow

Specifies the top line of the display (1-24).

IN procfile

Refers to a procedure file processed through [CodeLock](#).

,scrncol

Specifies the column of the upper-left corner of the display (1-80).

scrnname

Refers to a screen block stored in a procedure file or to any disk file whose contents are to be displayed.

About the DISPLAY Command

A screen file can be a procedure, ASCII, or snapshot file.

To produce a screen file that you can display with the DISPLAY command, use any of the following methods:

- The **Tools: R:BASE Editor** menu option in R:BASE for Windows.
- A word processor that creates ASCII files.

When you use the SNAP command, the snapshot file contains two screen attributes- colors and reverse video. DISPLAY interprets screen attributes stored in the snapshot file so the screen appears exactly the same as when the snapshot was taken.

When you use the **Take a Snapshot** option from the **Utilities** menu in R:BASE for Windows, the current screen is captured in bitmap form in the Windows Clipboard Viewer. This bitmap file cannot be used by the DISPLAY command.

When you display text from either an ASCII or snapshot file, the *scrname* option refers to the name of the screen file. R:BASE displays the entire contents of the screen file, which could look like the following lines.

```
This is the first line that displays on the screen
You can display up to 24 lines of text
```

When you display text that is stored in a procedure file created by CodeLock, you must use the IN *procfile* option. In this case, *scrname* refers to a screen block in the procedure file. A screen block in a procedure file looks similar to following lines. The word \$SCREEN on the first line identifies the block of text as a screen block. The second line contains the screen block name. Supply a drive letter if the screen file is not stored on the default drive, and a path if it is not in the current directory.

```
$SCREEN
```

```
scrnname
```

```
This is the first line that displays on the screen
You can display up to 24 lines of text.
```

Examples

The following command displays the screen named *screen1* in the procedure file MYPROC.APX on drive

B:.

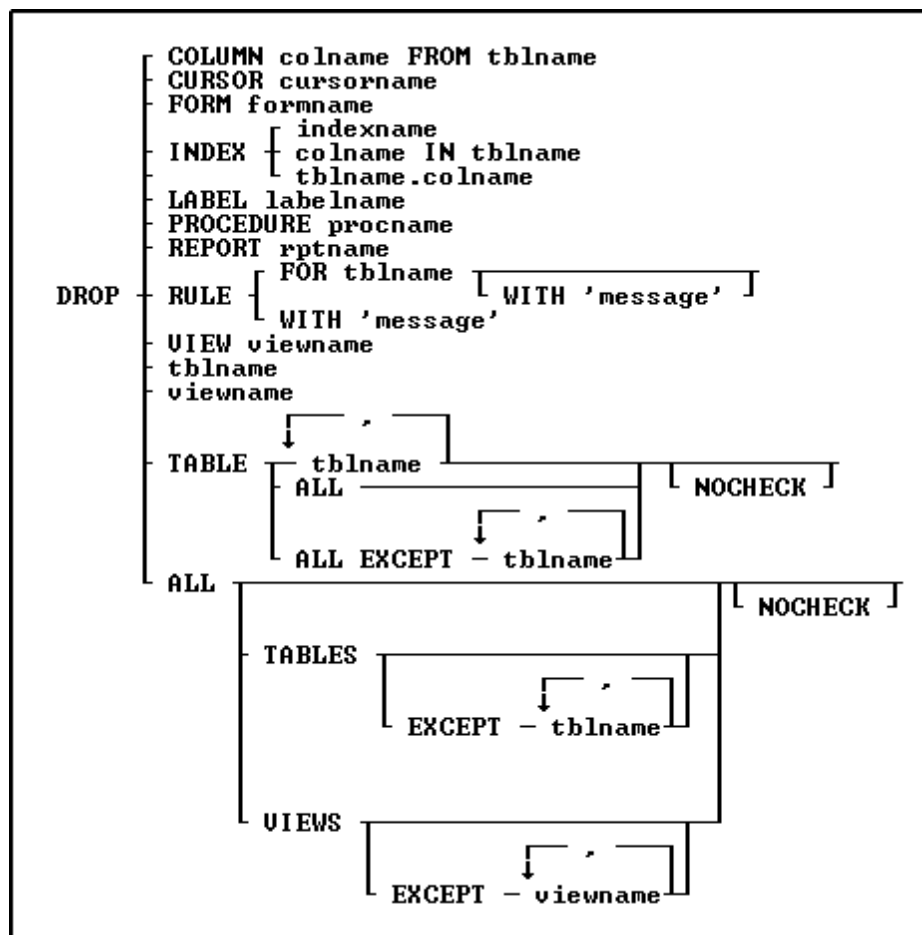
```
DISPLAY screen1 IN b:myproc.apx
```

In the following example, the first command line sets variable *v1* to a TEXT data type for text entry. The second command line displays the text that is stored in a screen file named MAINMENU in the current directory on the default drive. The third command line writes a user prompt starting at row 16, column 20 on screen. It accepts a single character as a response from the user.

```
SET VARIABLE v1 TEXT
DISPLAY mainmenu
FILLIN v1=0 USING 'Enter a letter (A-F)' AT 16,20
```

1.5.13 DROP

Use the DROP command to remove the specified item from the database.



Options

,

Indicates that this part of the command is repeatable.

ALL

Removes all tables from the database.

ALL EXCEPT tblname

Removes all tables from the database except the table(s) listed.

ALL EXCEPT viewname

Removes all views from the database except the view(s) listed.

colname IN tblname

Removes the index from the specified column in the specified table.

COLUMN colname FROM tblname

Removes a column's structure and data from the specified table.

CURSOR cursor

Removes the specified cursor. The DROP CURSOR command removes a cursor definition from memory, therefore freeing memory.

Cursors are dropped when you use the [CONNECT](#) command to open another database, or the [DISCONNECT](#) command to close the database.

FOR tblname

Removes the rule from the specified table for the specified table.

FORM formname

Removes the specified form.

INDEX

Removes the index from the column in the specified table.

indexname

Removes a named index.

LABEL labelname

Removes the specified label.

NOCHECK

Eliminates the confirmation message.

PROCEDURE procname

Removes the specified Stored Procedure from the database.

REPORT rptname

Removes the specified report.

RULE

Removes the specified rule.

TABLE

Removes the specified table.

tblname

Specifies the table name to be removed.

tblname.colname

Removes the index from the specified column in the specified table.

VIEW viewname

Removes the specified view.

viewname

Specifies the view name to be removed.

WITH 'message'

Removes the specified rule from all tables to which it applies. Omitting the WITH 'message' option removes all rules.

About the DROP Command

The table below lists guidelines for using the DROP command.

When you drop...	You must...
A column used in the definition of a computed column	Remove the computed column first.
A column used in a form, report, or label	Delete the column from the form, report, or label definition.
A column used in a rule	Delete the rule for that column in any table from which you removed the column.
A form, report, label, table, or view used in an application	Revise your application to reflect the changes in the database after dropping the form, report, or label.
A view or table used in a form report or label	Delete the form, report, or label from the database after dropping the form, report, or label.
A column or table used in a view	Delete the view from the database.
A table that has rules	Delete the rule with DROP if the table you are dropping is used in the WHERE clause of a rule definition. For example, a table name is used in a WHERE clause of a rule definition when a rule is used to verify a value in one table against values in another table.

R:BASE deletes the rules if the table is the table on which the rules are based.

After running the DROP command, the database item is gone, but the disk space the item occupied is not available. To recover that space, use the [PACK](#) or [RELOAD](#) commands.

You can rebuild a dropped index using [CREATE INDEX](#).

When [STATICDB](#) is set on-which activates a read-only schema mode-DROP is unavailable.

Database Access Rights with DROP

The DROP command requires that you enter the database owner's user identifier if a user identifier has been assigned with the [GRANT](#) command. However, if a user has CREATE or ALTER access rights, that user can use the DROP command to drop tables or any columns in tables to which the user has rights.

Removing Rules with DROP RULE

Before you remove a rule with the DROP RULE command, enter a [SELECT](#) command to verify that you would be removing the correct rule from the correct table(s). Use the conditions in a WHERE clause to enter the exact message and any table names that you plan to use in the DROP command. Once you have verified that the message would remove the correct rules, proceed with the DROP RULES command. For example, to verify that a DROP command with the message "Model number must be unique" would remove only the rules you want to remove, enter the following SELECT command. R:BASE would display all the rules for all the tables in the database to which this message applies.

```
SELECT * FROM SYS_RULES WHERE SYS_MESSAGE = 'Model number must be unique'
```

Examples

The following command removes the *empext* column from the *employee* table.

```
DROP COLUMN empext FROM employee
```

The following two command lines show alternative ways to remove the index from the *custid* column in the *transmaster* table.

```
DROP INDEX custid IN transmaster
DROP INDEX transmaster.custid
```

The following command removes from the database all rules with the message 'Model number must be

unique.'

DROP RULE WITH 'Model number must be unique'

The following command removes any rule from the *product* table that starts with the message 'Model number.' You can use the wildcard character for MANY (%) in a message.

DROP RULE FOR product WITH 'Model number%'

The following command removes the cursor named *cursor1* from memory.

DROP CURSOR cursor1

1.5.14 DUPROW

Use the DUPROW command to duplicate the current row of data currently being edited or entered in a form.

DUPROW

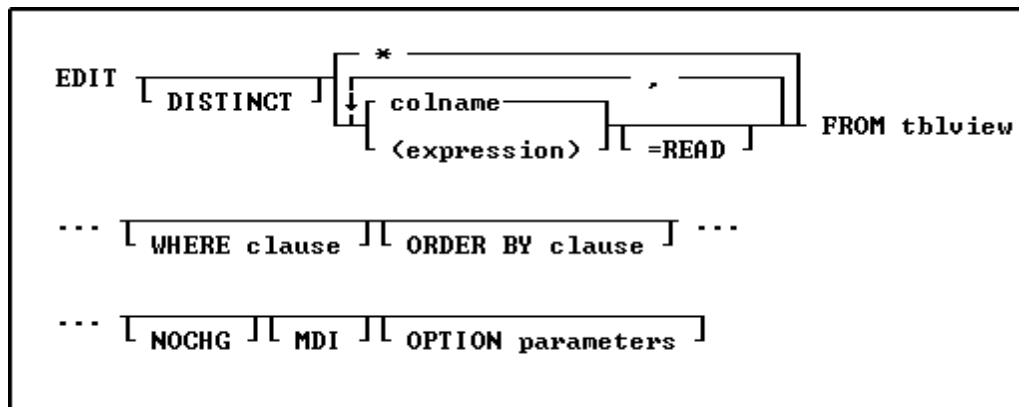
About the DUPROW Command

Use the DUPROW command to duplicate the current row of data currently being edited or entered in a form.

1.6 E

1.6.1 EDIT

Use the EDIT command to change data.



Options

DISTINCT

Suppresses the display of duplicate rows. If you specify DISTINCT, you cannot change the data that is displayed.

*

Specifies all columns.

,

Indicates that this part of the command is repeatable.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

<expression>

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

=READ

Specifies that the column is read-only, preventing changes to data.

FROM tbleview

Specifies the table.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

MDI

Specifies a modeless edit window-a modeless edit window allows you to access other windows without closing the edit window first.

OPTION parameters - provides the ability customize the Data Editor Window. These enhanced parameters are supported at the R> Prompt, in command files, EEPs and Stored Procedures.

WINDOW_STATE value

To setup Data Editor window state with the following supported values:

MAXIMIZED - show maximized
NORMAL - show normalized
MINIMIZED - show minimized

CAPTION value

To change Data Editor window caption.

LOCK nnn

To lock a number of columns in Data Editor window.

ROWVIEW

To open Data Editor window in "View as Row" mode.

READONLY value

To open Data Editor window in Read-Only mode. (Default is OFF)

- ON
- OFF

MODAL value

To open Data Editor as modal/non-modal window. (Default is ON)

- ON
- OFF

EXPORTRESULT *value*

To enable/disable the "Export Result to->" menu item. (Default is ON)

- ON
- OFF

SAVERESULT *value*

To enable/disable the "Save Data As->" menu item. (Default is ON)

- ON
- OFF

PRINTDATA *value*

To enable/disable the "Print Data" menu item. (Default is ON)

- ON
- OFF

About the EDIT Command

The EDIT command starts the Data Editor and displays the data you specify. In a multi-user environment, R:BASE checks whether the row of data you are editing has been changed by another user and notifies you if it has - this is called concurrency control.

You can use the EDIT command to modify data when the following conditions are met:

- You have not used the DISTINCT option in the command.
- You have rights to change the data in the specified table. Access rights are assigned with the [GRANT](#) command.

Notes:

- The EDIT command will display the data within a multi-table View (a View based on multiple tables), but the data will not be editable.
- A View with a [GROUP BY](#) parameter is also not editable.

Examples**Example 01:**

The following command displays all columns in the *transmastertable* and orders the rows by the customer identification numbers contained in the *custid* column.

```
EDIT * FROM transmaster ORDER BY custid
```

Example 02:

The command below displays the following columns from the *customertable*: *custid*, *company*, *custaddress*, *custcity*, *custstate*, and *custzip*. The only rows that are displayed are rows that have 100 for a customer identification. The *custid* and *company* columns are specified to be read-only and cannot be edited. The command specifies that the display of the *company* column is to be ten characters wide.

```
EDIT custid=READ, company=10=READ, custaddress, custcity, +
custstate, custzip FROM customer WHERE custid = 100
```

Example 03:

The following command open all records for the Customer table with a maximized Data Editor window and "Customer Data" as the caption.

```
EDIT ALL FROM customer OPTION WINDOW_STATE MAXIMIZED|CAPTION Customer Data
```

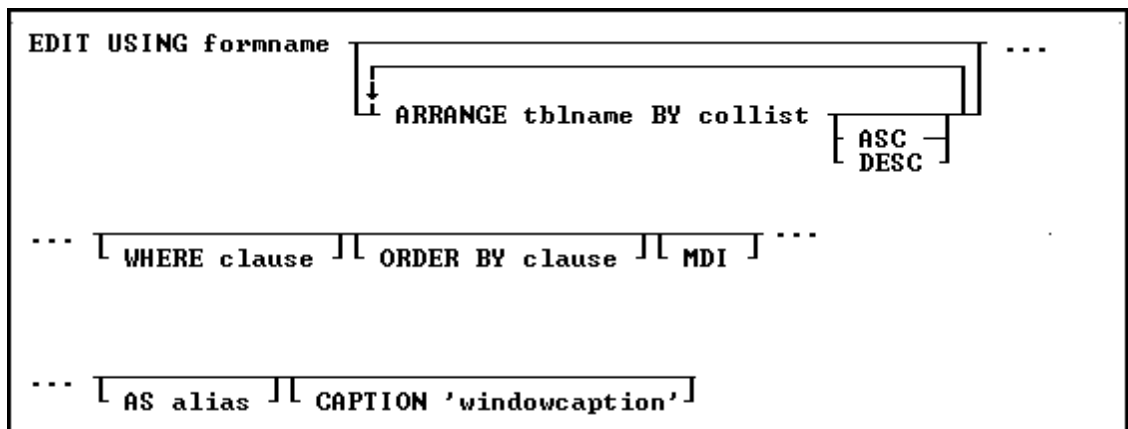
Example 04:

The following will start the Data Editor for *custid* 125, displaying the data as a single row in read only mode.

```
EDIT ALL FROM customer WHERE custid = 125 OPTION WINDOW_STATE NORMAL|ROWVIEW|
READONLY ON
```

1.6.2 EDIT USING

Use the EDIT USING command to display a form with which to view, add, change, or delete data.



Options

formname

Specifies a form name. In R:BASE eXtreme 9.0 (32), the form name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the form name is limited to 128 characters.

ARRANGE tblname BY collist

Sorts the rows displayed by a form's lower-level tables. Each lower-level table in the form that you want to sort requires its own ARRANGE clause. Now you can specify up to twenty-five tables to ARRANGE and up to five columns in each table on which R:BASE will sort.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

MDI

This option is used in R:BASE for Windows only; specifies a modeless form-a modeless form allows you to access other windows without closing the form first.

AS alias

This option is used in R:BASE for Windows only. Allows the user to specify a name for this instance of the

form. This option is used in conjunction with the MDI option so that the user can programmatically specify the form with the [SETFOCUS](#) command.

CAPTION 'windowcaption'

This option is used in R:BASE for Windows only. Specifies a caption for the window title bar.

Example:

```
EDIT USING formname CAPTION 'Form Caption'
```

```
EDIT USING formname MDI
```

About the EDIT USING Command

The EDIT USING command displays data in a previously created form. When the form was created, the user who created the form decided which database actions could be used on the form's tables. These database actions appear on the **Forms** menu.

In a multi-table form, the WHERE and ORDER BY clauses apply only to the first table in the form.

The EDIT USING command allows a form that includes a form called from an entry/exit procedure (EEP). The number of times a form can be invoked within another form is limited only by the amount of stack space available.

In R:BASE for Windows EDIT MDI *formname* AS *alias* can be used to start a form window that can later be accessed with a [SETFOCUS](#) command.

Notes:

- The EDIT USING command will display the data in a form based upon a multi-table View (a View based on multiple tables), but the data will not be editable.
- A View with a [GROUP BY](#) parameter is also not editable.

Examples

The following command displays the form named *custform* with the rows from the first table, ordered by customer identification.

```
EDIT USING CustomerEdit ORDER BY CustID
```

The command below displays the form named *custform* with the rows from the first table that have a customer identification number equal to 100.

```
EDIT USING CustomerEdit WHERE CustID = 100
```

The database RRBYW8 contains a five-table form called *SalesTransactions*. The command below shows how the form's four subordinate tables, *InvoiceHeader*, *InvoiceDetail*, *Employee* and *Contact*, can be sorted using an ARRANGE clause. The ARRANGE clause sorts the second table, *InvoiceHeader*, by the *TransID* column, in ascending order, sorts the third table, *InvoiceDetail*, by the *DetailNum* column, in ascending order, sorts the fourth table, *Employee*, by *EmplName* and *EmpFName*, in ascending order and the fifth table, *Contact*, by the *ContLName* and *ContFName*, in ascending order.

```
CONNECT RRBYW8
```

```
EDIT USING SalesTransactions +
ARRANGE InvoiceHeader BY TransID ASC, +
ARRANGE InvoiceDetail BY DetailNum ASC, +
ARRANGE Employee BY EmplName ASC, EmpFName ASC, +
ARRANGE Contact BY ContLName ASC, ContFName ASC +
```

ORDER BY CustID

1.6.3 ENTER

Use the ENTER command to use a form to add information to tables.

```
ENTER [ USING ] formname [ FOR n ROWS ] [ MDI ] ..  
... [ AS alias ] [ CAPTION 'windowcaption' ]
```

Options

USING

Specifies a modal form - modal means that you must close the form before you can click in another window. If you do not specify an option, a modal form is the default. In R:BASE for DOS, all forms are modal.

formname

Specifies a form name. In R:BASE eXtreme 9.0 (32), the form name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the form name is limited to 128 characters.

FOR n ROWS

Limits to *n* (an integer) the number of rows that can be entered into the form's first table during one session of data entry.

MDI

Specifies a modeless form - a modeless form allows you to access other windows without closing the form first. If you do not specify an option, a modal form is the default.

AS alias

Allows the user to specify a name for this instance of the form. This option is used in conjunction with the MDI option so that the user can programmatically specify the form with the [SETFOCUS](#) command.

CAPTION 'windowcaption'

Specifies a caption for the window title bar.

About the ENTER Command

This command displays a form for data entry. When the form was created, the creator of the form specified which database actions the user could use on the form's tables. These database actions appear on the **Forms** menu.

Notes:

- The ENTER command will not enter data in a form based upon a multi-table View (a View based on multiple tables), as the data is not editable.
- A View with a [GROUP BY](#) parameter is also not editable.

Examples

The following command displays the *transform* form and allows you to enter one row of data to the first table served by *transform*. You can enter as many rows of data in subsequent tables as needed for the one row entered in the first table. After you enter the specified number of rows, R:BASE returns you to the R> Prompt, or the next line in the command file. Specifying the number of rows of data to enter is useful in applications that require other actions to take place after loading each entry.

ENTER tranform FOR 1 ROW

1.6.4 ERASE

Use the ERASE command to delete a file.

ERASE filespec

Options

filespec

Specifies a file name with a drive and path specification in the form D:\PATHNAME\FILENAME.EXT.

About the ERASE Command

If the file is in the current directory, you can omit its path name in the file specification.

R:BASE requires you to close a database before erasing it. ERASE is irreversible. When you use wildcards with ERASE at the R> Prompt, R:BASE prompts you to press [Esc] to cancel, or [Enter] to delete the files.

Examples

The following command erases all files in the OLDSALES directory on the current drive.

```
ERASE \oldsales\*.*
```

The following command lines close then erase the four files of the *mydb* database in the OLDDATA directory on drive E:.

```
DISCONNECT  
DELETE e:\olddata\mydb.rb?
```

1.6.5 EXIT

Use the EXIT command to leave R:BASE and return to the operating system. EXIT closes an open database, releases memory taken by R:BASE, and releases file handles.

EXIT

1.6.6 EXITFORM

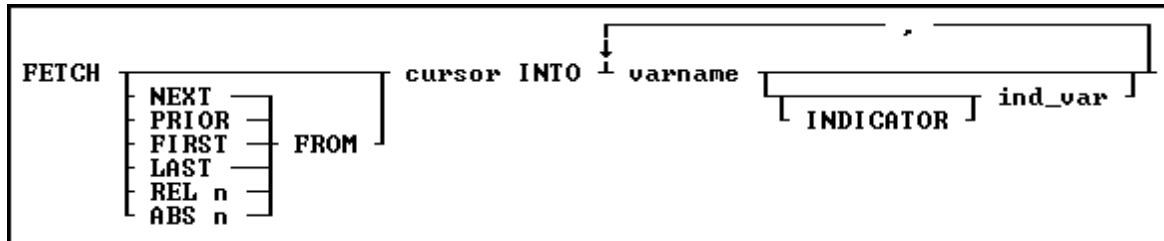
Use the EXITFORM command to exit the current form currently being edited or entered in R:BASE for DOS. Under Windows this is equivalent to the [CLOSEWINDOW](#) command.

EXITFORM

1.7 F

1.7.1 FETCH

Use the FETCH command to position the cursor on a row specified by the [DECLARE CURSOR](#) command, and place values from the columns into global variables.



Options

ABS *n*

The value *n* is the *n*th row in the cursor list. The current cursor location is not relevant. Positive numbers count from the first row in the list. Negative numbers force an end-of-data condition. This option applies only to scrolling cursors, which are defined with the `DECLARE CURSOR` command.

cursor INTO

Names the cursor from which to fetch data into the specified variable list.

FIRST

Specifies the first row in the cursor list. This option applies only to scrolling cursors, which are defined with the `DECLARE CURSOR` command.

INDICATOR ind_var

Stores the status of the variable: non-null (0) or null (-1).

LAST

Specifies the last row in the cursor list. This option applies only to scrolling cursors, which are defined with the `DECLARE CURSOR` command.

NEXT

Specifies the next entry the cursor points to. This option applies only to scrolling cursors, which are defined with the `DECLARE CURSOR` command.

PRIOR

Specifies the prior entry the cursor points to. This option applies only to scrolling cursors, which are defined with the `DECLARE CURSOR` command.

REL *n*

Moves the cursor *n* rows. Positive integers move forward, and negative integers move backwards. For example, if *n* is 5, the cursor moves forward 5 rows. This option applies only to scrolling cursors, which are defined with the [DECLARE CURSOR](#) command.

varname

Specifies a variable name, which must be unique among the variable names within the database. In R:BASE eXtreme 9.0 (32), the variable name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the variable name is limited to 128 characters.

About the FETCH Command

FETCH moves the cursor to the next available row referred to by the `DECLARE CURSOR` command and also accommodates scrollable cursors specified by `DECLARE CURSOR`. FETCH retrieves the values of columns in the order in which the columns were specified by `DECLARE CURSOR`. The [LIST CURSORS](#) command lists all the defined cursors.

FETCH *cursorname* without any variable specification will retrieve the next row from the cursor. Use the

[SET VAR](#) *varname* WHERE CURRENT OF *cursor* to retrieve the columns you need.

Using the Sqlcode Variable

You must check the *sqlcode* variable with each use of FETCH to verify that all rows specified by DECLARE CURSOR have been found. Use the `WHENEVER sqlcode` command to check for SQL processing errors other than data-not-found errors.

Use [WHENEVER NOT FOUND](#) to check for a data-not-found errors. When you use the `WHENEVER NOT FOUND` command, data-not-found error checking is automatic; however, you must use the [LABEL](#) command. When a data-not-found error occurs, control passes to the command line specified by the LABEL command and the subsequent error-handling commands.

Using Indicator Variables

If the data contains null values, use indicator variables to capture the status of a value. If you do not use indicator variables, R:BASE displays an error message when it encounters a null value, but produces no rows.

Placing a Value into a Numeric Variable

If you use FETCH to place a value into a variable that has not been previously defined and has a NUMERIC data type, then that variable acquires the precision and scale of the column from which the value is fetched.

Using the FETCH Command Without Variable Specification

Using `FETCH cursorname` without any variable specification will retrieve the next row from the cursor. Use the [SET VAR](#) *varname* WHERE CURRENT OF *cursor* to retrieve the columns you need.

Example

The following command lines fetch every other row from a table.

```
DROP CURSOR C1
DECLARE c1 SCROLL CURSOR FOR SELECT transid, transdate, +
    netamount FROM transmaster ORDER BY netamount DESC
OPEN c1
FETCH c1 INTO vtransid ind1, vtransdate ind2, vnetamount ind3
SELECT COUNT(*) INTO vtotcount i1 FROM transmaster
SET VAR vcount INT=0
WHILE sqlcode <> 100 THEN
    SET VAR vcount = (.vcount+1)
    WRITE 'Total count', .vtotcount, 'Cursor count', .vcount
    --fetch every other row
    FETCH REL 2 FROM c1 INTO vtransid int1, +
        vtransdate ind2, vnetamount ind3
ENDWHILE
CLOSE C1
DROP CURSOR C1
```

1.7.2 FILLIN

Use the FILLIN command to prompt the user to enter a value at the keyboard.

FILLIN <i>varname</i> [<i>=width</i>] [<i>USING 'message'</i>]

Options

=width

Specifies the width of the entry field. If set to zero, you do not need to press [Enter] after the response. A single key entry is accepted, although not displayed on the screen. When *varname*=0 is specified, you can capture these keystrokes: [Enter], [Esc], [Home], [End], [PgUp], [PgDn], [Up Arrow], [Down Arrow], [Tab], and [Shift]+[Tab]. For example, [PgDn] is recorded as [PgDn] in the variable. The [Down Arrow] key is recorded as [Down] and [Up Arrow] as [Up]. The variable used with =0 is always a TEXT data type. Alternatively, you can omit this option and use the LASTKEY function to capture the last key pressed. (This is useful with [EDIT](#).) If a height and length are specified, the width references the width of the FILLIN box.

USING 'message'

Specifies the message text (a 1- to 80-character text string in the currently defined QUOTES character) to be displayed. You can use a dot variable in place of it. For example, if you enter FILLIN *var1* USING . *var2*, the screen prompt is the contents of *variable2*.

varname

Specifies a variable name, which must be unique among the variable names within the database. In R:BASE eXtreme 9.0 (32), the variable name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the variable name is limited to 128 characters.

About the FILLIN Command

The value entered at the keyboard is stored in a global variable.

If you will be supporting Windows and DOS in a single installation OR are planning on converting from DOS to Windows it is recommended that you replace your FILLIN commands with the [DIALOG](#) command. Under DOS the functionality is almost, if not completely, identical. Under Windows you will find that the DIALOG command is more user friendly.

Setting the Data Type

If the data type of the value that is entered is unambiguous, then the variable is set to that data type. For example, if you enter 123, the variable is given an INTEGER data type. If you wanted the data type to be TEXT, you would have to set the variable's data type. If the data type is questionable—for example, when entering a date—be sure to define the variable's data type. Also, if you use =0 to accept a single keystroke, the variable's data type must be TEXT.

Examples

The following command lines ensure that the data type is TEXT. The quotation marks around the message are not displayed on screen.

```
SET VARIABLE vfill TEXT
FILLIN vfill=10 USING 'Enter a number:'
```

The following command prompts the user for a database name and stores it in the variable v2. The database name could then be used in a [CONNECT](#) command, for example, CONNECT .v2. The dot before the variable indicates that R:BASE is to use the current value of the variable.

```
FILLIN v2 USING 'Enter the name of the database you want to open: '
```

In the following command lines, the first command line sets variable v1 to TEXT and A as its value. The =1 in the FILLIN command lets the user enter a single character or accept the letter A by pressing [Enter]. After the user responds, the program stores the response in the variable v1 and continues to the next line in the command file.

```
SET VARIABLE v1 TEXT = 'A'
FILLIN v1=1 USING 'Enter a letter (A-F): '
```

The following command lines use the =0 option to allow a single keystroke entry, then check the variable for [PgUp] and [PgDn]. If the response is [PgUp], then *vend* contains the value [PgUp] and [GOTO](#) transfers processing to the lines following [LABEL top](#). If the response is [PgDn], control is passed to the lines following [LABEL bottom](#). If neither [PgUp] or [PgDn] is entered, control passes to [LABEL tryagain](#).

and the FILLIN is repeated.

```

LABEL top
.
.
.
LABEL tryagain
FILLIN vend=0 USING 'Press PgUp to start again, PgDn to continue.'
SWITCH (.vend)
CASE '[PgUp]'
    GOTO top
CASE '[PgDn]'
    GOTO bottom
DEFAULT
    GOTO tryagain
ENDSW
.
.
.
LABEL bottom

```

The following example creates a FILLIN box 20 characters wide by three lines deep that can contain a maximum of 100 characters. If you designate a box to contain more characters than the size of the box allows, you automatically create a scrolling box.

```
FILLIN v1=20,3,100 USING 'Notation:'
```

1.7.3 FLUSH

Use the FLUSH command to force database information to be written to disk.

FLUSH

1.7.4 FORMS

Use the FORMS command to start the Form Designer.

```

FORMS [ formname [ MODAL
          tblname ] ]

```

Options

formname

Names an existing form to modify. In R:BASE eXtreme 9.0 (32), the form name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the form name is limited to 128 characters.

MODAL

Allows an EEP command block to launch a Form, Label or Report in a designer. After making appropriate changes, you will have to save the changes and close the designer in order to continue the next command in your EEP.

tblview

This option is for R:BASE for DOS only; names the first table or view that a form serves. Use this option when you are creating a new form. This option is available only if you start to build a form from the R>

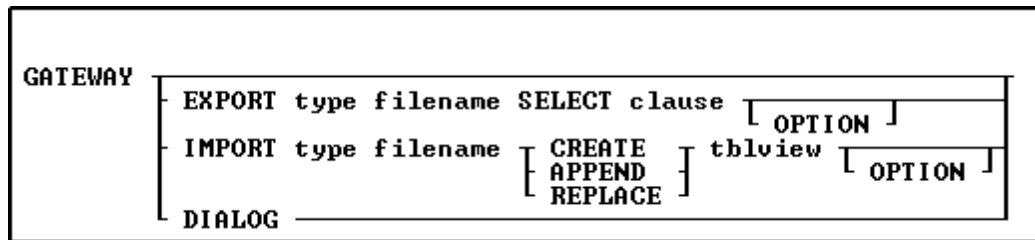
Prompt with a database open. Specify both *formname* and *tblview* when you want to build a new form. R:BASE displays the Form Designer screen where you can begin to define the new form. Use *formname* alone to specify the name of an existing form you want to modify.

1.8 G

1.8.1 GATEWAY

Use the GATEWAY command to start the import/export of data. Using the **GATEWAY** command at the R> Prompt or within command files you can automate the entire task of providing desired data formats to your end-users and/or clients. There are NO limits on importing or exporting numerous records into and out of R:BASE eXtreme for Windows databases with the GATEWAY command.

IMPORTANT NOTE ABOUT GATEWAY: You MUST be aware of both your source file structure and table structure for the command to function properly. The table and file structure MUST coincide with each other.



EXPORT

Specifies that you want to export an R:BASE table or view to an external file.

IMPORT

Specifies that you want to import an external file to an R:BASE table.

DIALOG

Opens the File Gateway dialog for manual Import/Export configuration.

About the GATEWAY Command

Use the Import/Export utility to import a file from another program into a new or existing table in an R:BASE database or export data from an R:BASE database into a file that can be imported into another program or R:BASE database.

You can access File Gateway graphic user interface module either by clicking on "**Open File Gateway to Import/Export Data**" icon on the R:BASE Main Toolbar or typing **GATEWAY** at the R> Prompt or in a command file.

In R:BASE for Windows, you can import or export the following types of files:

- Import Formats include:

- dBase (*.dbf)
- Paradox (*.db)
- Text (*.txt)
- Comma Delimited (*.csv)
- Text Data (*.dat)
- Text Printout (*.prn)
- Tab Delimited (*.tab)
- ASCII (*.asc)
- HTML (*.htm, *.html)
- MS Excel Spreadsheet (*.xls)
- MS Excel Workbook (*.xls)

Lotus 1-2-3 (*.wk1)
 Lotus 1-2-3 (*.wk2)
 Lotus Symphony (*.wr1)
 QuattroPro (*.wq1)
 XML (*.xml)
 MS Access Database (*.mdb)

- Export Formats include:

Paradox (*.db)
 dBase (*.dbf)
 Comma Separated Values (*.csv)
 HTML (*.HTM, *.HTML)
 MS Excel Spreadsheet (*.xls)
 MS Excel Workbook (*.xls)
 MS Word (*.doc)
 Tab Delimited (*.tab)
 SYLK Symbolic Link (*.slk)
 DIF Data Interchange Format (*.dif)
 Lotus 1-2-3 (*.wk1)
 QuattroPro (*.wq1)
 SQL Script (*.sql)
 XML (*.xml)
 MS Windows Clipboard
 Rich Text Format (*.rtf)
 SPSS Format (*.sav)
 Adobe Acrobat Document (*.pdf)
 LDAP Data Interchange Format (*.ldif)

1.8.1.1 EXPORT

Specifies that you want to export an R:BASE table or view to an external file.

```
GATEWAY EXPORT type filename [ SELECT clause ] [ OPTION parameter ]
```

type

Specifies one of the following codes for accepted file formats:

Code	Details
DB	Paradox
DBF	DBASE
CSV	ASCII File, Delimiter Comma, Qualifier Double Quote
FIX	ASCII File, Fixed Width
HTM	HTML
XLS	MS Excel Spreadsheet
XLSW	MS Excel Workbook
DOC	MS Word
TAB	ASCII File, Delimiter Tab, Qualifier None
SLK	SYLK Symbolic Link
DIF	DIF Data Interchange Format
WK1	Lotus 1_2_3
WQ1	QuattroPro
SQL	SQL Script

XML	XML
CLBR	MS Windows Clipboard
RTF	Rich Text Format
SAV	SPSS Format
PDF	Adobe Acrobat Portable Document Format
LDIF	LDAP Data Interchange Format

filename

Specifies the name of the file to import, or the name of the file to export to.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION

Specifies an additional parameter for the output. Options listed may vary based upon selected format. You must Use the pipe "|" symbol to separate OPTION parameters if more than one parameter is used.

- **COL_NAMES**
- **TAB_NAME**
- **TITLE**
- **HTML_PATERN**
- **QUALIFIER**
- **SEPARATOR**
- **BLANK_IF_ZERO**
- **REC_SEP**
- **STYLE**
- **SHOW_PROGRESS**
- **MESSAGES**
- **ACTION**
- **SPECIFICATION_FILE_NAME** - This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

1.8.1.1.1 DB (Paradox)

Syntax:

```
GATEWAY EXPORT DB filename.db +
SELECT clause +
OPTION SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.db

This parameter defines the file name of target file with exported data as Paradox format. File extension must be **db**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**BLANK_IF_ZERO**

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameter: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT DB D:\TEMP\EMPLOYEE.DB SELECT * FROM EMPLOYEE OPTION  
SHOW_PROGRESS ON
```

1.8.1.1.2 DBF (dBASE)

Syntax:

```
GATEWAY EXPORT DBF filename.dbf +  
SELECT clause +  
OPTION SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.dbf

This parameter defines the file name of target file with exported data as dBASE format. File extension must be **dbf**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**BLANK_IF_ZERO**

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The

default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT DBF D:\TEMP\EMPLOYEE.DBF SELECT * FROM EMPLOYEE OPTION  
SHOW_PROGRESS ON
```

1.8.1.1.3 CSV (Comma Separated Values)**Syntax:**

```
GATEWAY EXPORT CSV filename.csv +  
SELECT clause +  
OPTION COL_NAMES +  
|QUALIFIER symbol +  
|SEPARATOR symbol +  
|REC_SEP value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.csv

This parameter defines the file name of target file with exported data as Comma Separated Values format. File extension must be **csv**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

QUALIFIER

Changing the value of this option you can define any qualifier for fields in the target Comma Separated Values text file.

The popular value is Quotes (") but you can define any custom character you wish.

SEPARATOR

Defines the separator of fields in target text file.

The popular values are Semicolon (;), Comma (,), Space (#32) but you can define any other character if you wish.

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

REC_SEP

Defines the separator between lines in the target text file.

The popular value is CRLF (#13#10) or CR(#13) but you can define any other custom string.

Available Parameters: CR / LF / CRLF

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT CSV D:\TEMP\EMPLOYEE.CSV +
SELECT * FROM EMPLOYEE OPTION COL_NAMES ON +
|QUALIFIER " |SEPARATOR , |REC_SEP CRLF +
```


|SHOW_PROGRESS ON |ACTION OPENVIEW

1.8.1.1.4 FIX (Fixed Width)

Syntax:

```
GATEWAY EXPORT FIX filename.txt +  
SELECT clause +  
OPTION COL_NAMES value +  
|BLANK_IF_ZERO value +  
|SPACE_BETWEEN_COLUMNS value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.txt

This parameter defines the file name of target file with fixed width format.

SELECT clause

Specifies and limits rows of data. For more information, see the [SELECT](#) command.

OPTION

COL_NAMES

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

Available Parameters: ON / OFF

SPACE_BETWEEN_COLUMNS

Available Parameters: ON / OFF

If ON, a space will be added between columns. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by email as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT FIX D:\TEMP\EMPLOYEE.TXT +
SELECT * FROM EMPLOYEE OPTION COL_NAMES ON +
|SHOW_PROGRESS ON |ACTION OPENVIEW
```

1.8.1.1.5 HTM(HTML)

Syntax:

```
GATEWAY EXPORT HTM filename.htm +
SELECT clause +
OPTION HTML_PATERN filespec +
|SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.htm

This parameter defines the file name of target file with exported data as HTML format. File extension must be either **htm** or **html**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**HTML_PATERN**

In this option you can define your own pattern of html_file where you can customize any settings _ fonts, colors, metatags, scripts etc Using this option you can integrate the generated html_file into your own web_site.

By default the HTML Pattern have a simple structure:

```
<html>
<head>
<meta http_equiv="Content_Type"
content="text/html; charset=windows_1251">
<meta name="GENERATOR" content="RBG7 Gateway">
<title>Data Exported by R:BASE 7.6 File Gateway</title>
</head>
<body bgcolor="#FFFFFF">
</body>
```

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Examples:

```
GATEWAY EXPORT HTM D:\TEMP\EMPLOYEE.HTML +  
SELECT * FROM EMPLOYEE +  
OPTION SHOW_PROGRESS ON +  
|ACTION OPENVIEW
```

```
GATEWAY EXPORT HTM D:\TEMP\EMPLOYEE.HTML +  
SELECT (EmpLName+', '&EmpFName) AS Employee_List +  
FROM EMPLOYEE OPTION ACTION OPENVIEW
```

Notes:

The generated data table will be inserted before the </body> tag.

1.8.1.1.6 XLS (MS Excel Spreadsheet)

Syntax:

```
GATEWAY EXPORT XLS filename.xls +  
SELECT clause +  
OPTION COL_NAMES value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.xls

This parameter defines the file name of target file with exported data as MS Excel Spreadsheet. File extension must be **xls**.

SELECT clause

Use the SELECT command to identify rows of data from a table or view.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT XLS D:\TEMP\EMPLOYEE.XLS +
SELECT * FROM EMPLOYEE +
OPTION COL_NAMES ON +
|SHOW_PROGRESS ON +
|ACTION OPENVIEW
```

Note:

This option does not use the MS Excel application as OLE_server and is very quick.

1.8.1.1.7 XLSW (MS Excel Workbook)

Syntax:

```
GATEWAY EXPORT XLSW filename.xls +
SELECT clause +
OPTION COL_NAMES value +
```

```
|TAB_NAME value +  
|STYLE value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.xls

This parameter defines the file name of target file with exported data as MS Excel Workbook. File extension must be **xls**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

TAB_NAME

If defined, will be used as the label for the first tab on MS Excel Workbook.

STYLE

This option adds background color to exported data.

Available Parameters:

NORMAL
PRICELIST
MSMONEY
BRICK
DESERT
EGGPLANT
LILAC
MAPLE
MARINE
ROSE
SPRUCE
WHEAT
ZEBRA

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT XLSW D:\TEMP\EMPLOYEE.XLS +
SELECT * FROM EMPLOYEE +
OPTION COL_NAMES ON +
|TAB_NAME EMPLOYEES +
|STYLE ZEBRA +
|SHOW_PROGRESS ON +
|ACTION OPENVIEW
```

Notes:

- Use this option to export data as a MS Excel Workbook file using the MS Excel application as the OLE_server.
- Export process using this type can take a lot of time with large record_sets. However, you have additional options; for example STYLE. If you don't need visual styles in the spreadsheet and want to export data only, you can use the XLS option, which generates the Excel spreadsheet very quickly.
- When providing a file name to export to, you will be required to specify the complete path.

1.8.1.1.8 DOC (MS Word Document)

Syntax:

```
GATEWAY EXPORT DOC filename.doc +
SELECT clause +
OPTION TITLE value +
|COL_NAMES value +
|SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.doc

This parameter defines the file name of target file with exported data as MS Word Document. File extension must be **doc**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION

TITLE

If defined, will be the page header on each page of the document.

COL_NAMES

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT DOC D:\TEMP\EMPLOYEE.DOC +  
SELECT * FROM EMPLOYEE +  
OPTION TITLE Employee List +  
|COL_NAMES ON +  
|SHOW_PROGRESS ON +  
|ACTION OPENVIEW
```

Notes:

- This option uses MS Word application OLE_server to export data and create DOC file. To use this option, you need to have MS Word installed on your computer. Export process using this option can take a lot of time with large record_sets.
- As a faster method we suggest that you use RTF option.

1.8.1.1.9 TAB (Tab Separated Values)

Syntax:

```
GATEWAY EXPORT TAB filename.tab +  
SELECT clause +  
OPTION COL_NAMES value +  
|QUALIFIER symbol +  
|SEPARATOR symbol +  
|REC_SEP value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.tab

This parameter defines the file name of target file with exported data as Tab Separated Values format. File extension must be **tab**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

QUALIFIER

Changing the value of this option you can define any qualifier for fields in the target Tab Separated Values text file.

The popular value is Quotes (") but you can define any custom character you wish.

Use the word **PIPE** if you wish to use the pipe character as the qualifier.

SEPARATOR

Defines the separator of fields in target text file.

The popular values are Semicolon (;), Comma (,), Space (#32), TAB (#9) but you can define any other character if you wish.

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

REC_SEP

Defines the separator between lines in the target text file.

The popular value is CRLF (#13#10) or CR(#13) but you can define any other custom string.

Available Parameters: CR / LF / CRLF

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT TAB D:\TEMP\EMPLOYEE.TAB +  
SELECT * FROM EMPLOYEE +  
OPTION COL_NAMES ON +  
|QUALIFIER " +  
|REC_SEP CRLF +  
|SHOW_PROGRESS ON +  
|ACTION OPENVIEW
```

1.8.1.1.10 SLK (Symbolic Link)

Syntax:

```
GATEWAY EXPORT SLK filename.slk +  
SELECT clause +  
OPTION SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.slk

This parameter defines the file name of target file with exported data as Symbolic Link format. File extension must be **slk**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT SLK D:\TEMP\EMPLOYEE.SLK +
SELECT * FROM EMPLOYEE +
OPTION SHOW_PROGRESS ON +
|ACTION OPENVIEW
```

1.8.1.1.11 DIF (Data Interchange Format)

Syntax:

```
GATEWAY EXPORT DIF filename.dif +
SELECT clause +
OPTION SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.dif

This parameter defines the file name of target file with exported data as Data Interchange Format. File extension must be **dif**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**BLANK_IF_ZERO**

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The

default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT DIF D:\TEMP\EMPLOYEE.DIF +  
SELECT * FROM EMPLOYEE +  
OPTION SHOW_PROGRESS ON +  
|ACTION OPENVIEW
```

1.8.1.1.12 WK1 (Lotus)**Syntax:**

```
GATEWAY EXPORT WK1 filename.wk1 +  
SELECT clause +  
OPTION COL_NAMES value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.wk1

This parameter defines the file name of target file with exported data as Lotus 1-2-3 Format. File extension must be **wk1**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT WK1 D:\TEMP\EMPLOYEE.WK1 +
SELECT * FROM EMPLOYEE +
OPTION COL_NAMES ON +
|SHOW_PROGRESS ON
```

1.8.1.1.13 WQ1 (QuattroPro)

Syntax:

```
GATEWAY EXPORT WQ1 filename.wq1 +
SELECT clause +
OPTION COL_NAMES +
|SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.wq1

This parameter defines the file name of target file with exported data as Quattro Pro Format. File extension must be **wq1**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT WQ1 D:\TEMP\EMPLOYEE.WQ1 +  
SELECT * FROM EMPLOYEE +  
OPTION COL_NAMES ON +  
|SHOW_PROGRESS ON
```

1.8.1.1.14 SQL (SQL Script)**Syntax:**

```
GATEWAY EXPORT SQL filename.sql +  
SELECT clause +  
OPTION QUALIFIER symbol +  
|SHOW_PROGRESS value +  
|MESSAGES value +
```

```
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.sql

This parameter defines the file name of target file with exported data as SQL Script. File extension must be **sql**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**QUALIFIER**

Changing the value of this option you can define any qualifier for fields in the target SQL script file. The default value is single Quote (') but you can specify double Quotes (") if you wish.

Available Parameters: single quote (') / double quotes (")

BLANK_IF_ZERO

Available Parameters: ON /OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT SQL D:\TEMP\EMPLOYEE.SQL +
SELECT * FROM EMPLOYEE +
OPTION SHOW_PROGRESS ON
```

Notes:

Use this option to generate SQL script with dump of selected table data. Each record will include the INSERT INTO statements which you can execute in any SQL compliant database.

1.8.1.1.15 XML (Extended Markup Language)

Syntax:

```
GATEWAY EXPORT XML filename.xml +  
SELECT clause +  
OPTION SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.xml

This parameter defines the file name of target file with exported data as XML format. File extension must be **xml**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**BLANK_IF_ZERO**

Available Parameters: ON /OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT XML D:\TEMP\EMPLOYEE.XML +  
SELECT * FROM EMPLOYEE +  
OPTION SHOW_PROGRESS ON +
```

|ACTION OPENVIEW

NOTES:

Use this option to generate modern and universal XML format to exchange and transfer data between different sources. You can also view any data exported as XML format in MS Internet Explorer 5.0 (or higher) like a tree.

1.8.1.1.16 CLBR (MS Windows Clipboard)

Syntax:

```
GATEWAY EXPORT CLBR +
SELECT clause +
OPTION SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT CLBR +
```



```
SELECT * FROM EMPLOYEE +  
OPTION SHOW_PROGRESS ON +  
|ACTION OPENVIEW
```

Notes:

- Use this option to generate exported data as MS Windows Clipboard Data.
- ACTION OPENVIEW option will launch MS Clipbook Viewer. Then, you have to use Paste option to view all data.
- Refer to MS Clipbook Viewer for more details and handling exported data.

1.8.1.1.17 RTF (Rich Text Format)

Syntax:

```
GATEWAY EXPORT RTF filename.rtf +  
SELECT clause +  
OPTION TITLE value +  
|COL_NAMES value +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.rtf

This parameter defines the file name of target file with exported data as Rich Text Format. File extension must be **rtf**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**TITLE**

If defined, will be the page header on each page of the document.

COL_NAMES

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process the some messages are available (with error messages, for example). Otherwise the export will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT RTF D:\TEMP\EMPLOYEE.RTF +
SELECT * FROM EMPLOYEE +
OPTION TITLE Employee List +
|COL_NAMES ON +
|SHOW_PROGRESS ON +
|ACTION OPENVIEW
```

1.8.1.1.18 SAV (SPSS Format)

Syntax:

```
GATEWAY EXPORT SAV filename.sav +
SELECT clause +
OPTION SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.sav

This parameter defines the file name of target file with exported data as SPSS Format. File extension must be **sav**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process the some messages are available (with error messages, for example). Otherwise the export will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

1.8.1.1.19 PDF (Adobe Acrobat Portable Document)**Syntax:**

```
GATEWAY EXPORT PDF filename.pdf +  
SELECT clause +  
OPTION COL_NAMES +  
|SHOW_PROGRESS value +  
|MESSAGES value +  
|ACTION value +  
|SPECIFICATION_FILE_NAME filename
```

filename.pdf

This parameter defines the file name of target file with exported data as Adobe Acrobat Portable Document Format. File extension must be **pdf**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**COL_NAMES**

If ON, will add a first row with column names. If OFF, will only export data.

Available Parameters: ON / OFF

BLANK_IF_ZERO

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process the some messages are available (with error messages, for example). Otherwise the export will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT PDF D:\TEMP\EMPLOYEE.PDF +
SELECT * FROM EMPLOYEE +
OPTION COL_NAMES ON +
|SHOW_PROGRESS ON +
|ACTION OPENVIEW
```

1.8.1.1.20 LDIF (Data Interchange Format)

Syntax:

```
GATEWAY EXPORT LDIF filename.dif +
SELECT clause +
OPTION SHOW_PROGRESS value +
|MESSAGES value +
|ACTION value +
|SPECIFICATION_FILE_NAME filename
```

filename.dif

This parameter defines the file name of target file with exported data as LDAP Data Interchange Format. File extension must be **dif**.

SELECT clause

Specifies and limits rows of data. For more information, see the "[SELECT](#)" command.

OPTION**BLANK_IF_ZERO**

Available Parameters: ON / OFF

If ON, will export zero values as blank values. If OFF, will export zero values as zero values. The default is set to OFF.

SHOW_PROGRESS

Indicates whether to display status dialog during export process.

Set SHOW_PROGRESS to ON if you wish to show an animated status dialog during the export process. Set SHOW_PROGRESS to OFF if you wish to export data without animated status dialog.

Available Parameters: ON / OFF

MESSAGES

If ON, during export process some messages will be displayed such as -ERROR- messages, if any. If OFF, the entire export process will be in "silent" mode.

Available Parameters: ON / OFF

ACTION

You can define the default action which will be processed after export is complete, for example, to open the target export file in view/edit mode or send this exported file by e_mail as attachment.

Available Parameters: OPENVIEW

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for exporting. The export must be performed once using the graphic user interface "Export Wizard" and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

Example:

```
GATEWAY EXPORT LDIF D:\TEMP\EMPLOYEE.DIF +
SELECT * FROM EMPLOYEE +
OPTION SHOW_PROGRESS ON +
|ACTION OPENVIEW
```

1.8.1.2 IMPORT

Specifies that you want to import an external file to an R:BASE table.

```
GATEWAY IMPORT type filename [ CREATE
                              APPEND
                              REPLACE ] tblview [ OPTION parameter ]
```

type

Specifies one of the following codes for accepted file formats:

Code	Details
DB	Paradox
DBF	dBASE
TXT	ASCII File, Delimiter None, Qualifier None
CSV	ASCII File, Delimiter Comma, Qualifier Double Quote
QSV	ASCII File, Delimiter Comma, Qualifier Single Quote
ISV	ASCII File, Delimiter Semicolon, Qualifier Single Quote
TAB	ASCII File, Delimiter Tab, Qualifier None
CUS	ASCII File, Delimiter and Qualifier based on DB Settings
XLS	MS Excel
WKS	Lotus 1-2-3
HTM	HTML
WQ1	QuattroPro
XML	XML Format
MDB	MS Access Database

filename

Specifies the name of the file to import, or the name of the file to export to.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

Specifies an additional parameter for the input. Options listed may vary based upon selected format. You must Use the pipe "|" symbol to separate OPTION parameters if more than one parameter is used.

Extended parameters supported for each file format

- **SHEET_INDEX**
- **SOURCE_TABLE**
- **SPECIFICATION_FILE_NAME** - This property is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

1.8.1.2.1 DB (Paradox)

Syntax:

```
GATEWAY IMPORT DB +
filename.db {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value
```

filename.db

This parameter defines the file name of target file with data as Paradox format. File extension must be **db**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT DB D:\TEMP\FNAME.DB CREATE AAA OPTION COLUMN_COUNT 5
GATEWAY IMPORT DB D:\TEMP\FNAME.DB APPEND AAA
GATEWAY IMPORT DB D:\TEMP\FNAME.DB REPLACE AAA
```

NOTES:

This option allows to import data from Paradox format using BDE Engine. For use this option, you need to have BDE 32 bit version installed on your computer.

1.8.1.2.2 DBF(dBASE)

Syntax:

```
GATEWAY IMPORT DBF +
filename.dbf {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value
```

filename.dbf

This parameter defines the file name of target file with data as dBASE format. File extension must be **dbf**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION**SPECIFICATION_FILE_NAME**

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT DBF D:\TEMP\FNAME.DBF CREATE AAA OPTION COLUMN_COUNT 5
GATEWAY IMPORT DBF D:\TEMP\FNAME.DBF APPEND AAA
GATEWAY IMPORT DBF D:\TEMP\FNAME.DBF REPLACE AAA
```

NOTES:

This option allows to import data from dBASE format using BDE Engine. For use this option, you need to have BDE 32 bit version installed on your computer.

1.8.1.2.3 ASCII Text (*.txt, *.csv, *.tab)

Syntax:

```
GATEWAY IMPORT TXT +
filename.txt {CREATE/APPEND/REPLACE} +
<tablename> +
```

OPTION SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value

<u>Code</u>	<u>Delimiter</u>	<u>Qualifier</u>
TXT	None	None
CSV	Comma	Double Quotes
QSV	Comma	Single Quotes
ISV	Semicolon	Single Quotes
TAB	Tab	None
CUS	Database DELIMIT Setting	Database QUOTES Setting

filename.txt

This parameter defines the file name of target file with data as ASCII format.

CREATE

|Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT TXT D:\TEMP\FNAME.TXT CREATE AAA OPTION COLUMN_COUNT 5
GATEWAY IMPORT TXT D:\TEMP\FNAME.TXT APPEND AAA
GATEWAY IMPORT TXT D:\TEMP\FNAME.TXT REPLACE AAA
```

1.8.1.2.4 XLS (MS Excel)

Syntax:

```
GATEWAY IMPORT XLS +
filename.xls {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SHEET_INDEX 0 +
|SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value
```

filename.xls

This parameter defines the file name of target file with data as MS Excel format. File extension must be **xls**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION**SHEET_INDEX value**

This parameter is used to specify a specific sheet of the spreadsheet file, starting with 0.

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT XLS D:\TEMP\FNAME.XLS CREATE AAA OPTION SHEET_INDEX 0 |
COLUMN_COUNT 5
GATEWAY IMPORT XLS D:\TEMP\FNAME.XLS APPEND AAA OPTION SHEET_INDEX 0
GATEWAY IMPORT XLS D:\TEMP\FNAME.XLS REPLACE AAA OPTION SHEET_INDEX 0
```

NOTES:

Use this option to import data from MS Excel file using the MS Excel application as OLE_server.

1.8.1.2.5 WKS (Lotus 1-2-3)**Syntax:**

```
GATEWAY IMPORT WKS +
filename.wk1 {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SHEET_INDEX 0 +
|SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value
```

filename.wk1

This parameter defines the file name of target file with data as Lotus 1-2-3 format. File extension must be **wk1**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

SHEET_INDEX value

This parameter is used to specify a specific sheet of the spreadsheet file, starting with 0.

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT WKS D:\TEMP\FNAME.WK1 CREATE AAA OPTION SHEET_INDEX 0 |
COLUMN_COUNT 5
GATEWAY IMPORT WKS D:\TEMP\FNAME.WK1 APPEND AAA OPTION SHEET_INDEX 0
GATEWAY IMPORT WKS D:\TEMP\FNAME.WK1 REPLACE AAA OPTION SHEET_INDEX 0
```

1.8.1.2.6 HTM(HTML)

Syntax:

```
GATEWAY IMPORT HTML +
filename.htm {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value
```

filename.htm

This parameter defines the file name of target file with data as HTML format. File extension must be **htm** or **html**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT HTM D:\TEMP\FNAME.HTM CREATE AAA OPTION COLUMN_COUNT 5
GATEWAY IMPORT HTM D:\TEMP\FNAME.HTM APPEND AAA
GATEWAY IMPORT HTM D:\TEMP\FNAME.HTM REPLACE AAA
```

1.8.1.2.7 WQ1 (QuattroPro)

Syntax:

```
GATEWAY IMPORT WQ +
filename.wq1 {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SPECIFICATION_FILE_NAME filename +
|SHEET_INDEX 0 +
|COLUMN_COUNT value
```

filename.wq1

This parameter defines the file name of target file with data as QuattroPro format. File extension must be **wq1**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

SHEET_INDEX value

This parameter is used to specify a specific sheet of the spreadsheet file, starting with 0.

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT WKQ D:\TEMP\FNAME.WQ1 CREATE AAA OPTION SHEET_INDEX 0 |
COLUMN_COUNT 5
GATEWAY IMPORT WKQ D:\TEMP\FNAME.WQ1 APPEND AAA OPTION SHEET_INDEX 0
GATEWAY IMPORT WKQ D:\TEMP\FNAME.WQ1 REPLACE AAA OPTION SHEET_INDEX 0
```

1.8.1.2.8 XML (Extended Markup Language)

Syntax:

```
GATEWAY IMPORT XML +
filename.xml {CREATE/APPEND/REPLACE} +
<tablename> +
```

```
OPTION SPECIFICATION_FILE_NAME filename +
|COLUMN_COUNT value
```

filename.xml

This parameter defines the file name of target file with data as XML format. File extension must be **xml**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION**SPECIFICATION_FILE_NAME**

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT XML D:\TEMP\FNAME.XML CREATE AAA OPTION COLUMN_COUNT 5
GATEWAY IMPORT XML D:\TEMP\FNAME.XML APPEND AAA
GATEWAY IMPORT XML D:\TEMP\FNAME.XML REPLACE AAA
```

1.8.1.2.9 MDB (MS Access)

Syntax:

```
GATEWAY IMPORT MDB +
filename.mdb {CREATE/APPEND/REPLACE} +
<tablename> +
OPTION SPECIFICATION_FILE_NAME filename +
|SOURCE_TABLE tablename +
|COLUMN_COUNT value
```

filename.mdb

This parameter defines the file name of target file with data as MDB format. File extension must be **mdb**.

CREATE

Creates a destination table with the same name as the imported file.

APPEND

Adds the imported data to the end of the table or view specified in *tblview*.

REPLACE

Replaces the rows in the table or view specified in *tblview* with the imported data.

tblview

Specifies the name of a table or view FOR Import.

OPTION

SOURCE_TABLE value

This parameter is used to specify a specific table of the MDB file.

SPECIFICATION_FILE_NAME

This parameter is used to load a prepared specification with all pre-defined settings for importing. The import must be performed once using the graphic user interface and saved using the "Specifications.." button. The specifications will be stored in a file with the .RGW file extension.

COLUMN_COUNT

This parameter specifies the number of columns to be imported from file. COLUMN_COUNT is only supported when using CREATE in the GATWAY syntax.

Examples:

```
GATEWAY IMPORT MDB D:\TEMP\FNAME.MDB CREATE AAA +
  OPTION SOURCE_TABLE TblName|COLUMN_COUNT 5
GATEWAY IMPORT MDB D:\TEMP\FNAME.MDB APPEND AAA OPTION SOURCE_TABLE TblName
GATEWAY IMPORT MDB D:\TEMP\FNAME.MDB REPLACE AAA OPTION SOURCE_TABLE
TblName
```

1.8.2 GET

Retrieves a Stored Procedure.

```
GET [LOCK] procname TO filename
```

Options**filename**

The name of the ASCII text format file the Stored Procedure is placed in.

LOCK

Locks the procedure so it cannot be locked or unlocked by another user. When a procedure is locked, only the user placing the lock can replace the procedure. The [NAME](#) setting is used for identification of the user.

procname

The name of the procedure to retrieve.

About the GET Command

The GET command is used to read a Stored Procedure from the database into an ASCII file. If the LOCK option is used with the GET command, the procedure cannot be replaced by using the [PUT](#) command.

Rows are copied, not removed, from the source.

Example

The following command retrieves the SetOrderID procedure and places it into a file name SetOrdID.STP.

```
GET SetOrderID TO SetOrdID.STP
```

The following series of commands will retrieve the CreateTempTabs procedure and place it into a file name TempTabs.STP, then Edit the file, and finally replace the Stored Procedure from the file with an updated version.

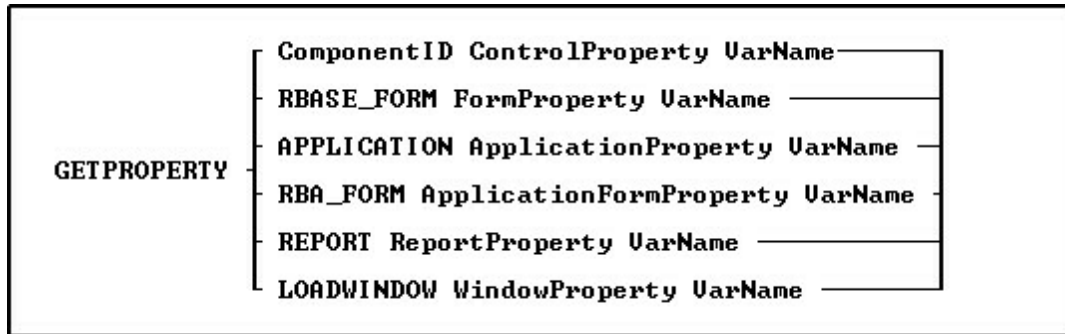
```
GET CreateTempTabs TO TempTabs.STP
```

RBE TempTabs.STP
 PUT TempTabs.STP AS CreateTempTabs

Note: The STP file extension is not required by R:BASE it is merely a suggestion for a meaningful naming convention.

1.8.3 GETPROPERTY

Use the GETPROPERTY command to capture the current properties of forms, form controls, applications, reports, report controls, labels, and label controls.



Options

APPLICATION ApplicationProperty

To capture an application property such as the CAPS lock status, if it is compiled, or the title

Component ID

The unique identifier that you assign to the control (i.e. DB Edit). The unique identifier within the GETPROPERTY command must match the unique identifier listed in the control properties. A "Component ID" field is located in the "Object Properties" for all forms and form controls. A "Component ID" option is located in the speed menu list when you right click on a report/label control. **Wildcards are supported for altering the properties of multiple components.**

ControlProperty

The name of the form/report/label object's property you wish to get the value for

LOADWINDOW WindowProperty

To capture the load window property such as the caption or title. The LOADWINDOW parameter is only supported in R:Compiler

RBA_FORM ApplicationFormProperty

To capture an application form property such as the current theme, the enabled status for an action, or a caption

RBASE_FORM FormProperty

To capture a form property such as height, width, etc.

REPORT ReportProperty

To capture a report property such as the report name, the total page count, or the number of tables using in the report

VarName

The variable name with resulting value as text

Notes:

- Use the following Web URL to create GETPROPERTY command syntax with the **GETPROPERTY Command Syntax Builder**:

<http://www.rbase.com/getproperty/>

- An optional add-on product containing all PROPERTY/GETPROPERTY commands and parameters constructed in an R:BASE application is also available. The product is called [R:Docs](#). Please contact RBTI Sales Staff at sales@rbase.com if you wish to acquire the annual subscription to stand-alone R:Docs database and application.

1.8.4 GOTO

Use the GOTO command in a program to pass control to the commands following the [LABEL](#) command.

GOTO lblName

Option

lblName

Specifies a 1 to 18 character name that labels a line to skip to when a GOTO command is executed in a command or procedure file.

About the GOTO Command

You should limit the use of the GOTO command because GOTO runs more slowly than other R:BASE control structures. Instead, when possible, use a [WHILE](#) loop, [SWITCH](#) structure, or [IF](#) structure to build the command-file logic. Never use GOTO to exit from a WHILE loop or SWITCH structure.

Using the LABEL Command with GOTO

GOTO must have a corresponding [LABEL](#) command within the same command block or file. The LABEL command may precede or follow GOTO in the same command file or, in a procedure file, within the same command block.

You can use a variable containing the name of the label instead of using the specific label name in the GOTO command. To do this, you must use a dot or ampersand variable to tell R:BASE to use the contents of a variable, rather than the variable name as the label name.

Examples

The following example uses a dotted variable containing the name of the label instead of using the specific label name in the GOTO command. If the variable was not a dotted variable, R:BASE would search for a label named *vlabel*. Because it is a dotted variable, R:BASE looks for the correct label name *label1*.

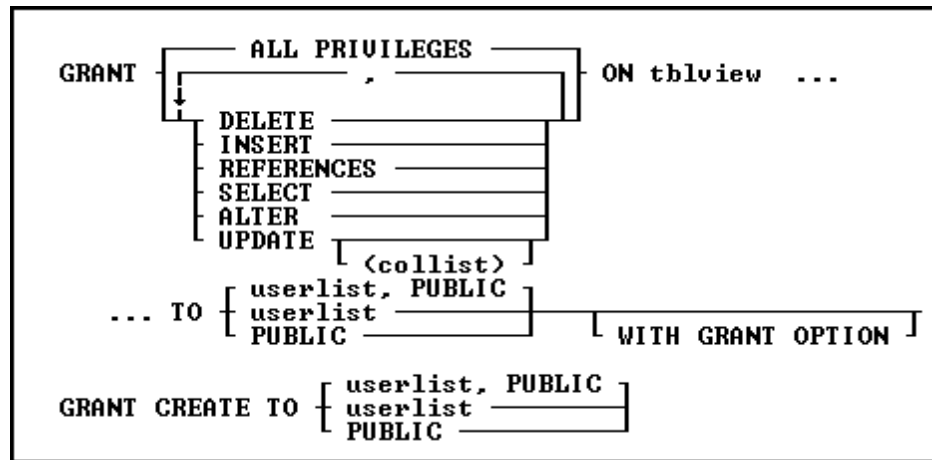
```
SET VARIABLE vlabel = 'label1'
GOTO .vlabel
```

The GOTO *lexit* command in the following example causes the commands following the ENDIF command to be skipped and the [QUIT TO](#) command to be run. The only way the commands between the IF structure and LABEL *lexit* command would be executed would be if the value of *v1* is not greater than the value of *v2*.

```
IF v1 > .v2 THEN
  GOTO lexit
ENDIF
.
.
.
LABEL lexit
QUIT TO caller
```

1.8.5 GRANT

Use the GRANT command to assign privileges to users of a table or view.



Options

,

Indicates that this part of the command is repeatable.

ALL PRIVILEGES

Grants all user privileges on the specified table, or on a view that can be updated.

ALTER

Grants permission to alter specific tables.

CREATE TO

Grants permission to users to create tables using the [CREATE TABLE](#) command. Users who have been granted permission to use this command have all privileges on the tables they create, including the WITH GRANT OPTION. However, users do not have privileges on any other tables in the database unless they are specifically granted permission by the owner.

DELETE

Grants permission to remove rows from the specified table or from a view that can be updated.

INSERT

Grants permission to add rows to the specified table or to a view that can be updated.

ON tblview

Specifies a table or view.

PUBLIC

Grants specified user privileges to all users.

REFERENCES

Grants permission to create a table with a foreign key that references a table with a primary key.

SELECT

Grants permission to display or print data for the specified table or view.

UPDATE (*collist*)

Grants permission to change the values of columns in the specified table or a view that cannot be updated. If you do not include the optional (*collist*), the user can update all columns in the table. If you list columns, the user can update only the specified columns.

userlist

Grants specified user privileges to listed users. You must separate user identifiers with a comma (or the

current delimiter).

userlist, PUBLIC

Grants specified user privileges to listed users and PUBLIC. Users in *userlist* can retain their user privileges if user privileges granted to PUBLIC are revoked. If, for example, *ralph*, *sam*, *jane*, and PUBLIC have been granted certain user privileges, revoking those privileges from PUBLIC would not affect the three listed users. You must separate the user identifier with a comma (or the current delimiter).

WITH GRANT OPTION

Allows the specified users to pass the granted user privileges to other users. When you use the [LIST ACCESS](#) command, an asterisk is displayed in front of the user privilege to show a user can grant the assigned user privilege to others; for example, *SELECT means a user has permission to display or print data for specified tables or views, and can grant SELECT rights to other users.

About the GRANT Command

As the database owner, you must first set your own user identifier. After setting your user identifier, you can assign privileges to other users for the tables or views in your database. You must specifically grant privileges to other users. You can assign privileges for a table to individual users, to PUBLIC, or to both. Each user can have a different set of user privileges for the same table, and you can grant a user the right to grant user privileges to others. You can set your user identifier with the [RENAME OWNER](#) command and assign user privileges to other users by using the GRANT command.

In R:BASE for Windows you can also set your user identifier by choosing **Utilities: Set User ID and Password**. To assign user privileges to other users, choose the **User Privileges** option from the **Utilities** menu.

In R:BASE for DOS, you can also set a user identifier and assign access rights in RBDefine; enter the RBDEFINE command at the R> Prompt.

Granting User Privileges

You grant user privileges or access rights on tables or views, however, UPDATE rights must be granted at the column level and CREATE rights must be granted at the database level. If you assign more than one user privilege in a single GRANT command, separate the user privileges with a comma (or the current delimiter).

You can grant the following user privileges: ALL PRIVILEGES, ALTER, CREATE, DELETE, INSERT, REFERENCES, SELECT, and UPDATE; however, you can grant only the SELECT user privilege on views that cannot be updated.

Using User Identifiers and Passwords

A user identifier can be any unique string of 36 characters (or less) that uniquely identifies a user to the system. To maximize security, create user identifiers that are difficult to guess—such as a random string of letters and numbers. Users can assign passwords to their user identifiers for an added level of security. For information about users assigning passwords see [SET USER](#).

In a database where users have been assigned rights, printing reports requires one of these conditions:

- A user has been granted SELECT privileges on the driving table or view and any look-up tables.
- A user has been granted SELECT privileges or ALL PRIVILEGES on all tables used for the report.
- PUBLIC has been granted SELECT privileges on the driving table or view.
- PUBLIC has been granted SELECT privileges or ALL PRIVILEGES on all tables used for the report.

The only exception to this system of assigning rights is password-protected forms. Passwords assigned to forms, override user privileges assigned with the GRANT command. If a form has not been assigned a password, the user privileges you granted to the tables associated with the form are in effect.

Once R:BASE determines that a user can have access to a password-protected form, R:BASE does not verify user privileges on the underlying tables. Therefore, access to a password-protected form overrides table-level user privileges, making it possible for a user who does not have user privileges on a table to modify the information in that table.

Creating New Tables

To create new tables in a database, a user must be assigned the CREATE user privilege. R:BASE assigns all user privileges to the user for all tables created, including the GRANT user privilege.

A user must be assigned the SELECT user privilege to create a new table from existing tables using the [INTERSECT](#), [JOIN](#), [PROJECT](#), [SUBTRACT](#), or [UNION](#) commands. R:BASE assigns users who use these commands all user privileges on the new table. These user privileges do not include the GRANT user privilege.

Creating Views

[CREATE VIEW](#) also requires the SELECT user privilege on the existing tables. R:BASE assigns users who create views the same user privileges they have on the source table. For views that cannot be updated, R:BASE only assigns users the SELECT user privilege.

Command Authorization Requirements

The following three tables list R:BASE commands and the user privileges they require.

R:BASE Commands that Require the SELECT Access Right

Command	SELECT Access Right on...
BACKUP DATA	Table
BROWSE	Table
COMPUTE	Table
CREATE VIEW	Component tables
CROSSTAB	Table
DECLARE CURSOR	Table
FETCH	Table
INTERSECT	Table 1 and table 2
JOIN	Table 1 and table 2
OPEN CURSOR	Table
PRINT	Driving table/view, and any look-up tables
PROJECT	Table 1
QUERY	Table
SELECT	Table
SET VARIABLE *	Table
SUBTRACT	Table 1 and table 2
TALLY	Table
UNION	Table 1 and table 2
UNLOAD DATA	Table

* SET VARIABLE requires the SELECT user privilege only when the value of the variable is derived from a column.

R:BASE Commands that Require the UPDATE User Privilege

Command	UPDATE User Privilege on...
CREATE INDEX	Column
EDIT ALL	Table. All columns in the table are displayed if you have SELECT permission on the table.
QUERY	Column
UPDATE	Column list

R:BASE Commands that Require Other User Privileges

Command	User Privilege	Access on...
DELETE	DELETE	Table or single-table view.
EDIT USING	UPDATE, SELECT, DELETE, ALL	If the form is protected by a password at either the read or write level, the password on the form is the overriding

	PRIVILEGES	user privilege. If the form is not protected by a password, the owner of the database must grant the specific user privileges on the tables used in the form.
ENTER	INSERT, ALL PRIVILEGES	If the form is protected by a password at either the read or write level, the password on the form is the overriding user privilege. If the form is not protected by a password, the owner of the database must grant the specific user privileges on the tables used in the form.
INSERT	INSERT	Table or single-table view.
LIST ALL	Any user privilege	Any user privilege granted allows users to list all tables for which they have user privileges.
LIST COLUMNS	Any user privilege	Any user privilege granted allows users to display columns for which they have user privileges.
LIST CONSTRAINTS	Any user privilege	Any user privilege granted allows users to display constraints for tables for which they have user privileges.
LIST INDEXES	Any user privilege	Any user privilege granted allows users to display indexes for which they have user privileges.
LIST SYS %	SELECT	The SELECT user privilege allows users to view a generalized list of system tables and views.
LIST TABLE SYS %	SELECT	The SELECT user privilege allows users to view a detailed list of system tables and views.
LIST TABLES	Any user privilege	Any user privilege granted allows users to display tables for which they have user privileges.
LIST VIEWS	Any user privilege	Any user privileges allows users to display views for which they have user privileges.

The following table lists the user privileges and the commands that use them. Some commands appear under more than one user privilege.

User Privileges for R:BASE Commands

Access Right	R:BASE Commands that Require The Access Right		
ALTER	ALTER TABLE	AUTONUM	DROP COLUMN
CREATE	ALTER TABLE	DROP	REVOKE 1
	AUTONUM	GRANT 1	RULES
	BACKUP ALL	PACK	UNLOAD ALL
	BACKUP STRUCTURE	RELOAD	UNLOAD STRUCTURE
	COMMENT ON	RENAME	
	CREATE TABLE	RESTORE	
Database owner's user identifier	ALTER TABLE	DROP	REVOKE 1
	AUTONUM	GRANT 1	RESTORE
	BACKUP ALL	PACK	RULES
	BACKUP STRUCTURE	RELOAD	UNLOAD ALL
	COMMENT ON	RENAME	UNLOAD STRUCTURE
	CREATE TABLE		
DELETE	DELETE	EDIT	LIST 3
Form password 2	EDIT USING 2	ENTER 1	
INSERT	EDIT	ENTER 2	LIST 3
	EDIT USING	INSERT	LOAD
REFERENCES	ENTER 2	INSERT	UPDATE
	EDIT	LIST 3	
	EDIT USING	LOAD	
SELECT	BACKUP DATA	FETCH	SELECT
	BROWSE	INTERSECT	SET VARIABLE 4
	COMPUTE	JOIN	SUBTRACT
	CREATE VIEW	LIST 3	TALLY
	CROSSTAB	PRINT	UNION
	DECLARE CURSOR	PROJECT	UNLOAD DATA
UPDATE	CREATE INDEX	EDIT USING	UPDATE
	EDIT	LIST 3	

1. GRANT and REVOKE do not require the database owner's user identifier for an user privilege that includes GRANT permission.
2. Form passwords override user privileges assigned with the GRANT command. If a form does not have a password, the INSERT, DELETE, SELECT, or UPDATE user privileges are required for the underlying tables.
3. Any user privilege granted allows users to list all tables for which they have user privileges.
4. SET VARIABLE requires the SELECT user privilege only when the value of the variable is derived from a column.

Revoking User Privileges

The database owner can remove user privileges with the REVOKE command. The syntax for the REVOKE command is the same as the syntax for the GRANT command. If you issue the REVOKE ALL PRIVILEGES command without specifying a table, R:BASE revokes all user privileges including ALTER and CREATE.

Examples

The following command grants user privileges to display the view named *slsview* to a specific user-*jane*, and to all users-PUBLIC.

```
GRANT SELECT ON slsview TO jane, PUBLIC
```

The following command grants user privileges to add or remove information to or from the *transmaster* table to any user entering the user identifier *sam* or *ralph*.

```
GRANT INSERT, DELETE ON transmaster TO sam, ralph
```

The following command grants user privileges to display and enter information in the *transmaster* table. Also, the command allows any user entering the user identifier *jane* to pass the SELECT and INSERT user privileges on to other users.

```
GRANT SELECT, INSERT ON transmaster TO jane WITH GRANT OPTION
```

The following command grants a user who is not the database owner permission to alter a specified table.

```
GRANT ALTER ON customer TO John
```

The following command line grants a user who is not the database owner permission to create tables.

```
GRANT CREATE TO John
```

1.9 H

1.9.1 HELP

Use the HELP command to start R:BASE online Help.

```
HELP [ CALL filename helpID ]
```

About the HELP Command

To start the online Help, press [F1]. You can also start *online Help* at the R> Prompt by entering "HELP," or "HELP" followed by a command name.

Press [Shift] + [F1] to display context-sensitive *online Help*.

The help file contains all the R:BASE help information and must be located on the default directory and drive, or on the current operating system path. All help files have the .CHM extension.

1.10 I

1.10.1 IF/ENDIF

Use an IF...ENDIF structure in a command file to cause a block of commands to be run when the specified conditions are met.

<pre>IF condlist THEN then-block ENDIF</pre>	<pre>IF condlist THEN then-block ELSE else-block ENDIF</pre>

Options

condlist

Lists a set of conditions that combine to form a statement that is either true or false. Conditions are combined with the connecting operators AND, OR, AND NOT, and OR NOT.

else-block

Contains one or more R:BASE commands to execute when the conditions specified in *condlist* are false.

then-block

Contains one or more R:BASE commands to execute when the conditions specified in *condlist* are true.

About the IF...ENDIF Command

When the conditions in an IF...ENDIF structure are true, R:BASE runs all the commands between THEN and ELSE, or if the ELSE option is not included, between the THEN and ENDIF.

If you use the ELSE option and the conditions are false, R:BASE runs the block of commands between the ELSE and the ENDIF. If you do not use the ELSE option and the conditions are false, R:BASE runs the command line immediately after ENDIF.

IF...ENDIF structures can be nested with other IF...ENDIF structures.

IF...ENDIF structures can be on a single line in a command file. You cannot put an IF...ENDIF structure on a single line when any of the following occur in a command file:

- The last command in the then-block is [QUIT](#).
- The structure contains an else-block.
- The structure contains a [RUN](#) command.

Using Conditions in an IF...ENDIF Structure

The conditions for an IF...ENDIF structure are listed in the table below.

Condition	Description
<i>varname</i> IS NULL	The value of the variable is null.
<i>varname</i> IS NOT NULL	The value of the variable is not null.
<i>varname</i> CONTAINS ' <i>string</i> '	The variable has a TEXT data type and contains a ' <i>string</i> ' as a substring in the variable value.
<i>varname</i> NOT CONTAINS ' <i>string</i> '	The variable has a TEXT data type and a ' <i>string</i> ' is not contained as a substring in the variable value.
<i>varname</i> LIKE ' <i>string</i> '	The variable equals a ' <i>string</i> .' A ' <i>string</i> ' can contain

	wildcards.
<i>varname</i> NOT LIKE ' <i>string</i> '	The variable does not equal the ' <i>string</i> '. A ' <i>string</i> ' can contain wildcards.
<i>varname</i> BETWEEN <i>value1</i> AND <i>value2</i>	The value of the variable is greater than or equal to <i>value1</i> and less than or equal to <i>value2</i> . The variable and the values must be the same data type.
<i>varname</i> NOT BETWEEN <i>value1</i> AND <i>value2</i>	The value of the variable is less than <i>value1</i> or greater than <i>value2</i> . The variable and the values must be the same data type.
<i>item1</i> op <i>item2</i>	<i>Item1</i> has the specified relationship to <i>item2</i> . <i>Item1</i> can be a column name, value, or expression; <i>item2</i> can be a column name, value, or expression.

The valid operators (*op*) for the conditions in an IF...ENDIF structure are listed in the table below. Do not use wildcard characters with these operators.

Operator	Description
=	Equals
<=	Less than or equal to
>=	Greater than or equal to
<	Less than
>	Greater than
<>	Not equal

An expression can be substituted for the first variable in each of the conditions. The second variable in the comparison must be dotted so that the value of the variable is used, not the variable name.

Wildcards can be used with LIKE or NOT LIKE (for example, *varname* LIKE '*string*%').

If you use multiple AND and OR operators, you must enter them in the correct order, or use parentheses to obtain the desired result. If [SET AND](#) is on (the default setting) conditions connected by AND are evaluated first; then conditions connected by OR are evaluated.

Examples

The following command lines show three nested IF...ENDIF structures.

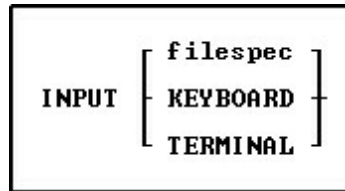
```
IF vqtyord > .vlastqty THEN
  IF vqtyord <> 0 THEN
    IF vchng > 0 AND vbackord IS NOT NULL THEN
      .
      .
      .
    ELSE
      .
      .
      .
    ENDIF
  ENDIF
ENDIF
```

The following example shows an IF...ENDIF structure on a single line.

```
IF v2 = 0 THEN ; SET VARIABLE v1 = (.v1 + 1) ; ENDIF
```

1.10.2 INPUT

Use the INPUT command to make a specified file current input device.



The INPUT command is used most often to load an ASCII file, created with the [OUTPUT](#) and [LOAD](#) commands, back into a database.

The INPUT command also transfers the contents of a backup file generated by [BACKUP](#) or [UNLOAD](#). INPUT executes the commands in the file and loads the data contained in the backup file. Do not use INPUT when the backup file is on more than one floppy disk.

filespec

Specifies an input file. If you specify a command file, it starts after you press [Enter].

KEYBOARD

Clears all nesting levels and returns control to the R> Prompt.

TERMINAL

Clears all nesting levels and returns control to the R> Prompt.

1.10.3 INSERT

Use the INSERT command to add data to a table or view without using a data-entry form.

```

INSERT INTO tblview [ <collist> ] [ VALUES <vallist>
                          SELECT clause ]
  
```

Options

(collist)

Specifies a list of one or more column names, separated by a comma (or the current delimiter). In an SQL command, any column name in the list can be preceded by a table or correlation name and a period (*tblname.colname*).

INTO tblview

Specifies the table or view name (views must be updatable).

SELECT clause

Finds values in a table, tables, or view to insert into the table or view specified by the INTO *tblview* option and the columns specified by the *collist* option.

VALUES (vallist)

Specifies a list of values to insert into the table specified by the INTO *tblview* option and the columns specified by the *collist* option. Separate values with a comma or the current delimiter.

For these data types...	Use this format for vallist
All data types except BIT, BITNOTE, LONG VARBIT, and VARBIT	'string' or value
BIT, BITNOTE, LONG VARBIT, LONG VARCHAR, VARBIT, and VARCHAR	['filename.ext'] or ['filename.ext', filetype, offset, length] Note: When you use VARCHAR, the filetype is always TXT. When you use VARBIT, BIT, and BITNOTE, filetype refers to the standard graphical file types.

About the INSERT Command

The INSERT command assigns a default value of null to any column not named in the *collist* unless a default value has been assigned to a column with the [CREATE TABLE](#) or [ALTER TABLE](#) command.

To ensure that rules are checked while adding data with the INSERT command, [SET RULES ON](#) before running the INSERT command.

The setting of the [SET ZERO](#) command affects the calculation of numeric computed columns. To have null values treated as zeros in expressions, set ZERO on. When ZERO is set off, if the value of a column used in a expression is null, the computed value will be null.

You cannot insert values into the table used in the [SELECT](#) clause.

To ensure that data is placed in the intended column, use the following guidelines:

- Do not embed commas within entries for CURRENCY, DOUBLE, INTEGER, NUMERIC, or REAL data types. R:BASE automatically inserts commas and the current currency symbol.
- When values for CURRENCY, DOUBLE, NUMERIC, or REAL or data types are decimal fractions, you must enter the decimal point. When values are whole numbers, R:BASE adds a decimal point for you at the end of the number. R:BASE adds zeros for subunits in whole currency values; For example, using the default currency format, R:BASE loads an entry of 1000 as \$1,000.00.
- When values for NOTE or TEXT data types contain commas, you can either enclose the entries within quotes, or use [SET DELIMIT](#) to change the default delimiter (comma) to another character.
- When values for NOTE or TEXT data types contain single quotes ('), and you are using the default QUOTES character ('), use two single quotes (') in the text string. For example, 'Walter Finnegan's order.'
- When a value you specify for a column is not the same data type as the column's data type, R:BASE displays an error message and you need to re-enter the entire row.
- When values for NOTE or TEXT data types exceed the maximum length of a column, R:BASE truncates the value and adds it to the table. A message is displayed that tells you which row has been truncated.

Inserting an Autonumbered Column

When you use INSERT to add a row, INSERT assigns the next available number to autonumbered columns in the table. Therefore, omit autonumbered columns and their values from a *collist*. Also, if you use the [SELECT](#) option, omit an autonumbered column from the *collist*. If a value is included for an autonumbered column that was omitted from the column list, R:BASE does not run the command because it cannot identify which column to load.

Inserting a Computed Column

Because a computed column's value is calculated, you cannot insert a new value. Omit computed columns from a *collist* or, if you are adding data to all columns, do not use a *collist* and do not specify a value for the computed column. R:BASE will skip the computed column when the row is inserted.

Examples

In the following example, the *sales* table has three columns, *col1*, *col2*, and *col3*; and *col2* is a computed column. To insert a row, you would only specify values for *col1* and *col3*. In this example, the value for *col1* is 100, and the value for *col3* is 200.

If the expression for *col2* was (*col1* + 200), then *col2* would have the value 300 when the row is inserted.

```
INSERT INTO sales VALUES (100, 200)
```

In the following example, a *vallist* adds a new row to the *product* table, filling the *model*, *prodname*, *proddesc*, and *listprice* columns.


```
INSERT INTO product (model, prodname, proddesc, listprice) +  
VALUES ('PB3060', 'Portable Advanced PC', 'System-Single +  
Drive w/Hard Disk-Portable', 3795)
```

The following command uses a *vallist* with global variables to insert the values from variables *v1*, *v2*, and *v3* into the *bonusrate* table.

```
SET VARIABLE v1 CURRENCY = 50000, v2 CURRENCY = 75000, +  
v3 REAL = .10  
INSERT INTO bonusrate VALUES (.v1, .v2, .v3)
```

The following example adds rows to *customer* table selected from *temp* table. It adds data into the *company* and *custphone* columns. The columns taken from the *temp* table can have different column names, the data types must be the same, and the order and number of columns in the column list of the source table (designated by SELECT) must match the column list of the destination table (designated by INTO).

```
INSERT INTO customer (company, custphone) +  
SELECT cname, phone FROM temp
```

1.10.4 INTERSECT

Use the INTERSECT command to form a new table by combining two tables that have one or more common columns.

```
INTERSECT tblname1 WITH tblname2 FORMING tblname3 [ USING collist ]
```

Options

FORMING tblname3

Specifies the name of the new table.

tblname1

Specifies the name of the first table to intersect.

USING collist

Specifies which columns to include in the new table. The clause must include at least one common column. You can use the clause to order the columns in the new table and to determine which columns from the original tables will be in the new table.

If you do not include the clause, R:BASE uses all columns from both tables. Columns from *tblname1* provide the order for rows and columns in the new table. R:BASE adds columns from *tblname2* that are not in *tblname1* to the right of the columns from *tblname1*.

The USING clause also determines which common columns are used for comparison. For example, if *firstname* and *lastname* are in both tables but only *lastname* is included in the USING clause, INTERSECT will compare the tables based only on *lastname*.

WITH tblname2

Specifies the name of the second table to intersect.

About the INTERSECT Command

A new table only contains the rows in which there were identical values in each of the common columns in both tables. If the common columns for a row in each table have the same value, R:BASE combines the two rows into one in the new table. If the common columns in each table have multiple matches, R:BASE adds a row for each match in the new table. For example, if *tblname1* has two rows with an identification number of 100 and *tblname2* has three rows with the same identification number, *tblname3* will have six rows, one for each possible combination.

If you combine one or more NOTE columns with the INTERSECT command, R:BASE might truncate the last NOTE column. The maximum total row length is 32,768 characters.

In the new table, R:BASE does not update the value in the row for the autonumbered column. The autonumbered column becomes a regular column.

Including Computed Columns in a New Table

R:BASE uses the current values in computed columns to determine whether the values in common computed columns are identical. To include a computed column in a new table, include each column used in the computed column's definition in the USING clause, placing the columns before the computed column. Without such a USING clause, R:BASE makes the computed column a regular column, and displays a message suggesting you rename the column in the new table. If you do not rename the column, the new table will have a column that has the same name but a different data type from the original table. If you have columns with the same name but different data types, you will not be able use the BACKUP and UNLOAD commands.

Increasing the Speed of INTERSECT

You can increase the speed of the INTERSECT command by creating an index for the common column in *tblname2*. The INTERSECT command does not use an index in *tblname1*. Below are guidelines for indexing columns in *tblname2*:

- If one table has more rows than the other table, place the larger table second (*tblname2*) in the command and create an index for its common column.
- If both tables have many rows, place the table with the shorter row length first (*tblname1*) and index the common column in the second table.
- If the tables have more than one common column, index the common column that has the most unique values in the second table (*tblname2*).

Examples

The following command forms the *models* table by combining the rows from the *component* and *compused* tables that contain identical values in a column common in both tables. Because the USING clause is not included, the *models* table contains all columns from both tables.

```
INTERSECT component WITH compused FORMING models
```

The following command lines create a table containing employee information on sales representatives and their sales bonuses. The *repsales* table only contains the rows that match in *employee* and *salesbonus* tables.

```
INTERSECT employee WITH salesbonus FORMING repsales +
  USING emplname, empid, hiredate, netamount, transdate, bonus, bonuspct
```

1.11 J

1.11.1 JOIN

Use the JOIN command to form a new table by combining rows from two existing tables based on a comparison of the columns you specify.

```
JOIN tblname1 USING colname1 WITH tblname2 USING colname2 ...
... FORMING tblname3 [ WHERE comparop ]
```

Option

FORMING tblname3

Specifies the name of the new table.

tblname1

Specifies the name of the first table to join.

USING colname1

Specifies the columns to be compared; *colname1* is a column in *tblname1*.

USING colname2

Specifies the columns to be compared; *colname2* is a column in *tblname2*.

WHERE comparop

States the condition used to compare columns. If omitted, the condition is = or EQ (equal). Use only the following comparison operators in a JOIN WHERE clause:

Comparison	Operator	Rows are included if...
=	EQ	<i>colname1</i> is equal to <i>colname2</i> (the default)
<>	NE	<i>colname1</i> is not equal to <i>colname2</i>
>	GT	<i>colname1</i> is greater than <i>colname2</i>
>=	GE	<i>colname1</i> is greater than or equal to <i>colname2</i>
<	LT	<i>colname1</i> is less than <i>colname2</i>
<=	LE	<i>colname1</i> is less than or equal to <i>colname2</i>

WITH tblname2

Specifies the name of the second table to join.

About the JOIN Command

JOIN combines all columns from two tables into a new table. R:BASE adds rows to the new table (*tblname3*) that satisfy the comparison condition (*comparop*) of a column from each of the existing tables (*tblname1* and *tblname2*). You must specify the columns to be compared: *colname1* is a column in *tblname1* and *colname2* is a column in *tblname2*.

The important characteristic of JOIN is the relationship between column values in the two tables. R:BASE compares the values of one column in the first table to the values of another column in the second table based on the operator you specify. The two tables do not need common columns to satisfy the comparison.

Column Names

If any columns in the existing tables have the same name, change the column name in one of the tables before you join them or before you use the new table. (R:BASE creates the new table and gives the two columns the same name.) If you want to use the columns with the same column name in the USING clause, change one of them before you join the tables. The column names can be different, but their data type and size must be the same.

When the comparison between the two specified columns is satisfied, R:BASE creates a row in the new table that is a combination of all columns from the row in *tblname1* and its corresponding row in *tblname2*.

NOTE Columns

If you use JOIN to combine one or more columns that have a NOTE data type, R:BASE, starting with the last column that has a NOTE data type, truncates the column until the row length is within the 32,768-character limit.

Autonumbered Columns

In the new table, R:BASE does not update the values for the autonumbered column. The autonumbered column becomes a regular column.

Improving Performance with Indexing

When a WHERE clause is used, you can speed up JOIN by creating an index for the compared column in *tblname2*. JOIN does not use an index in *tblname1*. If one table has more rows than the other table, name the larger table second in the command and index its compared column. If both tables have many rows, place the table with the shorter row length first, then create an index for the compared column in the second table.

Example

The following command creates a new table named *prodcom* by combining rows from the *prodlocation* and *component* tables. The rows included in *prodcom* are based on a comparison of the values in the *model* column in *prodlocation* and the values in the *compid* column in *component*. The WHERE clause specifies that the new table will contain only the rows in which the *model* value equals the *compid* value. Because the equal sign (=) is the default operator, you could omit the WHERE =. All the columns in *prodlocation* and *component* are included in *prodcom*.

```
JOIN prodlocation USING model WITH component USING compid +
FORMING prodcom WHERE =
```

1.12 L

1.12.1 LABEL

In a [GOTO](#) or [WHENEVER](#) statement, use the LABEL command to identify the command line to which control should be passed.

LABEL <i>lblname</i> [MODAL]

Option

lblname

Specifies a 1 to 18 character name that labels a line to skip to when a GOTO command is executed in a command or procedure file.

MODAL

Allows an EEP command block to launch a Form, Label or Report in a designer. After making appropriate changes, you will have to save the changes and close the designer in order to continue the next command in your EEP.

About the LABEL Command

After a command file is run once, R:BASE stores the labels in memory. When the command file is run again, R:BASE retrieves the location of a label when the [GOTO](#) or [WHENEVER](#) is run. However, if the GOTO or WHENEVER command is going to be run only once, place the LABEL command below the GOTO or WHENEVER command because it is more efficient for R:BASE to search downward in the command file for the matching label.

Example

In the following example, if the *vcctrvar* variable in the [IF...THEN](#) statement is equal to 999, control passes to the command lines following the LABEL command, which defines the *endproc* label. If *vcctrvar* does not equal 999, the command lines following the ELSE statement are performed.

```
IF vcctrvar = 999 THEN
  GOTO endproc
ELSE
  *( commands to execute )
ENDIF

.
.
.

LABEL endproc
  *( commands to execute )
```

1.12.2 LAUNCH

The LAUNCH command invokes/runs any associated files along with required programs.

LAUNCH filename.ext !<parameters>!W

Options

filename.ext

The name of file which has a corresponding association registered with Windows.

!<parameters>

To specify additional parameters

!W

To specify Wait Until Finished option

About the LAUNCH Command

LAUNCH command is used to execute (or LAUNCH) another application or process, from within R:BASE eXtreme for Windows and higher. It designates which file to execute. This means that the FileName can be set to any file that has a corresponding association registered with Windows. For example, setting FileName to 'README.TXT' will invoke the Windows NotePad when launched. If a full path is not specified, the current search path is used.

The LAUNCH command allows you to set the file name property to the file to be executed, and specify any command line parameters using the "Parameters". The selected program will start when the LAUNCH command is used. The launched application will start in the same directory as the executable file or in the directory referenced, if specified.

The LAUNCH command also allows you to use the "Wait Until Finished" option to specify whether or not you want R:BASE to sleep while the launched program executes or continue running. If the Wait Until Finished parameter "|W", i.e. pipe W with no space, is added to the LAUNCH command, the R:BASE will sleep until the launched process completes. If the Wait Until Finished parameter "|W" is not specified, the launched process is executed in its own thread, and the R:BASE will continue to execute.

Launch Command Features:

- No need to use the ZIP OUT command or run batch files, etc.
- No flashing screens (in and out of DOS).
- LAUNCH command is also supported in EEPs.
- The LAUNCH command will execute ANY windows program as long as it is installed and the file extension is associated in the registry.

Examples:

Example 01:

LAUNCH charts.VBS

will invoke VBScript:Windows Scripting Host Sample and will demonstrate how to access Excel using the Windows Scripting Host in an active R:BASE session.

or

LAUNCH c:\windows\samples\wsh\charts.vbs

Example 02:

LAUNCH myfile.DOC

will invoke MS Word (if installed), and let you edit the file in an active R:BASE session.

or

```
LAUNCH c:\mydocs\myfile.doc
```

Example 03:

```
LAUNCH myfile.WPD
```

will invoke Corel WordPerfect (if installed), and let you edit the file in an active R:BASE session.

or

```
LAUNCH c:\mydocs\myfile.wpd
```

Example 04:

```
LAUNCH myfile.TXT
```

will invoke MS NotePad, and let you edit the file in an active R:BASE session.

or

```
LAUNCH c:\mydocs\myfile.txt
```

Example 05:

```
LAUNCH myfile.PDF
```

will invoke Adobe Acrobat Reader, and let you view the file in an active R:BASE session.

or

```
LAUNCH c:\mydocs\myfile.pdf
```

Example 06: (To automate e-mails via R:BASE)

If you know the e-mail tags of your default e-mail program, using the LAUNCH command you could easily achieve that task of automating e-mails.

```
SET VAR vMailTo TEXT = NULL
SET VAR vCustID INTEGER = 2001
SELECT ('mailto:'+EmailAddress) INTO vMailTo INDIC IvMailTo +
FROM Contacts WHERE CustId = .vCustID
LAUNCH .vMailTo
```

That will launch default e-mail programs with filled in To:

You could follow the same scenario for other e-mail tags, such as:

Subject:

Cc:

Bcc:

Attached:

Message:

Example 07: (Simple LAUNCH command)

```
LAUNCH README.TXT
```

This will invoke the Windows NotePad with README.TXT.

Example 08: (LAUNCH command with Parameters)

```
LAUNCH C:\WordDocuments\TestFile.DOC|/MacroName
```

This will invoke the MS Word with TestFile.DOC running MacroName.

Notice the pipe "|" symbol right after TestFile.DOC to specify additional parameter.

Example 09: (LAUNCH command with Parameters and Wait Until Finished option)

```
LAUNCH C:\WordDocuments\TestFile.DOC|/MacroName|W
```

This will invoke the MS Word with TestFile.DOC using MacroName as a main window while R:BASE sleeps in the background. You will have to Exit MS Word in order to give control back to the R:BASE window/application.

Notice the additional pipe "|" symbol with "W" right after /MacroName to specify Wait Until Finish option.

Example 10: (LAUNCHing NotePad with Wait Until Wait option)

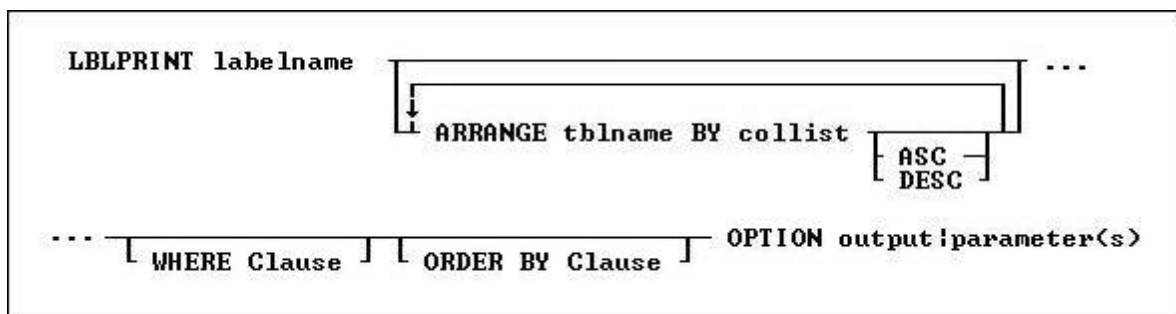
```
LAUNCH README.TXT| |W
```

This will invoke the Windows NotePad with README.TXT as a main window while R:BASE sleeps in the background. You will have to Exit NotePad in order to get control back to R:BASE the window/application.

Notice the two (2) pipes with no spaces right after README.TXT. The first pipe with no parameters and no space while the second pipe with Wait Until Finished "W" option.

1.12.3 LBLPRINT

Use the LBLPRINT command to send a set of labels to a screen, printer, file, or any combination of destinations.



Options

labelname

Specifies a label name. In R:BASE eXtreme 9.0 (32), the label name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the label name is limited to 128 characters.

ARRANGE tblname BY collist

Sorts the rows displayed by a report's lower-level tables. Each lower-level table in the report that you want to sort requires its own ARRANGE clause. Now you can specify up to twenty-five tables to ARRANGE and up to five columns in each table on which R:BASE will sort.

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

OPTION

specifies output options of the report along with additional parameters for each output such as:

- [SCREEN](#)
- [PRINTER](#)
- [BMP](#)
- [EMF](#)
- [ETXT](#) (Report Emulation Text)
- [GIF](#)
- [HTML](#)
- [JPG](#)
- [PDF](#)
- [RTF](#)
- [TIFF](#)
- [TXT](#)
- [WMF](#)
- [XHTML](#)
- [XLS](#)

1.12.3.1 SCREEN

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION SCREEN

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

WINDOW_STATE specifies the state of preview window. By default, the window state is normalized.

Values:

MAXIMIZED
MINIMIZED
NORMAL

TOOLBAR_VISIBLE specifies the option to display preview toolbar with all buttons such as, Print, Whole Page, Page Width, 100%, Zoom Level, First, Page Previous Page, Next Page, Last Page and Cancel/Close.

Values:

TRUE
FALSE



BUTTONBAR_VISIBLE specifies the option to display button bar with Whole Page, Page Width and 100%. If set to FALSE, all three buttons (Whole Page, Page Width and 100%) will not be visible.

Values:

TRUE
FALSE

BTNPRINT_VISIBLE specifies the option to display Printer button on the button toolbar.

Values:

TRUE
FALSE

BTNWHOLEPAGE_VISIBLE specifies the option to display Whole Page button on the button toolbar.

Values:

TRUE
FALSE

BTNPAGEWIDTH_VISIBLE specifies the option to display Page Width button on the button toolbar.

Values:

TRUE
FALSE

BTNP100PERCENT_VISIBLE specifies the option to display 100% button on the button toolbar.

Values:

TRUE
FALSE

ZOOMPERCENT_VISIBLE specifies the option to display Zoom Level button on the button toolbar.

Values:

TRUE
FALSE

BTNFIRST_VISIBLE specifies the option to display First Page button on the button toolbar.

Values:

TRUE
FALSE

BTNPRIOR_VISIBLE specifies the option to display Previous Page button on the button toolbar.

Values:

TRUE
FALSE

PAGENO_VISIBLE specifies the option to display Page Number button on the button toolbar.

Values:

TRUE
FALSE

BTNNEXT_VISIBLE specifies the option to display Next Page button on the button toolbar.

Values:

TRUE
FALSE

BTNLAST_VISIBLE specifies the option to display Last Page button on the button toolbar.

Values:

TRUE
FALSE

BTNCANCEL_VISIBLE specifies the option to display Cancel/Close button on the button toolbar.

Values:

TRUE
FALSE

ZOOM_TYPE specifies the zoom type, such as PAGE_WIDTH, WHOLE_PAGE, 100_PERCENT or PERCENTAGE.

Values:

PAGE_WIDTH
WHOLE_PAGE

100_PERCENT

PERCENTAGE (to be used when specifying the ZOOMPERCENT option)

ZOOMPERCENTAGE specifies the zoom level percentage, such as 70, 100, 120, 200, 250.

PREVIEW_CAPTION specifies the caption for the preview window

Note:

You must use ZOOM_TYPE value as PERCENTAGE when specifying the ZOOMPERCENTAGE value. See examples below for a more practical use of this option.

MDI specifies the option to display the preview window in a modeless window, to access other windows without closing it first.

LEFT *nnn* - specifies the left coordinate of the print preview window

TOP *nnn* - specifies the top coordinate of the print preview window

WIDTH *nnn* - specifies the width of the print preview window

HEIGHT *nnn* - specifies the height of the print preview window

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

THEMENAME - specifies one of 86 pre-defined themes. For a list, see Themes.

Examples:

Example 01 (Typical Print Preview on SCREEN with Toolbar):

LBLPRINT Invoice **WHERE** TransID = 1002 **OPTION** SCREEN

Example 02 (Typical Print Preview on SCREEN with maximized preview window):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED

Example 03 (Typical Print Preview on SCREEN with zoom type as page width):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**ZOOM_TYPE** PAGE_WIDTH

Example 04 (Typical Print Preview on SCREEN with zoom type as whole page):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**ZOOM_TYPE** WHOLE_PAGE

Example 05 (Typical Print Preview on SCREEN with zoom type as 100%):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**ZOOM_TYPE** 100_PERCENT

Example 06 (Typical Print Preview on SCREEN with zoom type as 90%):

LBLPRINT Invoice **WHERE** CustID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**ZOOM_TYPE** PERCENTAGE|**ZOOMPERCENT** 90

Example 07 (Print Preview Tool Bar without Printer Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**BTNPRINT_VISIBLE** FALSE

Example 07 (Print Preview Tool Bar without Whole Page Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**BTNWHOLEPAGE_VISIBLE** FALSE

Example 07 (Print Preview Tool Bar without Page Width Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**BTNPAGEWIDTH_VISIBLE** FALSE

Example 08 (Print Preview Tool Bar without 100% Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**BTN100PERCENT_VISIBLE** FALSE

Example 09 (Print Preview Tool Bar without Zoom Percentage Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**ZOOMPERCENT_VISIBLE** FALSE

Example 10 (Print Preview Tool Bar without First Page Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**BTNFIRST_VISIBLE** FALSE

Example 11 (Print Preview Tool Bar without Previous Page Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**BTNPRIOR_VISIBLE** FALSE

Example 12 (Print Preview Tool Bar without Page Number Icon):

LBLPRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**PAGENO_VISIBLE** FALSE

Example 13 (Print Preview Tool Bar without Next Page Icon)

```
LBLPRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN|WINDOW_STATE MAXIMIZED +
|BTNNEXT_VISIBLE FALSE
```

Example 14 (Print Preview Tool Bar without Last Page Icon)

```
LBLPRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN|WINDOW_STATE MAXIMIZED +
|BTNLAST_VISIBLE FALSE
```

Example 15 (Print Preview Tool Bar without Cancel/Close Icon)

```
LBLPRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN|WINDOW_STATE MAXIMIZED +
|BTNCANCEL_VISIBLE FALSE
```

Example 16 (Print Preview without the entire Tool Bar)

```
LBLPRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN|WINDOW_STATE MAXIMIZED +
|TOOLBAR_VISIBLE FALSE
```

Example 17 (Print Preview with a MDI window using specified coordinates)

```
LBLPRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN MDI|LEFT 20|TOP 20|WIDTH 800|HEIGHT 500
```

1.12.3.2 PRINTER

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION PRINTER

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

TRAY specifies the name of the bin (paper tray) containing the paper on which the report will be printed. A common use of this property is to set it to Manual Feed on mailing label reports. That way you can provide a means to load the special label paper without manually setting the printer.

COLLATION indicates whether multiple copies will print in pre-sorted order.

Values:

ON
OFF

COPIES specifies the number of copies to print. If the Collation property is set to ON, the copies will print in sorted order.

DOCUMENT_NAME When a report is printing, this is the name of the document as it should appear when listed in the Print Manager and on network header pages. This value can also be displayed in reports using the DocumentName type of the SystemVariable component.

DUPLEX This property indicates what type of two-sided printing should occur. Not all printers support duplex printing.

Values:

NONE - Duplex printing is disabled.

HORIZONTAL - Duplex printing should be performed across the page, from left to right, so the text will read as in a book

VERTICAL - Duplex printing should be performed down the page, from top to bottom, so the text will read as in tablet

ORIENTATION The value of the Orientation property determines whether a report prints vertically or horizontally on a page.

Values:

PORTRAIT - The print job prints vertically on the page.

LANDSCAPE - The print job prints horizontally on the page.

PAPER_NAME The name of the paper on which the report will be printed. When you specify a value for the PAPER_NAME property, the PaperWidth and PaperHeight are automatically set to the proper values. For example, if Legal is specified, the PAPER_WIDTH will be set to 8.5" and the PAPER_HEIGHT will be set to 14" when ORIENTATION is PORTRAIT (if the ORIENTATION is LANDSCAPE, the values will be reversed).

PAPER_WIDTH The value of the PAPER_WIDTH are always expressed in the units specified by the Report.

PAPER_HEIGHT The value of the PAPER_HEIGHT are always expressed in the units specified by the Report.

PRINTER_NAME The name of the printer that will be used to print the report.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages when used in combination with PAGE_LIST and PAGE_LIST_VALUES.

Values:

ALL

FIRST

LAST

PAGE_LIST - specifies the option to print a given number of pages. This option must be used with PAGE_SETTINGS and PAGE_LIST_VALUES. See examples below for more practical use of this option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with PAGE_SETTINGS and PAGE_LIST. See examples below for more practical use of this option.

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

Examples:

Example 01 (To print Customer Shipping Labels on your label printer):

```
LBLPRINT CustShippingLabel WHERE CustID = 127 OPTION PRINTER +  
| COPIES 2 +  
| PRINTER_NAME DYMO LabelWriter 2 +  
| ORIENTATION LANDSCAPE
```

1.12.3.3 BMP

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION BMP

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

- ON (default)
- OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE (default)
- YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the BMP file.

Values:

- TILE
- TOPLEFT
- TOPCENTER
- TOPRIGHT
- CENTERLEFT
- CENTER

CENTERRIGHT
 BOTTOMLEFT
 BOTTOMCENTER
 BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

MONOCHROME specifies whether bmp-file produced are color or monochrome images.

Values:

ON
 OFF

PIXELFORMAT specifies the pixel format of the bmp-file (bits/pixel).

Values:

1 : 1bit
 4 : 4bit
 8 : 8bit
 15 : 15bit
 16 : 16bit
 24 : 24bit
 32 : 32bit

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
 OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
 OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as BMP file)

```

LBLPRINT Invoice WHERE TransID = 1068 +
OPTION BMP +
| FILENAME Invoice1068.BMP +
| SHOW_CANCEL_DIALOG OFF +
| BACKGROUND_COLOR WHITE +
| MONOCHROME OFF +
| PIXELFORMAT 32 +
| SCALEX 1 +
| SCALE 1 +
| OPEN ON
RETURN

```

1.12.3.4 EMF

LBLPRINT labelname **ARRANGE** clause **WHERE** clause .. **ORDER BY** clause .. **OPTION** EMF

Note: Each additional **OPTION** parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the EMF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as EMF file)

```
LBLPRINT Invoice WHERE TransID = 1068 +  
OPTION EMF +  
|FILENAME Invoice1068.EMF +  
|SHOW_CANCEL_DIALOG OFF +  
|BACKGROUND_COLOR WHITE +  
|SCALEX 1 +  
|SCALE 1 +  
|OPEN ON
```

RETURN**1.12.3.5 ETXT**

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION ETXT

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

The **Report Emulation Text File (ETXT)** option allows you to print the exact report as plain readable text including all bands, such as Report Header, Page Header, Break Headers, Break Footers, Page Footers and Report Footers, if any.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print columnar report as Emulation Text - ETXT file)

```
LBLPRINT ColumnarReport +  
OPTION ETXT +  
|FILENAME ColumnarReport.TXT +  
|SHOW_CANCEL_DIALOG OFF +  
|OPEN ON  
RETURN
```

1.12.3.6 GIF

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION GIF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the GIF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT

CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

MONOCHROME specifies whether bmp-file produced are color or monochrome images.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the bmp-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as GIF file)

```
LBLPRINT Invoice WHERE TransID = 1068 +
OPTION GIF +
|FILENAME Invoice1068.GIF +
|SHOW_CANCEL_DIALOG OFF +
|BACKGROUND_COLOR WHITE +
|MONOCHROME OFF +
|PIXELFORMAT 32 +
|SCALEX 1 +
|SCALE 1 +
|OPEN ON
RETURN
```

1.12.3.7 HTML

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION HTML

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the HTML file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA

GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

RICHTEXT_ENCODING_TYPE specifies the way in which RichText contents should be encoded in exported HTML document when INCLUDE_RICHTEXT option is set to ON.

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object.

Values:

IMAGE
PLAINTEXT

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as dots per inch.

Values:

-1

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

INCLUDE_HYPERLINKS specifies whether hyperlinks in your report should be included in the exported file.

Values:

ON
OFF

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords Meta content.

OPTIMIZE_FOR_IE specifies whether the entire report output as HTML document should be optimized for MS Internet Explorer browser.

Values:

ON
OFF

PAGE_END_LINES indicates whether a line should be drawn at the end of each page. Setting PAGE_END_LINES to ON draws a line after each page. PAGE_END_LINES is considered only when SEP_FILE_PER_PAGE is set to OFF.

Values:

ON
OFF

OUTPUT_STYLES_TO_CSS_FILE specifies whether to output styles to a CSS (Cascade Style Sheet) file.

Values:

ON
OFF

CSSFILE specify the CSS file name if OUTPUT_STYLES_TO_CSS_FILE is set to ON.

OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE specifies whether to output scripts to a Java Script File.

Values:

ON
OFF

JAVA_SCRIPT_FILE_NAME specify the file name if the `OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE` is set to ON.

GENERATE_TOC specifies whether to generate a **T**able of **C**ontents based on all report Break Header (s), if defined.

Values:

ON
OFF

TOC_FILE_NAME specify the file name for the Table of Contents, if the `GENERATE_TOC` option is set to ON.

IMAGE_FOLDER specify the folder path and name for HTML document images.

DEFAULT_FONT_NAME specify the default font name for HTML document.

DEFAULT_FONT_SIZE specify the default font size for HTML document.

DEFAULT_FONT_COLOR specify the default font color for HTML document.

DEFAULT_FONT_BOLD specifies whether the font is bold faced.

Values:

ON
OFF

DEFAULT_FONT_ITALIC specifies whether the font is italicized.

Values:

ON
OFF

DEFAULT_FONT_UNDERLINE specifies whether the font is underlined.

Values:

ON
OFF

DEFAULT_FONT_STRIKEOUT specifies whether the font is displayed with a horizontal line through it.

Values:

ON
OFF

SEP_FILE_PER_PAGE specifies whether to create a separate file for each page of the report.

Values:

ON
OFF

NAV_SHOW Set to ON to show a collection of links that enable easy navigation between pages of the exported HTML. The links function much like the navigator buttons in the standard preview of Report Preview: one link to jump to the first page; one to the last page; one to the next page and another to get back to the previous page. `NAV_SHOW` is only effective when the option for `SEP_FILE_PER_PAGE` is set to ON.

Values:

ON
OFF

LINK_BACK_COLOR specifies the background color of navigator. This option will only be used if the `SEP_FILE_PER_PAGE` and `NAV_SHOW` options are set to ON.

LINK_H_BACK_COLOR specifies the background color of navigator when the mouse is hovered over it.

This option will only be used if the `SEP_FILE_PER_PAGE` and `NAV_SHOW` options are set to ON.

LINK_H_FORE_COLOR specifies the color of navigator when the mouse is hovered over it. This option will only be used if the `SEP_FILE_PER_PAGE` and `NAV_SHOW` options are set to ON.

USE_TEXT_LINKS specifies the type of the navigator links that should be encoded for a multi-file HTML output. Setting `USE_TEXT_LINKS` to ON uses the text provided in **LINK_TEXT_FIRST**, **LINK_TEXT_PREV**, **LINK_TEXT_NEXT** and **LINK_TEXT_LAST** as the hyperlink's caption. Setting to OFF uses Images specified by **LINK_IMG_FILE_FIRST**, **LINK_IMG_FILE_PREV**, **LINK_IMG_FILE_NEXT**, **LINK_IMG_FILE_LAST**. The property is effective only when the options for `SEP_FILE_PER_PAGE` and `NAV_SHOW` are set to ON.

Values:

ON
OFF

LINK_TEXT_FIRST specifies the hyperlink's text caption for first page when `USE_TEXT_LINKS` option is set to ON.

LINK_TEXT_PREV specifies the hyperlink's text caption for previous page when `USE_TEXT_LINKS` option is set to ON.

LINK_TEXT_NEXT specifies the hyperlink's text caption for next page when `USE_TEXT_LINKS` option is set to ON.

LINK_TEXT_LAST specifies the hyperlink's text caption for last page when `USE_TEXT_LINKS` option is set to ON.

LINK_FONT_NAME specify the font name for navigator links.

LINK_FONT_SIZE specify the font size for navigator links.

LINK_FONT_COLOR specify the font color for navigator links.

LINK_FONT_BOLD specifies whether the font is bold faced for navigator links.

Values:

ON
OFF

LINK_FONT_ITALIC specifies whether the font is italicized for navigator links.

Values:

ON
OFF

LINK_FONT_UNDERLINE specifies whether the font is underlined for navigator links.

Values:

ON
OFF

LINK_FONT_STRIKEOUT specifies whether the font for navigator link is displayed with a horizontal line through it.

Values:

ON
OFF

LINK_IMG_FILE_FIRST specifies the hyperlink's image file for first page when `USE_TEXT_LINKS` option is set to OFF.

LINK_IMG_FILE_PREV specifies the hyperlink's image file for previous page when `USE_TEXT_LINKS` option is set to OFF.

LINK_IMG_FILE_NEXT specifies the hyperlink's image file for next page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_LAST specifies the hyperlink's image file for last page when USE_TEXT_LINKS option is set to OFF.

NAV_TYPE specifies the type of the navigator. If NAV_TYPE is set to SCREEN then the navigator will be fixed to screen, else the navigator will be fixed to page. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

PAGE
SCREEN

NAV_ORIENT specifies the orientation of the navigator. if NAV_ORIENT is set to HORIZONTAL then, a Horizontal navigator is encoded, else a Vertical navigator is encoded. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

VERTICAL
HORIZONTAL

NAV_POS specifies the position of the navigator. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

DISABLE_RIGHT_CLICK specifies whether to disable the typical speed menu options when you right click on your mouse button.

Values:

ON
OFF

DISABLE_COPYING specifies whether to disable the option to highlight and copy the text.

Values:

ON
OFF

AUTO_MAXIMIZE_WINDOW specifies whether to automatically maximize the window when HTML document is opened.

Values:

ON
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0,

then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print SalesSumLab label as HTML file)

```
LBLPRINT SalesSumLab +
OPTION HTML +
|FILENAME SalesSumAndSubTot.htm +
|SHOW_CANCEL_DIALOG ON +
|BACKGROUND_FILE NONE +
|BACKGROUND_TYPE CENTER +
|BACKGROUND_COLOR WHITE +
|INCLUDE_LINES ON +
|INCLUDE_SHAPES ON +
|INCLUDE_RICHTEXT OFF +
|RICHTEXT_ENCODING_TYPE PLAINTEXT +
|INCLUDE_IMAGES ON +
|IMAGE_FORMAT JPG +
|JPEG_QUALITY 100 +
|IMAGE_DPI -1 +
|PIXELFORMAT 24 +
|INCLUDE_HYPERLINKS ON +
|TITLE Customer Sales Sub-Totals and Totals +
|SUBJECT ConComp Sales Summary +
|AUTHOR Accounting Team of ConComp +
|KEYWORDS SubTotals Totals Sales Summary +
|OPTIMIZE_FOR_IE ON +
|PAGE_END_LINES ON +
|OUTPUT_STYLES_TO_CSS_FILE ON +
|CSSFILE +
|OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE ON +
|JAVA_SCRIPT_FILE_NAME +
|GENERATE_TOC ON +
|TOC_FILE_NAME Index.htm +
|IMAGE_FOLDER Images +
|DEFAULT_FONT_NAME Verdana +
```

```

|DEFAULT_FONT_SIZE 9 +
|DEFAULT_FONT_COLOR BLACK +
|DEFAULT_FONT_BOLD OFF +
|DEFAULT_FONT_ITALIC OFF +
|DEFAULT_FONT_UNDERLINE OFF +
|DEFAULT_FONT_STRIKEOUT OFF +
|SEP_FILE_PER_PAGE ON +
|NAV_SHOW ON +
|LINK_BACK_COLOR WHITE +
|LINK_H_BACK_COLOR BLUE +
|LINK_H_FORE_COLOR WHITE +
|USE_TEXT_LINKS OFF +
|LINK_TEXT_FIRST First +
|LINK_TEXT_PREV Previous +
|LINK_TEXT_NEXT Next +
|LINK_TEXT_LAST Last +
|LINK_FONT_NAME Wingdings +
|LINK_FONT_SIZE 10 +
|LINK_FONT_COLOR BLUE +
|LINK_FONT_BOLD OFF +
|LINK_FONT_ITALIC OFF +
|LINK_FONT_UNDERLINE OFF +
|LINK_FONT_STRIKEOUT OFF +
|LINK_IMG_FILE_FIRST Images\FirstOff.gif +
|LINK_IMG_FILE_PREV Images\PrevOff.gif +
|LINK_IMG_FILE_NEXT Images\NextOff.gif +
|LINK_IMG_FILE_LAST Images\LastOff.gif +
|NAV_TYPE PAGE +
|NAV_ORIENT HORIZONTAL +
|NAV_POS BOTTOMCENTER +
|DISABLE_RIGHT_CLICK OFF +
|DISABLE_COPYING OFF +
|AUTO_MAXIMIZE_WINDOW ON +
|OPEN ON
RETURN

```

1.12.3.8 JPG

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION JPG

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the GIF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

GRAYSCALE determines whether the image output of a JPEG image is black and white, or color. Use GrayScale for reading in and writing out, that is, for decompressing and compressing, a JPEG image. This property affects how the image is displayed.

GrayScale is used for speed in output optimization. When GrayScale is set to ON, the color is separated from luminosity. Color takes the most time to decompress, so for previewing the image, GrayScale can be set to True for speed. The output then contains 255 shades of gray.

Values:

ON
OFF

PROGRESSIVE_ENCODING determines whether an image can be progressively displayed when it is decompressed.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the jpg-file (bits/pixel).

Values:

1 : 1bit
 4 : 4bit
 8 : 8bit
 15 : 15bit
 16 : 16bit
 24 : 24bit
 32 : 32bit

QUALITY determines the closeness of the saved, JPEG image to the original. As a side effect it also matter to the size of the saved file. Setting a larger value improves image quality but also increases file size; a smaller value decreases image quality and also reduces file size. Alter it according to the need.

Values:

1-100

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
 OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
 OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as JPG file)

```
LBLPRINT Invoice WHERE TransID = 1068 +
OPTION JPG +
| FILENAME Invoice1068.JPG +
| SHOW_CANCEL_DIALOG OFF +
| BACKGROUND_COLOR WHITE +
| GRAYSCALE OFF +
| PIXELFORMAT 32 +
```

```
|QUALITY 100 +
|SCALEX 1 +
|SCALE 1 +
|OPEN ON
RETURN
```

1.12.3.9 PDF

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION PDF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

```
ON (default)
OFF
```

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the PDF file.

Values:

```
TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT
```

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

```
AQUA
```

BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

RICHTEXT_ENCODING_TYPE specifies the way in which RichText contents should be encoded in exported HTML document when INCLUDE_RICHTEXT option is set to ON.

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object.

Values:

IMAGE
PLAINTEXT

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:
1-100

IMAGE_DPI specifies the depth of image as dots per inch.

Values:
-1

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:
1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

INCLUDE_HYPERLINKS specifies whether to convert all Internet addresses in your report to hyperlinks.

Values:
ON
OFF

GENERATE_TOC specifies whether to generate a **T**able **o**f **C**ontents based on all report Break Header (s), if defined.

Values:
ON
OFF

Set GENERATE_TOC to ON to generate a table of contents based on report headers, if defined, and encode it as outlines in PDF document.

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords in PDF document summary properties.

USE_COMPRESSION specifies whether compression should be used.

Values:
ON
OFF

COMPRESSION_METHOD specifies the type of compression to be used to compress text and Image.

Values:
FASTEST
NORMAL
MAXCOMPRESS

FONT_ENCODING specifies the type of font encoding to be used to encode font.

Values:
WIN_ANSI

MAC_ROMAN
MAC_EXPERT
STANDART
PDF_DOC

EMBED_USED_FONTS specifies whether all the used fonts should be embedded in the PDF document.

Values:

ON
OFF

Set EMBED_USED_FONTS to ON to embed all the fonts used in the report. Fonts are embedded only if embedding is permitted depending on the Embedding Flags of the font.

ENCRYPT_FILE specifies whether the PDF document should be encrypted.

Values:

ON
OFF (default)

Set ENCRYPT_FILE to ON to encrypt the PDF document.

USER_PASSWORD specifies the user password of the encrypted PDF document. Opening the document with the correct user password allows additional operations to be performed according to the user access permissions specified by the permissions properties such as, ENABLE_PRINTING (ON/OFF), ENABLE_COPYING (ON/OFF), and ENABLE_MODIFYING (ON/OFF). These permissions are considered by the viewer applications if the document is opened with the User password. If ENABLE_PRINTING is OFF then printing is not enabled. If ENABLE_COPYING is OFF then page contents cannot be selected and copied. If ENABLE_MODIFYING is OFF then the PDF document cannot be modified.

OWNER_PASSWORD specifies the owner password of the encrypted PDF document. Opening the document with the correct owner password (assuming it is not same as the user password) allows full (owner) access to the document. This unlimited access includes the ability to change the document's passwords and access permissions.

ENCRYPTION_LEVEL specifies the level of encryption of the PDF document.

Values:

40BIT
128BIT

Set ENCRYPTION_LEVEL to 40BIT for a 40 bit encryption of the PDF document, i.e a 40 bit key is used to encrypt the PDF document. Set ENCRYPTION_LEVEL to 128BIT for a 128 bit encryption of the PDF document, i.e a 128 bit key is used to encrypt the PDF document. ENCRYPTION_LEVEL is considered only if ENCRYPT_FILE is set to ON.

ENABLE_PRINTING specifies whether to allow printing when USER_PASSWORD is set.

Values:

ON (default)
OFF

If ENABLE_PRINTING is OFF then the printing is not enabled.

ENABLE_COPYING specifies whether to allow page contents to be selected and copied when USER_PASSWORD is set.

Values:

ON (default)
OFF

If ENABLE_COPYING is OFF then the page contents cannot be selected and copied.

ENABLE_MODIFYING specifies whether the PDF document can be modified when USER_PASSWORD is set.

Values:

ON (default)
OFF

If ENABLE_MODIFYING is OFF then the PDF document cannot be modified.

HIDE_TOOLBAR specifies whether to hide the viewer application's toolbars, such as Acrobat Reader or R:PDF Viewer, when the document is active. If the value is ON then the viewer applications toolbars are hidden.

Values:

ON
OFF (default)

HIDE_MENUBAR specifies whether to hide the viewer application's menu bar, such as Acrobat Reader or R:PDF Viewer, when the document is active. If the value is ON then the viewer applications menu bar is hidden.

Values:

ON
OFF (default)

HIDE_WINDOW_UI specifies whether to hide user interface elements in the document's window (such as scroll bars and navigation controls), leaving only the document's contents displayed. If the value is ON then documents user interface elements are hidden.

Values:

ON
OFF (default)

FIT_WINDOW specifies whether to resize the document's window to fit the size of the first displayed page. If the value is ON then the first page displayed is resized to fit the document's window.

Values:

ON
OFF (default)

CENTER_WINDOW specifies whether to position the document's window in the center of the screen. If the value is ON then the document is positioned at the center of the screen when opened.

Values:

ON
OFF (default)

PAGE_LAYOUT specifies the page layout to be used when the PDF document is opened.

Values:

SINGLE_PAGE
ONE_COLUMN
TWO_COLUMN_LEFT
TWO_COLUMN_RIGHT

If the value is set to SINGLE_PAGE then one page is displayed at a time. If the value is set to ONE_COLUMN then pages are displayed in one column. If the value is set to TWO_COLUMN_LEFT then pages are displayed in two columns with odd numbered pages on the left. If the value is set to TWO_COLUMN_RIGHT then pages are displayed in two columns with odd numbered pages on the right.

PAGE_MODE specifies how the document should be displayed when opened.

Values:

ONE
OUTLINES
THUMBS
FULLSCREEN

If the value is set to ONE then neither document outlines nor thumbnail images are visible. If the value is set to OUTLINES then document outline is visible. If the value is set to THUMBS then thumbnail images are visible. If the value is set to FULLSCREEN then document is opened in Full Screen mode with no menu bar, window controls, or any other window visible.

NON_FULL_SCREEN_PAGE_MODE specifies how to display the document on exiting full screen mode. This is meaningful only if the PAGE_MODE value is set to FULLSCREEN.

Values:

NONE
USE_OUTLINES
USE_THUMB

If the value is set to NONE then neither document outlines nor thumbnail images are visible. If the value is set to USE_OUTLINES then document outline is visible. If the value is set to USE_THUMB then thumbnail images are visible.

PRESENTATION_MODE similar to transition effects specifies how the viewer application, such as Acrobat Reader or R:PDF Viewer, should display/present the document. Some viewer applications allow a document to be displayed in the form of a presentation or "slide show," advancing from one page to the next either automatically or under user control.

Values:

SPLIT_HOR_INWARD
SPLIT_HOR_OUTWARD
SPLIT_VERT_INWARD
SPLIT_VERT_OUTWARD
BLINDS_HOR
BLINDS_VERT
BOX_INWARD
BOX_OUTWARD
WIPE_LEFT_TO_RIGHT
WIPE_BOTTOM_TO_TOP
WIPE_RIGHT_TO_LEFT
WIPE_TOP_TO_BOTTOM
DISSOLVE
GLITTER_LEFT_TO_RIGHT
GLITTER_TOP_TO_BOTTOM
GLITTER_TOPLEFT_TO_BOTTOMRIGHT
NONE

SPLIT_HOR_INWARD: Two lines sweep across the screen, revealing the new page. The lines are horizontal and move inward from the edges of the page.

SPLIT_HOR_OUTWARD: Two lines sweep across the screen, revealing the new page. The lines are horizontal and move outward from the center of the page.

SPLIT_VERT_INWARD: Two lines sweep across the screen, revealing the new page. The lines are vertical and move inward from the edges of the page.

SPLIT_VERT_OUTWARD: Two lines sweep across the screen, revealing the new page. The lines are vertical and move outward from the center of the page.

BLINDS_HOR: Multiple lines, evenly spaced across the screen, synchronously sweep in the same direction to reveal the new page. The lines are horizontal and move downward.

BLINDS_VERT: Multiple lines, evenly spaced across the screen, synchronously sweep in the same direction to reveal the new page. The lines are vertical and move to the right.

BOX_INWARD: A rectangular box sweeps inward from the edges of the page revealing the new page.

BOX_OUTWARD: A rectangular box sweeps outward from the center of the page revealing the new page.

WIPE_LEFT_TO_RIGHT: A single line sweeps across the screen from Left to Right edge of the page

WIPE_BOTTOM_TO_TOP: A single line sweeps across the screen from Bottom to the Top edge of the page

WIPE_RIGHT_TO_LEFT: A single line sweeps across the screen from Right to the Left edge of the page

WIPE_TOP_TO_BOTTOM: A single line sweeps across the screen from Top to Bottom edge of the page

DISSOLVE: The old page "dissolves" gradually to reveal the new one.

GLITTER_LEFT_TO_RIGHT: Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from Left side of the screen to the Right side.

GLITTER_TOP_TO_BOTTOM: Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from Top side of the screen to the Bottom side.

GLITTER_TOPLEFT_TO_BOTTOMRIGHT: Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from TopLeft corner of the screen to the BottomRight corner.

NONE: The new page simply replaces the old one with no special transition effect.

DURATION specifies the duration of the presentation mode (transition effect) in seconds.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print SalesSum as secure PDF document)

LBLPRINT SalesSum +

OPTION PDF +

| **FILENAME** SalesSumAndSubTotals.PDF +

| **SHOW_CANCEL_DIALOG** ON +

| **BACKGROUND_FILE** NONE +

| **BACKGROUND_TYPE** CENTER +

| **BACKGROUND_COLOR** WHITE +

| **INCLUDE_LINES** ON +

| **INCLUDE_SHAPES** ON +

| **INCLUDE_RICHTEXT** ON +

| **RICHTEXT_ENCODING_TYPE** PLAINTEXT +

```

| INCLUDE_IMAGES ON +
| IMAGE_FORMAT JPG +
| PIXELFORMAT 32 +
| JPEG_QUALITY 100 +
| IMAGE_DPI -1 +
| INCLUDE_HYPERLINKS ON +
| GENERATE_TOC ON +
| TITLE Customer Sales Sub-Totals and Totals +
| SUBJECT Sales Summary +
| AUTHOR Accounting Team of Consolidated Computer Company +
| KEYWORDS SubTotals Totals Sales Summary +
| USE_COMPRESSION ON +
| COMPRESSION_METHOD MAXCOMPRESS +
| FONT_ENCODING WIN_ANSI +
| EMBED_USED_FONTS ON +
| ENCRYPT_FILE ON +
| USER_PASSWORD RRBYW10 +
| OWNER_PASSWORD RRBYW10Owner +
| ENCRYPTION_LEVEL 128BIT +
| ENABLE_PRINTING OFF +
| ENABLE_COPYING OFF +
| ENABLE_MODIFYING OFF +
| HIDE_TOOLBAR OFF +
| HIDE_MENUBAR OFF +
| HIDE_WINDOW_UI OFF +
| FIT_WINDOW OFF +
| CENTER_WINDOW OFF +
| PAGE_LAYOUT SINGLE_PAGE +
| PAGE_MODE NONE +
| NON_FULL_SCREEN_PAGE_MODE NONE +
| PRESENTATION_MODE NONE +
| DURATION 0 +
| OPEN ON
RETURN

```

1.12.3.10 RTF

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION RTF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of

report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the RTF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON

OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as dots per inch.

Values:

-1

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

GRAPHIC_DATA_IN_BINARY specifies whether graphic data is to be encoded in binary format.

Values:

ON (default)
OFF

Set to ON to encode graphic data in binary format. If OFF, graphic data is encoded in hexadecimal format.

OPTIMIZATION specifies the type of encoding of the RTF document.

Values:

POSITIONAL_ACCURACY (default)
EASE_OF_EDITING

Resulting RTF document can be either optimized for positional accuracy (meaning that the placement of the objects will be as close as possible to a printed report), or for ease of editing (meaning that everything is treated as simple text and it is much easier to edit the file at the expense of aesthetics).

Set OPTIMIZATION EASE_OF_EDITING if you wish to easily modify the resulting RTF document.

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords Meta content in RTF document property summary.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print invoice as RTF document)

```
LBLPRINT Invoice WHERE TransID = 1068 +  
OPTION RTF +  
|FILENAME Invoice1068.RTF +  
|SHOW_CANCEL_DIALOG ON +  
|BACKGROUND_FILE NONE +  
|BACKGROUND_TYPE CENTER +  
|INCLUDE_LINES ON +  
|INCLUDE_SHAPES ON +  
|INCLUDE_RICHTEXT ON +  
|INCLUDE_IMAGES ON +  
|IMAGE_FORMAT JPG +  
|JPEG_QUALITY 100 +  
|IMAGE_DPI -1 +  
|PIXELFORMAT 32 +  
|GRAPHIC_DATA_IN_BINARY ON +  
|OPTIMIZATION POSITIONAL_ACCURACY +  
|TITLE Customer Invoice +  
|SUBJECT Sales Order +  
|AUTHOR Accounting Team of Consolidated Computer Company +  
|KEYWORDS Invoice +  
|OPEN ON  
RETURN
```

1.12.3.11 TIFF

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION TIFF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the BMP file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

MONOCHROME specifies whether tiff-file produced are color or monochrome images.

Values:

ON
OFF

MULTIPAGE specifies the output report to one multi-page file or one file per page.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the tiff-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

COMPRESSION When an image is loaded, this parameter indicates the compression used by the image. When saving in image, the parameter may be set to specify which compression method to use.

Values:

NONE
PACKBITS
GROUP3_1D
GROUP3_2D
GROUP4
ZLIB
JPEG

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as TIF file)

```
LBLPRINT Invoice WHERE TransID = 1068 +
OPTION TIFF +
|FILENAME Invoice1068.TIF +
|SHOW_CANCEL_DIALOG OFF +
|BACKGROUND_COLOR WHITE +
|MONOCHROME OFF +
|MUTIPAGE OFF +
|PIXELFORMAT 32 +
|SCALEX 1 +
|SCALE 1 +
|COMPRESSION NONE +
|OPEN ON
RETURN
```

1.12.3.12 TXT

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION TXT

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

The TXT "Text File" option allows you to define a particular band to print the data as Comma Delimited, Tab Delimited or Fixed Length file. This is the perfect option to export selective and/or summarized data used in a report band.

Caution:

This option requires the report setup to define the exact band to be print as text file.

Label Designer | File | Print to Text File Setup ...

If a user attempts to print the label as Text File (OPTION TXT) without first having defined the Text File Setup in Report Designer, the program will return an -ERROR- message and not just write a zero length file.

The same rule applies when printing the report as Text File using the Print to a File option from the Database Explorer.

Database Explorer | Labels | Run Label | Print Target .. File

For step-by-step instructions on how to set up a report to print as Text File, please refer to the article on From The Edge: Printing Reports as Text File (May 6, 2005).

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print customer data as a text file)

```
LBLPRINT Custdata +  
OPTION TXT +  
| FILENAME Custdata.TXT +  
| SHOW_CANCEL_DIALOG OFF +  
| OPEN ON  
RETURN
```

1.12.3.13 WMF

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION WMF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the WMF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as WMF file)

```
LBLPRINT Invoice WHERE TransID = 1068 +
OPTION WMF +
|FILENAME Invoice1068.WMF +
|SHOW_CANCEL_DIALOG OFF +
|BACKGROUND_COLOR WHITE +
|SCALEX 1 +
|SCALE 1 +
|OPEN ON
RETURN
```

1.12.3.14 XHTML

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION XHTML

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All

report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the XHTML file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

RICHTEXT_ENCODING_TYPE specifies the way in which RichText contents should be encoded in exported HTML document when INCLUDE_RICHTEXT option is set to ON.

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object.

Values:

IMAGE
PLAINTEXT

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as dots per inch.

Values:

-1

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

INCLUDE_HYPERLINKS specifies whether hyperlinks in your report should be included in the exported file.

Values:

ON
OFF

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords Meta content.

PAGE_END_LINES indicates whether a line should be drawn at the end of each page. Setting **PAGE_END_LINES** to ON draws a line after each page. **PAGE_END_LINES** is considered only when **SEP_FILE_PER_PAGE** is set to OFF.

Values:

ON
OFF

OUTPUT_STYLES_TO_CSS_FILE specifies whether to output styles to a CSS (Cascade Style Sheet) file.

Values:

ON
OFF

CSSFILE specify the CSS file name if **OUTPUT_STYLES_TO_CSS_FILE** is set to ON.

OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE specifies whether to output scripts to a Java Script File.

Values:

ON
OFF

JAVA_SCRIPT_FILE_NAME specify the file name if the **OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE** is set to ON.

GENERATE_TOC specifies whether to generate a **Table of Contents** based on all report Break Header (s), if defined.

Values:

ON
OFF

TOC_FILE_NAME specify the file name for the Table of Contents, if the **GENERATE_TOC** option is set to ON.

IMAGE_FOLDER specify the folder path and name for XHTML document images.

DEFAULT_FONT_NAME specify the default font name for XHTML document.

DEFAULT_FONT_SIZE specify the default font size for XHTML document.

DEFAULT_FONT_COLOR specify the default font color for XHTML document.

DEFAULT_FONT_BOLD specifies whether the font is bold faced.

Values:

ON
OFF

DEFAULT_FONT_ITALIC specifies whether the font is italicized.

Values:

ON
OFF

DEFAULT_FONT_UNDERLINE specifies whether the font is underlined.

Values:

ON
OFF

DEFAULT_FONT_STRIKEOUT specifies whether the font is displayed with a horizontal line through it.

Values:

ON
OFF

SEP_FILE_PER_PAGE specifies whether to create a separate file for each page of the report.

Values:

ON
OFF

NAV_SHOW Set to ON to show a collection of links that enable easy navigation between pages of the exported XHTML. The links function much like the navigator buttons in the standard preview of Report Preview: one link to jump to the first page; one to the last page; one to the next page and another to get back to the previous page. NAV_SHOW is only effective when the option for SEP_FILE_PER_PAGE is set to ON.

Values:

ON
OFF

LINK_BACK_COLOR specifies the background color of navigator. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

LINK_H_BACK_COLOR specifies the background color of navigator when the mouse is hovered over it. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

LINK_H_FORE_COLOR specifies the color of navigator when the mouse is hovered over it. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

USE_TEXT_LINKS specifies the type of the navigator links that should be encoded for a multi-file HTML output. Setting USE_TEXT_LINKS to ON uses the text provided in **LINK_TEXT_FIRST**, **LINK_TEXT_PREV**, **LINK_TEXT_NEXT** and **LINK_TEXT_LAST** as the hyperlink's caption. Setting to OFF uses Images specified by **LINK_IMG_FILE_FIRST**, **LINK_IMG_FILE_PREV**, **LINK_IMG_FILE_NEXT**, **LINK_IMG_FILE_LAST**. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

ON
OFF

LINK_TEXT_FIRST specifies the hyperlink's text caption for first page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_PREV specifies the hyperlink's text caption for previous page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_NEXT specifies the hyperlink's text caption for next page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_LAST specifies the hyperlink's text caption for last page when USE_TEXT_LINKS option is set to ON.

LINK_FONT_NAME specify the font name for navigator links.

LINK_FONT_SIZE specify the font size for navigator links.

LINK_FONT_COLOR specifies the font color for navigator links.

LINK_FONT_BOLD specifies whether the font is bold faced for navigator links.

Values:

ON
OFF

LINK_FONT_ITALIC specifies whether the font is italicized for navigator links.

Values:

ON
OFF

LINK_FONT_UNDERLINE specifies whether the font is underlined for navigator links.

Values:

ON
OFF

LINK_FONT_STRIKEOUT specifies whether the font for navigator link is displayed with a horizontal line through it.

Values:

ON
OFF

LINK_IMG_FILE_FIRST specifies the hyperlink's image file for first page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_PREV specifies the hyperlink's image file for previous page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_NEXT specifies the hyperlink's image file for next page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_LAST specifies the hyperlink's image file for last page when USE_TEXT_LINKS option is set to OFF.

NAV_TYPE specifies the type of the navigator. If NAV_TYPE is set to SCREEN then the navigator will be fixed to screen, else the navigator will be fixed to page. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

PAGE
SCREEN

NAV_ORIENT specifies the orientation of the navigator. if NAV_ORIENT is set to HORIZONTAL then, a Horizontal navigator is encoded, else a Vertical navigator is encoded. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

VERTICAL
HORIZONTAL

NAV_POS specifies the position of the navigator. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT

BOTTOMCENTER
BOTTOMRIGHT

DISABLE_RIGHT_CLICK specifies whether to disable the typical speed menu options when you right click on your mouse button.

Values:

ON
OFF

DISABLE_COPYING specifies whether to disable the option to highlight and copy the text.

Values:

ON
OFF

AUTO_MAXIMIZE_WINDOW specifies whether to automatically maximize the window when HTML document is opened.

Values:

ON
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print SalesSum as XHTML file)

```
LBLPRINT SalesSum +
OPTION XHTML +
|FILENAME SalesSumAndSubTot.htm +
|SHOW_CANCEL_DIALOG ON +
|BACKGROUND_FILE NONE +
|BACKGROUND_TYPE CENTER +
|BACKGROUND_COLOR WHITE +
|INCLUDE_LINES ON +
|INCLUDE_SHAPES ON +
|INCLUDE_RICHTEXT OFF +
|RICHTEXT_ENCODING_TYPE PLAINTEXT +
|INCLUDE_IMAGES ON +
|IMAGE_FORMAT JPG +
|JPEG_QUALITY 100 +
|IMAGE_DPI -1 +
|PIXELFORMAT 24 +
```

```
|INCLUDE_HYPERLINKS ON +
|TITLE Customer Sales Sub-Totals and Totals +
|SUBJECT ConComp Sales Summary +
|AUTHOR Accounting Team of ConComp +
|KEYWORDS SubTotals Totals Sales Summary +
|PAGE_END_LINES ON +
|OUTPUT_STYLES_TO_CSS_FILE ON +
|CSSFILE +
|OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE ON +
|JAVA_SCRIPT_FILE_NAME +
|GENERATE_TOC ON +
|TOC_FILE_NAME Index.htm +
|IMAGE_FOLDER Images +
|DEFAULT_FONT_NAME Verdana +
|DEFAULT_FONT_SIZE 9 +
|DEFAULT_FONT_COLOR BLACK +
|DEFAULT_FONT_BOLD OFF +
|DEFAULT_FONT_ITALIC OFF +
|DEFAULT_FONT_UNDERLINE OFF +
|DEFAULT_FONT_STRIKEOUT OFF +
|SEP_FILE_PER_PAGE ON +
|NAV_SHOW ON +
|LINK_BACK_COLOR WHITE +
|LINK_H_BACK_COLOR BLUE +
|LINK_H_FORE_COLOR WHITE +
|USE_TEXT_LINKS OFF +
|LINK_TEXT_FIRST First +
|LINK_TEXT_PREV Previous +
|LINK_TEXT_NEXT Next +
|LINK_TEXT_LAST Last +
|LINK_FONT_NAME Wingdings +
|LINK_FONT_SIZE 10 +
|LINK_FONT_COLOR BLUE +
|LINK_FONT_BOLD OFF +
|LINK_FONT_ITALIC OFF +
|LINK_FONT_UNDERLINE OFF +
|LINK_FONT_STRIKEOUT OFF +
|LINK_IMG_FILE_FIRST Images\FirstOff.gif +
|LINK_IMG_FILE_PREV Images\PrevOff.gif +
|LINK_IMG_FILE_NEXT Images\NextOff.gif +
|LINK_IMG_FILE_LAST Images\LastOff.gif +
|NAV_TYPE PAGE +
|NAV_ORIENT HORIZONTAL +
|NAV_POS BOTTOMCENTER +
|DISABLE_RIGHT_CLICK OFF +
|DISABLE_COPYING OFF +
|AUTO_MAXIMIZE_WINDOW ON +
|OPEN ON
RETURN
```

1.12.3.15 XLS

LBLPRINT labelname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION XLS

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

IMPORTANT: When printing to the XLS format, the objects must be spaced apart appropriately or "padded" in order for the report data to appear correctly in the Excel output. Experimenting with vertical and horizontal will also allow you to attain your desired output. Also, please refrain from using elaborate fonts that may not be supported within Excel.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

ALL_TEXT_IN_GENERAL_FORMAT Specifies whether all the text in the report should be converted to general format or the format of the text should be detected and encoded accordingly.

Vales:

ON
OFF

Set ALL_TEXT_IN_GENERAL_FORMAT to ON to encode all the text contents in the report in general format. General format text have no specific number format. Set ALL_TEXT_IN_GENERAL_FORMAT to OFF to detect the format of the text and encode accordingly. In this case number and other formats are identified and encoded.

CELL_ATTR Cell attributes include shading of the cell whose color is not WHITE.

Values:

ON
OFF

INCLUDE_IMAGES specifies whether Image objects on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report.

Values:

BMP
WMF

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF (default)

INCLUDE_RICH_TEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON (default)
OFF

RICHTEXT_ENCODING_TYPE indicates the way in which RichText contents should be encoded in Excel document.

Values:

IMAGE (default)
PLAINTEXT

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object. When RICHTEXT_ENCODING_TYPE is set to PLAINTEXT make sure you set the Default Font property of the RichText control in report designer to true and set the required font in the Font property.

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF (default)

LINE_SPACE specifies the spacing to be used between lines of text.

Values:

ACTUAL
0
1
2
3
4
5

ONE_SHEET_PER_PAGE specifies whether each page in the report should be encoded into separate excel sheets or all the pages should be encoded in a single excel sheet.

Values:

ON
OFF (default)

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

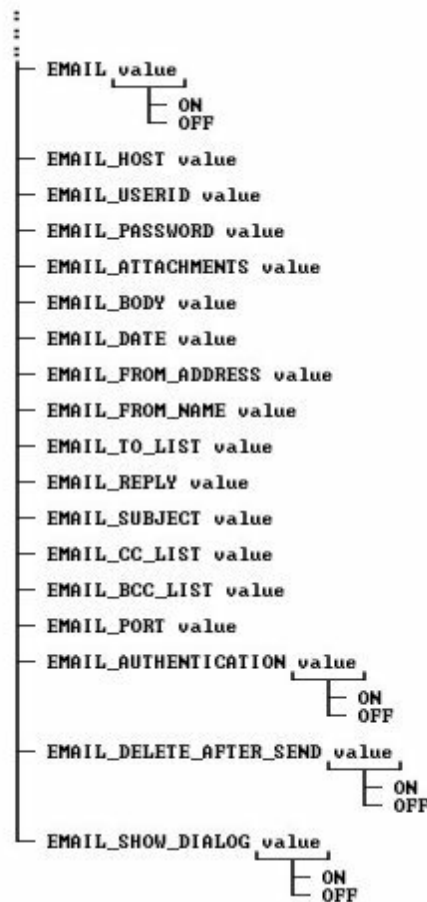
Example:

Example 01 (To print Invoices as XLS file)

```
LBLPRINT Invoice WHERE CustID = 127 +  
OPTION XLS +  
| FILENAME Invoice.XLS +  
| SHOW_CANCEL_DIALOG OFF +  
| ALL_TEXT_IN_GENERAL_FORMAT OFF +  
| CELL_ATTR ON +  
| INCLUDE_IMAGES OFF +  
| INCLUDE_LINES OFF +  
| INCLUDE_RICH_TEXT ON +  
| RICHTEXT_ENCODING_TYPE IMAGE +  
| INCLUDE_SHAPES OFF +  
| LINE_SPACE ACTUAL +  
| ONE_SHEET_PER_PAGE ON +  
| OPEN ON  
RETURN
```

1.12.3.16 EMAIL SET ON

Additional options when the EMAIL value is set to **ON**:



Note: Each additional OPTION parameter must be separated by pipe | symbol

OPTION Parameters:

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

EMAIL_HOST specifies Hostname/IP Address of SMTP Server (Outgoing). Also sometimes known as your 'Sending Mail Server', your SMTP server receives and processes all of your outgoing mail. For example, your Outgoing Mail Server might be called something like 'my-internet-provider.com' or 'outgoing-mail.mycompany.com'. The SMTP server name is sometimes the same as your incoming mail server name. If your ISP or your network administrator has not given this server name to you, you can leave the box blank. If you do not know what your outgoing mail server name is, you need to contact your Internet Service Provider or Network Administrator.

EMAIL_USERID specifies a valid User Name or Login Name for SMTP Server. This is the name you use to log into your email account. Your ISP or organization provides it. It is generally, but not necessarily, the name that appears before the at sign (@) in your return email address. For example, if your email address was <jim@my-server.com> - then your username would be 'jim' (no quotes). If you do not know what your username is, you need to contact your Internet Service Provider or Network

Administrator.

EMAIL_PASSWORD specifies valid SMTP Server Password for the user.

EMAIL_ATTACHMENTS specifies file list. Each file must be separated by # symbol.

EMAIL_BODY This option contains body of the mail.

EMAIL_DATE This option holds the date the message was sent. (Default = .#DATE)

EMAIL_FROM_ADDRESS specifies the address of the E-mail account from which the mail is to be sent. Sender E-Mail Address (From E-Mail Address:) Enter your e-mail address here. For example, your email address might be 'jim@my-internet-service.com' or 'molly@my-business-domain.com' Take care not to make a mistake here - or else it will make it difficult for others to reply to your emails. If you do not know what your e-mail address is, you need to contact your Internet Service Provider or Network Administrator.

EMAIL_FROM_NAME represents the person's name in an E-mail address. Your real name or what you want to appear as your name on messages that you send out.

EMAIL_TO_LIST identifies the recipient(s) of a message. Specify the list of valid e-mail address(s) to which the e-mail is to be sent.

EMAIL_REPLY identifies the delivery address for message replies. This is typically the same as Sender E-Mail Address. Enter your e-mail address here. For example, your email address might be 'jim@my-internet-service.com' or 'molly@my-business-domain.com' Take care not to make a mistake here - or else it will make it difficult for others to reply to your emails. If you do not know what your e-mail address is, you need to contact your Internet Service Provider or Network Administrator.

EMAIL_SUBJECT identifies the subject for the message. Subject is a text string that identifies the subject for the message.

EMAIL_CC_LIST specied the list of e-mail addresses as Carbon Copies to be sent along with the e-mail.

EMAIL_BCC_LIST specifies the list of e-mail addresses as Blind Carbon Copies to be sent along with the mail.

EMAIL_PORT specifies the Integer value of SMTP Server Port. (Default = 25).

EMAIL_AUTHENTICATION Set the value for this option to ON if your SMTP Server requires user authentication.

Values:

ON
OFF

EMAIL_DELETE_AFTER_SEND Set the value for this option to ON if you wish to delete the report files after sending e-mail.

EMAIL_SHOW_DIALOG Set the value for this option to ON if you would like to show e-mail setup dialog.

Example:

```
CONNECT RRBYW10
LBLPRINT Invoice WHERE TransID = 1031 +
OPTION PDF|FILENAME Invoice_1031.PDF +
|EMAIL ON +
|EMAIL_TO_LIST yourname@yourdomain.com, whoelse@yourdomain.com +
|EMAIL_FROM_NAME John D. Johnson +
|EMAIL_FROM_ADDRESS johndjohnson@yourdomain.com +
```

```

|EMAIL_SUBJECT Imagine The Possibilities! +
|EMAIL_BODY Attached is the invoice - sent via R:BASE. Because we Can! +
|EMAIL_ATTACHMENTS C:\Letters\otherstuff.pdf#C:\Letters\morestuff.pdf +
|EMAIL_DELETE_AFTER_SEND ON +
|EMAIL_SHOW_DIALOG OFF
RETURN

```

1.12.4 LIST

Use the LIST command to display or print information about a database.

LIST	
ALL	
CASCADE	
COLUMNS	
COLUMN	[colname]
CONSTRAINTS	[FOR tblname]
CURSORS	
DATABASES	
FKEYS	[FOR tblname]
FORMS	[FOR tblname]
FUNCTIONS	
INDEXES	[indexname] [FOR tblname]
LABELS	[FOR tblname]
PKEYS	[FOR tblname]
PROCEDURE	[procname]
REPORTS	[FOR tblname]
RULES	
SYS_%	
TABLES	[SYS_%] [tblname]
TRIGGERS	[FOR tblname]
UKEYS	[FOR tblname]
VIEWS	[viewname]

Options

ALL

Displays by table the column numbers, column names, data types, sizes, indexes, autonumbering status, expressions for computed columns, default values, column descriptions, table description, and row count for each table in the open database. A blank line will be inserted between each table.

CASCADE

Displays a list of tables with the CASCADE flag, and the type of flag; UPDATE, DELETE, or BOTH.

COLUMNS

Displays the name, data type, size, table name, index status, autonumbering status, expression, any default value, and description for each column in the open database.

If you specify part of a column name followed by a wildcard character, R:BASE will only show the column names that match that pattern.

COLUMN colname

Displays the name, data type, size, table name, index status, autonumbering status, expression, any default value, and description for the column that you enter in every table.

CONSTRAINTS

Displays the name, type, table name, and referenced table for each constraint in the database.

CONSTRAINTS FOR tblname

Specifies a table name for which to list constraints.

CURSORS

Specifies the status of cursors and cursor definitions.

DATABASES

Lists all databases in the current directory.

FKEYS

Displays the key ID number, the type of key (foreign), the table name the key is located in, and the table that the foreign key references.

FKEYS FOR tblname

Displays the key ID number, the type of key (foreign), the column name of the foreign key in the specified table, the table name that the foreign key references, and the column name that the foreign key references.

FORMS

Displays the name, driving table or view, and description for every form in the open database.

FORMS FOR tblname

Displays the name, driving table or view, and description for all forms defined for *tblname*.

FUNCTIONS

Displays all defined DLL functions.

INDEXES

Displays the name, table name, and column names for all indexes in the open database.

INDEXES FOR tblname

Displays the name, table name, column names, data types, orders, and sizes for all indexes defined for *tblname*.

INDEXES indexname

Displays the name, table name, column names, data types, orders, and sizes for *indexname*.

LABELS

Displays the name, driving table or view, and description for every label in the open database.

LABELS FOR tblname

Displays the name, driving table or view, and description for all labels defined for *tblname*.

PKEYS

Displays the key ID number, the type of key (primary) and if it is referenced in another table, the table name the key is located in.

PKEYS FOR tblname

Displays the key ID number, the type of key (primary), the column name of the primary key in the specified table, the table name that references the primary key, and the column name that references the primary key.

PROCEDURE

Displays the name and description for every procedure in the open database.

PROCEDURE procname

Displays the name, description, ID, date last modified, version, locked by (if locked) and return type for the specified procedure. Also, if the procedure was stored with parameters, the number of parameters, names and descriptions will be displayed.

REPORTS

Displays the name, driving table or view, and description for every report in the open database.

REPORTS FOR tblname

Displays the name, driving table or view, and description for all reports defined for *tblname*.

RULES

Displays whether rules checking is set on or off, the table name, whether the conditions in the WHERE clause must be met, the rule message, and the WHERE clause for every rule in the open database.

SYS_%

Displays a summary of system tables and views.

TABLES

Displays the database name, table name, number of columns, number of rows for each table in the open database, and lists all views, attached dBASE files, and attached foreign server tables. In addition, the LIST TABLES command shows the number of tables, columns, and indexes in the database. These numbers all include system tables. Temporary tables in a read-only schema mode are displayed dimmed. (For more information about the read-only scheme mode, see [SET STATICDB](#).) When you enter the LIST command by itself, R:BASE executes the LIST TABLES command.

TABLES SYS_%

Displays a detailed list of system tables and views.

TABLES tblname

Displays the name and description of the table, table number, name, data type, size, index status, autonumbering status, expression, default value, description, constraints, and index types for each column in the table. Columns are displayed in the order they were defined. The same output is displayed if you enter LIST *tblname* as a command.

TRIGGERS

Displays all of the tables that have triggers and their triggers in the open database.

TRIGGERS FOR tblname

Displays the triggers for the specified table.

UKEYS

Displays the key ID number, the type of key (unique), and the table name the key is located in.

UKEYS FOR tblname

Displays the key ID number, the type of key (unique), and the column name the unique key is assigned to.

VIEWS

Displays the database name and the names of all the views in the open database. Temporary views in read-only schema mode are dimmed.

VIEWS viewname

Displays the name and the [SELECT](#) clause defined for the view.

About the LIST Command

If the LIST command is entered without any options, R:BASE executes the LIST TABLES command. Column numbers that are displayed with the LIST command can be entered in other commands instead of column names. The total number of tables and columns listed includes R:BASE system tables (of which some are hidden) and their associated columns.

When a table has a LOCK on it, the LIST command will display "(L)" in front of the table name.

When a table is a TEMPORARY table, the LIST command will display "(T)" in front of the table name.

Database Access Rights with LIST

Before using the LIST command, enter the correct user identifier with the [CONNECT](#) or [SET USER](#) command.

When access rights have been assigned to a user with the GRANT command, R:BASE displays only those tables, columns, and views for which the user has been granted access.

In a multi-user environment, locked tables are displayed in reverse video when the LIST command is run. When you run the LIST TABLE *tblname* command, R:BASE tells you if the lock is an edit, cursor, local, or remote lock.

Example

Column numbers that are displayed with the LIST command can be entered in other commands instead of column names. The following command would display the first column, *custid*, in the *customer* table.

```
SELECT #1 FROM customer
```

1.12.5 LIST ACCESS

Use the LIST ACCESS command to allow the owner of a database to view all user's access rights assigned by the [GRANT](#) command, and to allow users to view their access rights and rights granted to PUBLIC.

LIST ACCESS		FOR user	ON tblview
ALL	_____		
ALTER	_____		
CREATE	_____		
DELETE	_____		
INSERT	_____		
SELECT	_____		
UPDATE	_____		
REFERENCES	_____		

Options

ALL

Displays the users and tables to which the ALL PRIVILEGES access right has been assigned.

ALTER

Displays the users and tables to which the ALTER access right has been assigned.

CREATE

Displays the users and tables to which the CREATE access right has been assigned.

DELETE

Displays the users and tables to which the DELETE access right has been assigned.

FOR user

Displays the tables and access rights for a specific user.

INSERT

Displays the users and tables to which the INSERT access right has been assigned.

ON tblview

Displays the users and their access rights for a specific table or view.

REFERENCES

Displays the users who have rights to reference a primary-key or unique-key table.

SELECT

Displays the users and tables to which the SELECT access right has been assigned.

UPDATE

Displays the users, tables, and columns to which the UPDATE access right has been assigned.

About the LIST ACCESS Command

The LIST ACCESS command is available when access rights have been granted. When access rights have been granted with the WITH GRANT OPTION, the LIST ACCESS command displays an asterisk in front of the access right. When the LIST ACCESS command is entered without any options, it displays all access rights that have been granted on all tables to which the current user identifier has access.

The LIST ACCESS command requires a user identifier-either the database owner's, user's, or NONE. When you enter the owner's user identifier, the LIST ACCESS command allows you to view all current access rights assigned with the [GRANT](#) command. When you enter your user identifier, the LIST ACCESS command displays only the access rights granted to PUBLIC and to you. Users other than the database owner who are assigned access rights with the WITH GRANT OPTION are able to display the access rights they grant to other users.

Examples

The following commands assign access rights.

```
GRANT INSERT ON employee TO mike, dale
GRANT SELECT, INSERT ON transmaster TO dawn +
    WITH GRANT OPTION
GRANT UPDATE ON transmaster TO andrea
GRANT UPDATE (company, custaddress, custcity) +
    ON customer TO mike, PUBLIC
```

Based on the access rights assigned with the GRANT commands above, the LIST ACCESS command would display the information below. Column names listed under Column are columns that are specified with the UPDATE access right. Columns are only displayed when a user has been assigned UPDATE rights to specific columns.

User Identifier	Table	Access	Column
MIKE	Employee	INSERT	
DALE	Employee	INSERT	
DAWN	TransMaster	*SELECT	
DAWN	TransMaster	*INSERT	
ANDREA	TransMaster	UPDATE	
MIKE	Customer	UPDATE	company custaddress custcity
PUBLIC	Customer	UPDATE	company custaddress custcity

Based on the access rights assigned with the GRANT commands above, the LIST ACCESS UPDATE command would display the information below.

User Identifier	Table	Access	Column
ANDREA	TransMaster	UPDATE	
MIKE	Customer	UPDATE	company custaddress custcity
PUBLIC	Customer	UPDATE	company custaddress custcity

Based on the access rights assigned with the GRANT commands above, the LIST ACCESS ON *transmaster* command would display the information below. The asterisk (*) by SELECT and INSERT shows that Dawn can grant those access rights to other users.

User Identifier	Table	Access	Column
DAWN	TransMaster	*SELECT	
DAWN	TransMaster	*INSERT	
ANDREA	TransMaster	UPDATE	

1.12.6 LOAD

Use the LOAD command to add data to a table or to a single table view that can be updated.

The four syntaxes for the LOAD command are:

Syntax 1:

```
LOAD tblview WITH PROMPTS [ FOR n ROWS ] [ USING collist ]
```

Syntax 2:

```
LOAD tblview FROM filespec [ AS ASCII  
                           [ AS FORMATTED USING colname start end  
                           AS CSU ] ]
```

Syntax 3:

```
LOAD tblview FROM filespec [ FOR n ROWS ] [ USING collist ] ...
```

```
... [ ; CHECK — ] [ ; FILL — ] [ ; NUM — ]  
    [ ; NOCHECK ] [ ; NOFILL ] [ ; NONUM ]
```

Syntax 4:

```
LOAD tblview [ FOR n ROWS ] [ USING collist ]  
CHECK/NOCHECK  
FILL/NOFILL  
NUM/NONUM  
data-block  
END
```

Options

' Indicates that this part of the command is repeatable.

AS ASCII

The LOAD AS ASCII command is designed strictly for speed of operation. LOAD AS ASCII checks rules and constraints. However, following is a list of the limitations of the LOAD AS ASCII command:

- It does not check the data types; therefore, invalid data will be loaded as null values into columns; no error messages about this conversion are displayed.
- It does not display error messages when columns must be truncated, or when excess data

- exists on any line.
- It does not echo data to the screen regardless of the setting for [SET ECHO](#).

To achieve maximum speed of loading, the data must look like the data that R:BASE unloads with the UNLOAD DATA AS ASCII command. That is, the data must conform to the following:

- The carriage return/line feed characters define the end of the line for a given row; the maximum row size is 32,768 characters.
- The data cannot include variables.
- The data cannot include comments.

AS FORMATTED USING

Loads data from an ASCII file when the data is formatted in fixed column locations, with the following restrictions:

- The carriage return/line feed characters define the end of the line for a given row; the maximum row size is 32,768 characters.
- The data cannot include variables.
- The data cannot include comments.
- You must specify the name of each column of the table to be loaded, and the starting and ending position of its data in the line, which is specified in the USING clause of this command.

CHECK

NOCHECK

CHECK turns on rule checking. When rule checking is on, R:BASE checks input against data validation rules. NOCHECK turns off rule checking. CHECK and NOCHECK override the current setting of the [SET RULES](#) condition. The default is CHECK.

colname start end

Specifies the name of a column in the table and the starting and ending position of its data in the line; this option is used with the AS FORMATTED option.

data-block

Includes lines of data to be loaded, as well as the LOAD subcommands.

For these data types...	Use this format for <i>data-block</i>
All data types except BIT, BITNOTE, LONG VARBIT, and VARBIT	'string' or value
BIT, BITNOTE, LONG VARBIT, LONG VARCHAR, VARBIT, and VARCHAR	['filename.ext'] or ['filename.ext', filetype, offset, length] Note: When you use VARCHAR, the filetype is always TXT. When you use VARBIT, BIT, and BITNOTE, filetype refers to the standard graphical file types.

FILL

NOFILL

FILL makes null any columns that have not been assigned values. All of the missing values must be at the end of the row. If a rule specifies that a column requires an entry other than null, do not use FILL. NOFILL turns off FILL and requires a value for each column. The default is NOFILL.

FOR n ROWS

Directs R:BASE to stop processing after loading *n* rows, where *n* is a positive whole number. In the fourth syntax diagram, END is not used if FOR *n* ROWS is included.

FROM filespec

Loads data into the specified table with data from an external ASCII delimited file.

NUM

NONUM

NUM specifies that autonumbering columns will be numbered as they are loaded. NONUM turns off autonumbering while loading, thereby allowing loading of a specific value for autonumber columns. The default is NUM.

tblview

Specifies a table or view name to load.

USING collist

Specifies the column(s) to use with the command.

WITH PROMPTS

Loads data into the specified table from keyboard entries. R:BASE asks for the values of each column by displaying the column name and its data type. To end the loading session, press [Esc].

About the LOAD Command

You cannot load data into a multi-table view.

Instead of using LOAD, you can also use [INSERT](#), the Data Editor, or a Form to add data to a table.

You can use the LOAD command to load data into R:BASE from a file that was not created by R:BASE. The file must be an ASCII file, either delimited or fixed.

The LOAD command will differentiate between END and 'END'; FILL and 'FILL'; NOFILL and 'NOFILL'; CHECK and 'CHECK'; NOCHECK and 'NOCHECK'; NUM and 'NUM'; NONUM and 'NONUM'. So, make sure to use the proper syntax when creating LOAD statements.

To ensure that data is placed in the intended column, use the following guidelines:

- Do not embed commas within entries for CURRENCY, DATE, DATETIME, DOUBLE, INTEGER, NUMERIC, or REAL data types. R:BASE automatically inserts commas and the current currency symbol.
- When values for CURRENCY, DOUBLE, NUMERIC, or REAL or data types are decimal fractions, you must enter the decimal point. When values are whole numbers, R:BASE adds a decimal point for you at the end of the number. R:BASE adds zeros for subunits in whole currency values. For example, using the default currency format, R:BASE loads an entry of 1000 as \$1,000.00.
- When values for NOTE or TEXT data types contain commas, you can either enclose the entries within quotes, or use [SET DELIMIT](#) to change the default delimiter (comma) to another character.
- When values for NOTE or TEXT data types contain single quotes ('), and you are using the default QUOTES character ('), use two single quotes (') in the text string. For example, 'Walter Finnegan's order.'
- When a value you specify for a column is not the same data type as the column's data type, R:BASE displays an error message and you need to re-enter the entire row.
- When values for NOTE or TEXT data types exceed the maximum length of a column, R:BASE truncates the value and adds it to the table. A message is displayed that tells you which row has been truncated.

Loading with a USING Clause

A USING clause is helpful when you do not have all the information that is to be added to a table. The following example lets you enter some information for a product but does not require that all columns be entered. The *model* and *listprice* columns are the first and last columns in the *product* table. The *prodname* and *proddesc* columns are not included in the command and are loaded with null values. You can later edit the *product* table to enter data into the columns that have null values.

```
LOAD product USING model listprice
```

Loading with the CHECK Option

The [SET RULES](#) condition does not have any effect on the CHECK option because CHECK has precedence over a RULES setting. When RULES is set off, the CHECK option still verifies data entry against existing rules.

When a user identifier has been assigned to the database owner, you must enter the owner's user identifier with the [CONNECT](#) or [SET USER](#) command before you use the CHECK or NOCHECK option. R:BASE does not accept the CHECK or NOCHECK option unless the owner's user identifier has been entered.

Loading Computed Columns

You cannot load data directly into a computed column. After you load the column values that are used to calculate the computed column, R:BASE fills the computed column with the computed value.

The setting of the [SET ZERO](#) condition affects the calculation of numeric computed columns. To have null values treated as zeros in expressions, set ZERO on. When ZERO is set off, if the value of a column used in an expression is null, the computed value will be null.

Loading Negative CURRENCY values

When loading negative CURRENCY values into a table, the format must include the hyphen, i.e. -\$500.00. Negative CURRENCY values encased in parenthesis are not recognized, i.e. (\$500.00).

Loading with Prompts

When you run the LOAD command using prompts, you load one row of data at a time into the table you specified. (See Example 1). For each new row you add, R:BASE displays the name and data type of the row's column as prompts. At each prompt, you enter the value that you want the column to contain. You are prompted for each column in the row beginning with the first column, unless you used a USING *collist* clause to limit the number of the columns to load, or to change the order in which the columns are loaded. Any columns not listed in the *collist* are given null values when the rows are entered.

When you load data with prompts, the default length for a text entry is 80 characters. To enter columns with a NOTE or TEXT data types that contain more than 80 characters, load the data without prompts, make a custom data-entry form, or set the WIDTH so you can enter more characters.

R:BASE does not prompt you for computed or autonumbered column values.

Loading without Prompts

Loading without prompts is faster but requires that you remember the order of the columns in the table. When you load without using prompts and not from an ASCII file, the LOAD command provides its own distinctive prompt. The following options can be entered at this prompt: CHECK, NOCHECK, FILL, NOFILL, NUM, and NONUM.

Loading from an ASCII File

Use the LOAD command from the R> Prompt or a command file to load data into an existing table from both delimited and fixed field ASCII files. Each record in the ASCII file corresponds to one row of data in a table, and each item of data in a record corresponds to one column value in a row. Therefore, organize data in the file in the same order as the columns in the table to be loaded.

Items of data in a line of the ASCII file must be delimited to be properly placed within the columns of a row. The delimiter character must be the same as the current delimiter character specified with the [SET DELIMIT](#) setting. (The default delimiter is a comma.) R:BASE also accepts a blank space as a delimiter, regardless of the setting of the DELIMIT setting.

Data can be loaded in a fixed-field formatted ASCII file with the AS FORMATTED option. The column name and the start and end positions within the file must be specified for each value in the row of data that is to be loaded. When the start and end positions are specified, the delimiter character does not have any effect because the start and end positions for each column identify the data.

When loading from a file, be sure that the current null symbol is not a blank. If the first four characters of a field in a file are blank, R:BASE adds the column as a null column and does not read any additional characters that make up the field value.

When loading data from an ASCII file, make sure the file meets the following requirements listed in the table below.

Elements in an ASCII File	Requirement
INTEGER data types	Items of data to be loaded into columns with INTEGER data types cannot contain internal commas unless the item is enclosed in quotes. The default QUOTES character in R:BASE is a single quote ('); if your ASCII file uses double quotes ("), change the QUOTES setting before you load the file. If the file does not have quotes around the integer values containing commas, you must edit the ASCII file to remove any commas from the integer values, or enclose each integer value in quotes.
Embedded punctuation	Items of data containing ampersands, commas, embedded blanks, plus signs, equal signs, or semicolons must be enclosed in quotes if they are to be loaded into columns with a TEXT or NOTE data type. The default QUOTES character in R:BASE is a single quote ('); if your

	ASCII file uses double quotes ("), change the QUOTES setting before you load the file.
Embedded quotes	Items of data requiring quotes can also contain embedded quotes. For example, the item 'Basic' Keyboard contains both a blank space and embedded quotes. Using single quotes ('), which is the default QUOTES setting, to add enclosing quotes, the item would look like this: "'Basic' Keyboard'
Currency	R:BASE automatically adds a currency symbol, commas, and zeros for currency units. For example, using the default currency format, R:BASE loads an entry of 1000 as \$1,000.00.
Dates	The SET DATE SEQUENCE command sets the sequence for the dates in the file are loaded if the dates match the current date sequence established with the SET DATE command.
Computed columns	If the table being loaded has computed columns and the file contains values for the computed columns, R:BASE tries to load the computed column's value from the file into the column following the computed column. This results in an error because the data type of the next column might not be the correct data type, or the file will have too many values for the table because R:BASE does not load the computed column's value from the file.
Rules processing	Unless you run the SET RULES OFF condition before loading the file, rules processing is in effect. When an incoming data item violates a rule, R:BASE does not load the row. Instead, R:BASE displays the message for the rule that has been violated. To see the data that causes a rule violation, SET ECHO ON when loading a table and use the [Pause] key to stop the screen from scrolling when the rule violation occurs.

Loading a Data Block

The data block shown in the diagram can include lines of data and any of the options available with LOAD-CHECK/NOCHECK, FILL/NOFILL, and NUM/NONUM. You can intersperse the options with data lines, and you can enter more than one option on a line if you separate the options with semicolons. However, you cannot combine data and options on the same line.

R:BASE displays the dialog prompt to accept data-block entry. LOAD adds data to a table, row by row, without using a data-entry form and without prompting for each data item.

You can enter the options for the LOAD command at the dialog prompt at any time during data loading. Or you can include them on the command line, separated from the command by semicolons, as shown in the example below. (Do not use this format in command or procedure files. All options must follow the LOAD command on separate lines in command or procedure files.)

```
LOAD transdetail ; CHECK ; NUM
```

You can use global or system variables instead of constant values in the data block.

To enter values properly, use the following guidelines.

- Enter column values in the order that columns are defined in the table, and separate the values with a delimiter character. The default delimiter character is the comma.
- You can enter up to 75 characters on a single line. If a row is longer than 75 characters, continue on to the next line by typing past the end of the current line or by entering a plus (+) sign at any point on the current line. The plus sign must be the last entry on the line. The new line will begin with a +> prompt to indicate the continuation of the current line. If you are using this form of the LOAD command in a command file, you must use a + to continue on the next line; the lines will not automatically wrap.
- For other requirements on loading data, see "Loading from an ASCII File" earlier in this entry.

Examples

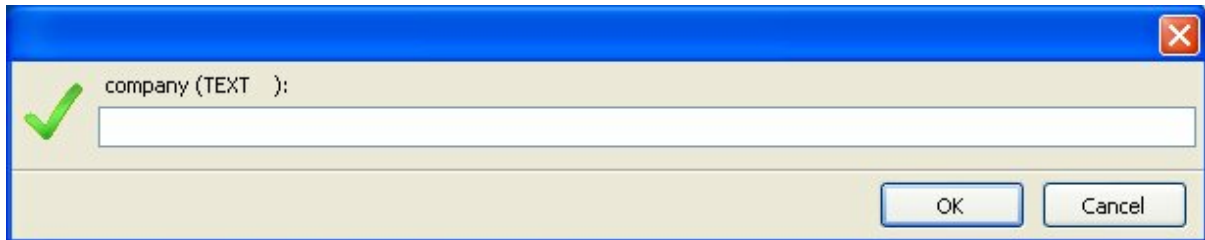
Example 01:

The following command line allow you to load rows containing new customer information to the *customer* table. R:BASE asks for each column by column name and data type. Two columns in the table, *custid* and *custphone*, are omitted from the list. R:BASE automatically supplies a number for the *custid* column because it is an autonumbered column. R:BASE leaves the *custphone* column empty (null) when data is

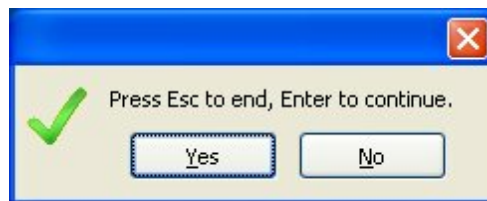
loaded, and does not prompt for either column.

```
LOAD customer WITH PROMPTS USING company, +
custaddress, custcity, custstate, custzip
```

After the above command is run, the WITH PROMPTS option displays the message below.



If you press the [Esc] key or the "Cancel" button before you have finished entering data in a row, the row is not added to the table. You will be prompted to add another row.



To continue, press the "Yes" button. To exit, press the "No" button or the [Esc] key.

Example 02:

The following command loads five rows of data into the *customer* table from CUST.DAT, a delimited ASCII file. The data in the ASCII file must be in the same order as the columns in the *customer* table. Only the first five lines from the file will be loaded:

```
LOAD customer FROM cust.dat FOR 5 ROWS
```

Example 03:

In the following example, the command line tells R:BASE to start loading data for the *customer* table. A dialog prompt is displayed for each new row. Each column value would be entered in this one dialog and separated with a comma, or the current delimiter. The length of the text available to fit in the dialog is 4096 characters.

```
LOAD customer
```

Example 04:

After the command line in the following example is run, R:BASE expects the next five lines entered at the dialog prompt to contain data to be loaded into the *customer* table. After the fifth line of data is entered, the loading ends. To end loading before five rows of data are entered, enter END.

```
LOAD customer FOR 5 ROWS
```

Example 05:

The following command lines show you how to load data into the *company*, *custaddress*, *custcity*, *custstate*, and *custzip* columns of the *customer* table. The *custid* and *custphone* columns in the *customer* table will not have data loaded and will be given null values.

```
LOAD customer FROM customer.fix AS FORMATTED +
USING company 11 50, custaddress 51 80, +
custcity 81 100, custstate 101 102, custzip 103 112
```

Example 06:

When you use the LOAD command, you must omit values for computed or autonumbered columns. Instead, enter the value for the next column in the data list. In the following example, to add a row to the *transdetail* table, which has a computed column, you would only enter data for the first five columns; the sixth column is a computed column based on the fourth and fifth columns. The columns entered are *transid*, *detailnum*, *model*, *units*, and *price*. The computed column is *extprice* and has the expression (*units * price*).

```
LOAD transdetail
  6000,1,'CX3000',100,$1900
END
```

1.13 M

1.13.1 MAXIMIZE

Use the MAXIMIZE command to maximize the R> Prompt window.

MAXIMIZE

1.13.2 MINIMIZE

Use the MINIMIZE command to minimize the R> Prompt window.

MINIMIZE

1.13.3 MKDIR

Use the MKDIR command to create a new directory.

```
MKDIR [dirname]
      [C:\...\dirname]
```

Options

pathname

Specifies one or more directory names that lead from the root directory to a specified directory.

About the MKDIR Command

The R:BASE MKDIR command is similar to the operating system MKDIR command. For more information about the MKDIR command, refer to your operating system manual.

Examples

The following command creates the OLDDATA directory as a sub-directory in the current directory.

```
MKDIR olddata
```

1.14 N

1.14.1 NEWPAGE

Use the NEWPAGE command to send a form feed to the current output device: printer, screen, or file.

NEWPAGE

About the NEWPAGE Command

When the current output device is the computer screen, entering the NEWPAGE command clears the window and moves the R> Prompt and cursor to the top of a new window. When the current output device is a printer, entering the NEWPAGE command sends a form feed that advances the paper to the top of the next page. When the current output device is a file, entering the NEWPAGE command sends a form feed character to the file.

1.14.2 NEWROW

Use the NEWROW command in an entry/exit procedure (EEP) to add a row.

NEWROW

About the NEWROW Command

The NEWROW command displays a blank data entry form for the user to enter data.

To see the effect of the NEWROW command, you must include the [SCREEN RESTORE OFF](#) command in the EEP.

1.14.3 NEXTROW

Use the NEXTROW command in an entry/exit procedure (EEP) to move to the next row.

NEXTROW

1.14.4 NEXTTAB

Use the NEXTTAB command in an entry/exit procedure (EEP) to move to the next table.

NEXTTAB

1.14.5 NORMALIZE

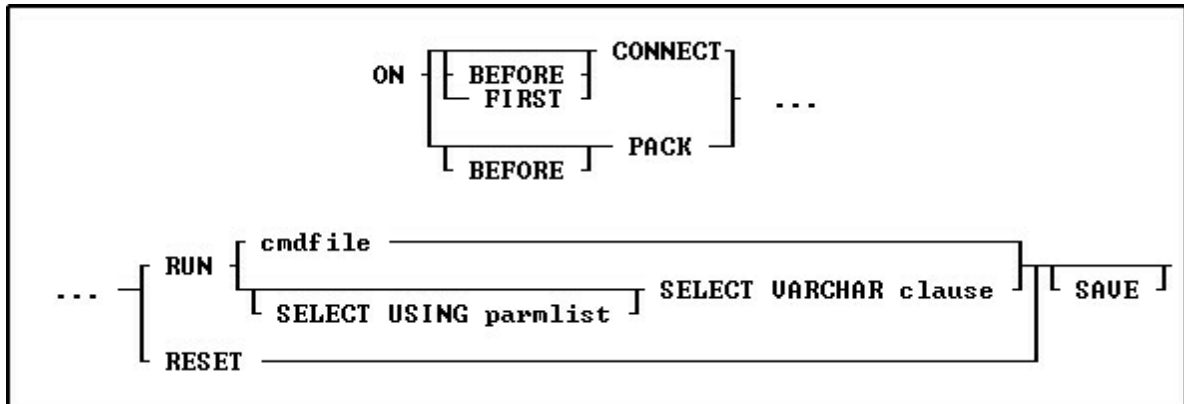
Use the NORMALIZE command to size the R> Prompt window to the default size.

NORMALIZE

1.15 O

1.15.1 ON CONNECT/PACK

Use the ON CONNECT/PACK command to run a command file before or after connecting to or packing a database.



Options

BEFORE

Indicates the command file should be run before the CONNECT/PACK.

CONNECT

Indicates the command file should be run when connecting a database.

FIRST

Indicates the command file should be run on the first connect only.

PACK

Indicates the command file should be run when packing a database.

RESET

Clears out the ON CONNECT/PACK command recorded in the database.

cmdfile

Specifies the name of the command file to execute.

SAVE

Specifies that the parameters in the **cmdfile** or from the **RESET** will be saved to the database. Any options without the SAVE parameters will only be active for a session.

SELECT VARCHAR clause

Specifies a column defined with the VARCHAR data type from a table, from which you can run the contents. The [SELECT](#) clause must limit the data to only one row; otherwise, an error is returned.

USING parmlist

Lists the values the command file uses when it runs. The parameter list can contain up to 18 values. The first value in the list is referenced in the executed file as %1, the second as %2, and so on through %9. They are treated just like other variables. To reference the contents of these variables, preface the variable name with a dot (.); for example, set v1 =.%1.

About the ON CONNECT/PACK command

The ON CONNECT/PACK command is useful for running a specific command file whenever a database is connected or packed. A common use would be to run a command file that does an [AUTOCHK](#) command before connecting. The [ABORT ON](#) command could then be used to cause the CONNECT/PACK to be

aborted after the command file finishes.

1.15.2 ON ERROR

Use the ON ERROR command to execute a command if an error occurs.

ON ERROR { command RESET

Options

command

Specifies the command to run when an error occurs.

RESET

Turns ON ERROR processing off.

About the ON ERROR Command

The default is ON ERROR RESET.

Any command that can be used in an entry/exit procedure (EEP) can be used with ON ERROR except: [TRACE](#), form specific commands such as [NEXTROW](#), [SKIP](#), etc., and commands more than 2000 characters long. If a command can't be used in an EEP, such as [DROP](#), [CREATE](#), [CONNECT](#), and [DISCONNECT](#), it can't be used with ON ERROR.

Since this command remains in effect until it is re-issued, you should place ON ERROR RESET at the end of command files that use it. If ON ERROR runs a command file, the file must end with a [RETURN](#) command.

1.15.3 OPEN

Use the OPEN command before using the cursor designated by the [DECLARE CURSOR](#) command.

OPEN cursor [RESET]

Options

cursor

Specifies a 1 to 18 character *cursor* name that has been previously specified by the DECLARE CURSOR command.

RESET

Reopens a cursor with the current values of any variables referenced in the DECLARE CURSOR statement. This improves performance by eliminating the need to re-optimize the query.

About the OPEN Command

OPEN evaluates the [SELECT](#) clause of the [DECLARE CURSOR](#) command using the current values of any variables that it contains. Then OPEN stores that copy of the cursor definition and places the cursor before the first row.

After you close a cursor with the [CLOSE](#) command, you can reopen it by repeating the OPEN command. Every time you open a cursor, R:BASE reads the rows again, so that any changes you previously made through the cursor are visible when you look at the rows.

You can display all currently defined cursors with the [LIST CURSOR](#) command.

When using the RESET option, the WHERE clause is evaluated with the current values of any referenced variables, and the cursor is reopened without requiring a CLOSE command. The cursor is positioned at the beginning of the result set when the [FETCH](#) command is run.

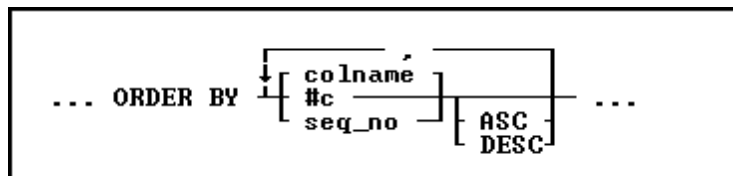
Example

The following command lines show the OPEN command with the RESET option.

```
DROP CURSOR c1
DROP CURSOR c2
SET VAR vc1custid INTEGER
DECLARE c1 CURSOR FOR SELECT custid FROM customer
-- Selects the transaction rows for the customer
DECLARE c2 CURSOR FOR SELECT transid, invoicetotal +
    FROM transmaster WHERE custid = .vc1custid
-- Process the query in cursor c1 and get the first custid
OPEN c1
FETCH c1 INTO vc1custid IND c1ind1
WHILE sqlcode <> 100 THEN
    -- Process the query in c2. As each row is fetched in the
    -- customer table, the custid changes; each time
    -- the "OPEN c2 RESET" command processes the c2's query it
    -- retrieves different rows
    OPEN c2 RESET
    -- Fetch the transid (invoicenumber) and invoice total amount
    FETCH c2 INTO vtransid, vamt
    WHILE sqlcode <> 100 THEN
        WRITE .vtransid, .vamt
        FETCH c2 INTO vtransid, vamt
    ENDWHILE
    FETCH c1 INTO vc1custid
ENDWHILE
CLOSE c1
CLOSE c2
DROP CURSOR c1
DROP CURSOR c2
```

1.15.4 ORDER BY

Use the ORDER BY clause with an R:BASE command to specify the order in which rows of data from a table are displayed.



Options

,

Indicates that this part of the command is repeatable.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

#c

Takes the place of a column name and refers to the column numbers displayed with the [LIST TABLE](#) command.

colname

Sorts by any column name or combination of column names.

seq_no

Refers to the items listed in the [SELECT](#) command that is using the ORDER BY command, ordered from left to right. An item can be a column name, expression, or SELECT function.

About the ORDER BY Command

The syntax for the ORDER BY clause is the same for all commands. ORDER BY must refer to only one table or view.

You can significantly reduce the time R:BASE takes to process an ORDER BY clause when the column or columns listed in the ORDER BY clause are included in an index with the same column sort order as that specified in the ORDER BY clause.

Using the SET SORT Command

The ORDER BY command uses the R:BASE automatic sort optimizer. If you are sorting extremely large tables, and if your disk space is limited, the automatic sort optimizer might be unable to sort the data. Instead, use the [SET SORT ON](#) command because it uses the least disk space necessary to sort data; however, the SET SORT ON command is slower than the automatic sort.

Examples

The following command displays data from the *custid*, *company*, and *custcity* columns from the *customer* table.

```
SELECT custid, company, custcity FROM customer
```

The ORDER BY clause in the command below arranges the *custid* values in descending order.

```
SELECT custid, company, custcity FROM customer +  
ORDER BY custid DESC
```

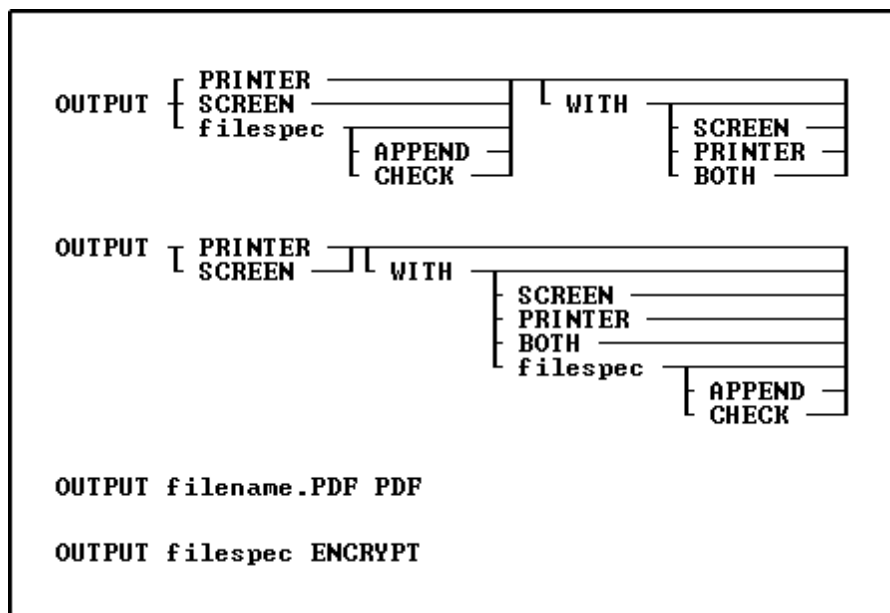
You can substitute a column's sequence number for a column named in the ORDER BY clause. You must use a sequence number when referring to an expression, function, constant, or when a UNION operator is used. The following command is equivalent to the command example above.

```
SELECT custid, company, custcity FROM customer ORDER +  
BY 1 DESC
```

R:BASE for DOS only: You can also specify the maximum and minimum memory allocated with the SET SORT command using the MAX and MIN functions. You can show the current memory allocation settings with SHOW SORT using the MAX, MIN, and LAST functions-LAST shows the amount of memory you need to perform the last sort.

1.15.5 OUTPUT

Use the OUTPUT command to direct messages and results of commands to a file, printer, screen, or a combination of the three destinations.



Options

APPEND

Appends data to the end of an existing file without overwriting the file. If you specify APPEND when the specified file does not exist, R:BASE creates the file with that name.

BOTH

Refers to PRINTER and SCREEN.

CHECK

Checks for file existence and prompts the user for confirmation before writing to it.

ENCRYPT

A 512-bit encryption method is used to obscure any output information, making it unreadable without R:BASE and your decryption password. Immediately after using the ENCRYPT parameter in your OUTPUT command, you will be prompted for a password. The password is limited 32 characters.

When running encrypted files with R:BASE, you would [RUN](#) the file followed by the password. When opening an encrypted file, you will be prompted for the password.

filespec

Indicates the output device. Specify a file name, with or without an extension. You can also specify a drive and/or path.

printer

Indicates the output device. *Printer* is either the word PRINTER or a valid output port such as LPT1 or COM1. An output port is treated as a file that is automatically copied to the printer device when an OUTPUT SCREEN command is issued.

SCREEN

Indicates the output device is the screen.

filename.PDF PDF

Will create the output as PDF.

About the OUTPUT Command

The default output device is the screen. After sending data to a printer or file, change the output device to the screen. When using the OUTPUT *filespec* command, you might lose the data in the file if you exit from R:BASE without sending an OUTPUT SCREEN command to close the file.

You cannot combine two *filespec* designations such as `OUTPUT filespec WITH filespec`.

R:BASE cannot tell that the argument LPT1 or COM1 is not the name of a file. For example, if you use the command `OUTPUT LPT1`, R:BASE creates a file named LPT1. Because R:BASE thinks LPT1 is a file, R:BASE does not print the last buffer of output until you redirect output to the screen.

Sending Data to a Printer

Printer refers to the system default printer. In Windows to change a printer, use the Windows Control Panel. In DOS, *printer* refers to the printer that is attached to LPT1.

If you are using R:BASE on a local area network, you can direct output to any printer to which you have network access. A common method is to enter the `OUTPUT` command followed by either the network name of the printer or the printer port on the server.

Sending Data to the Screen

The [SET LINES](#) command determines how many lines display on a screen. If a file contains more lines than are designated to display on a screen, R:BASE pauses after each screen display. When displaying a report on a screen, the number of lines specified in the report are displayed.

If the primary output device is a file or a printer and the secondary output device is the screen, R:BASE pauses after each screen display.

Backing Up with OUTPUT

The `OUTPUT` command can be used with the [BACKUP](#) or [UNLOAD](#) commands to back up your database to a file. If some part of the command is incorrect—for example, if you fail to specify the correct user identifier for the database—the resulting file contains only the error message produced by the incorrect entry. Use the [TYPE](#) or [DISPLAY](#) command to display the contents of the file and view the results.

Using SET ECHO ON

To send both your commands and the results of the commands to an output device, enter the command [SET ECHO ON](#) before beginning the `OUTPUT` session. The `SET ECHO OFF` command only sends command results to an output device.

Examples

In the following example, the `OUTPUT` command sends data and messages to the printer and simultaneously displays the data and messages on screen with pauses between screens, if needed.

`OUTPUT PRINTER WITH SCREEN`

In the following example, the first `OUTPUT` command directs output to the `BACKUP.DAT` file on drive A:. The `UNLOAD` command sends the data stored in the *transmaster* table to the file. The second `OUTPUT` command closes the file and redirects output to the screen.

```
OUTPUT a:\backup.dat
UNLOAD DATA FOR transmaster
OUTPUT SCREEN
```

The following command sends data and messages to the printer, designated as COM1. COM1 is treated as a file that is automatically copied to a printer device when the `OUTPUT SCREEN` command is issued.

`OUTPUT COM1`

Additional parameters of `FONTNAME` and `FONTSIZE` expands the use of printer outputs with your favorite fonts and size.

`OUTPUT PRINTER <FONTNAME font name> <FONTSIZE font size>`

The default font name is Courier New and font size 10.

Example 01:

`OUTPUT PRINTER`

```
SELECT Company, CustCity, CustState FROM Customer
OUTPUT SCREEN
-- Resulting output will be Courier New 10
```

Example 02:

```
OUTPUT PRINTER FONTSIZE 8
SELECT Company, CustCity, CustState FROM Customer
OUTPUT SCREEN
-- Resulting output will be Courier New 8
```

Example 03:

```
OUTPUT PRINTER FONTSIZE 12
SELECT Company, CustCity, CustState FROM Customer
OUTPUT SCREEN
-- Resulting output will be Courier New 12
```

Example 04:

```
OUTPUT PRINTER FONTNAME Arial FONTSIZE 10
SELECT Company, CustCity, CustState FROM Customer
OUTPUT SCREEN
-- Resulting output will be Arial 10
```

Example 05:

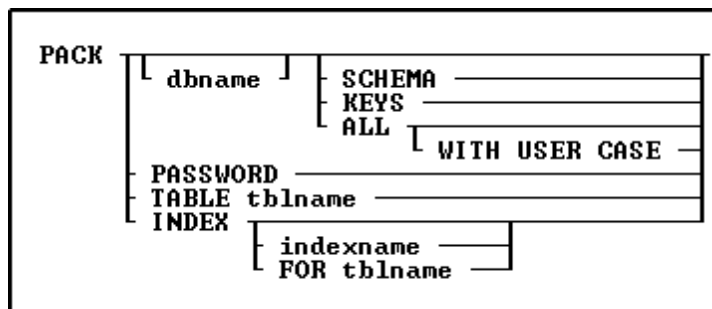
```
OUTPUT CustomerList.PDF PDF
SELECT Company FROM Customer ORDER BY Company
OUTPUT SCREEN
```

Then, you can view the PDF output using the [LAUNCH](#) command.

1.16 P

1.16.1 PACK

Use the PACK command on an open database to recover unusable disk space.



Options

ALL

Packs File 1 (schema information), File 2 (data), File 3 (indexes), and File 4 (large object data).

dbname

Specifies the name of the database to pack.

FOR tblname

Specifies a particular table whose indices you want to PACK.

INDEX

Use this option to PACK all indices for the currently connected database. PACK INDEX is supported in multi-user environments.

indexname

Packs a specified index from File 3 (indexes); this option will work when [STATICDB](#) is set on. PACK INDEX indexname is now supported in multi-user environments. This command will execute on the currently connected database.

KEYS

Packs only File 3 (indexes).

PASSWORD

Use this option to clean out bogus rows from SYS_PASSWORDS table.

PACK PASSWORD command is also supported in a multi-user session. Database must be connected in order to use this command.

SCHEMA

Packs only File 1 (schema information).

TABLE tblname

Packing a single table when [MULTI](#) is set ON. This parameter is very beneficial with databases that are always in use.

WITH USER CASE

Replaces the case folding/collating tables in the database with those defined in the user configuration file (RBENGINE9.CFG).

About the PACK Command

Disk space becomes unusable when you delete rows or indexes, remove columns or tables, or add or modify columns with the [ALTER TABLE](#) command. To use the PACK command, a database must be open. If the database you want to pack is not in the current directory, include the drive, path, and database name. When you pack a database that is in a different directory, R:BASE closes any open database, then opens the database you want to pack.

[PACK](#) requires the database owner's user identifier if the database is protected by the owner's user identifier.

You cannot use PACK when a database is stored on a network drive and [MULTI](#) has been set on, unless you are ONLY packing one table. Set MULTI off before packing the database.

PACK is unavailable when transaction processing is on.

Because you pack an open database, back up your database before you pack it. An interruption to a pack could cause damage to your database.

PACK Versus RELOAD

Both the [PACK](#) and [RELOAD](#) commands recover unusable disk space; however, RELOAD requires more disk space than PACK because RELOAD copies a database table by table, collects the rows of each table, then reorganizes the rows on the disk.

PACK KEYS versus PACK INDEX

PACK KEYS is to be used with [MULTI](#) set OFF as it recreates a new index file with clean indexes. PACK INDEX can be used while users are connected to the database with MULTI set ON, only adding to the current index file. After using PACK KEYS, you should see a decrease in the index file size.

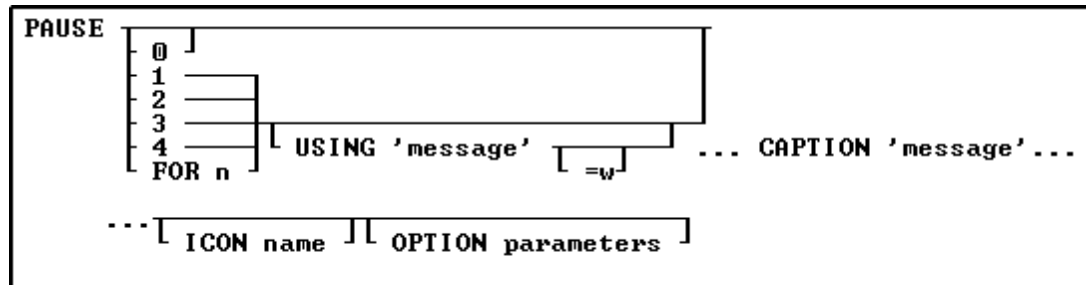
Example

The following command packs the *concomp* database in the RBTI directory on drive C:

```
PACK c:\rbti\concomp
```


1.16.2 PAUSE

Use the PAUSE command to display a message and can also suspend the running of the command file. For assistance with building your PAUSE commands, refer to the [R:Pause Builder Plugin](#).



Options

0

Pauses without a message.

1

Displays "Press OK to continue" or any message using the USING option in a dialog box. The R> Prompt screen then clears with the next keystroke.

2

Same as the 1 option, except the R> Prompt screen does not clear with the next keystroke.

3

Does not pause the running of the command file and therefore does not wait for the next keystroke.

4

If a PAUSE 3 dialog already exists only the message will be repainted to avoid flickering. Otherwise, this is exactly like the PAUSE 3 option.

FOR n

Sets the pause duration in seconds; **n** must be a positive integer. Any keystroke interrupts the pause, regardless of duration. In the absence of a USING clause, no message is displayed. With a USING clause, the message is displayed in a dialog box

USING 'message'

Displays the specified message in a dialog box. This value can also be passed as a variable.

Using this default parameter the maximum width of message text in a Windows dialog box is 77 characters and is limited to one line. However, if you need to display a multi-line PAUSE window, you can create separate lines with the ASCII characters for a carriage return and indent ([Tab] key). A sample is provided below.

=width

Specifies the wrap width for the dialog box message.

CAPTION 'message'

Specifies the text of the message to display in the dialog box caption. The value can also be passed as a variable.

ICON value

Icon "value" Parameter	Icon
APPS	

ATTENTION	
CONFIRM	
ERROR	
HELP	
INFO	
QUESTION	
SERIOUS	
STOP	
WARNING	
WINDOWS	

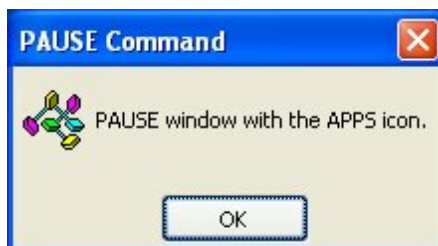
Additional [OPTION parameters](#)

Additional parameters are available to increase the visual display of the PAUSE window. To use the graphic PAUSE Builder, choose "Utilities" > "Plugins" > "Internal Plugins" > "PAUSE Builder" from the main Menu Bar. All OPTION parameters and values must be separated by the "|" (pipe) character.

Examples:

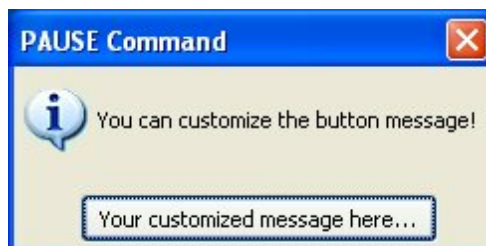
CLS

PAUSE 2 USING 'PAUSE window with the APPS icon.' CAPTION 'PAUSE Command' ICON APPS



CLS

PAUSE 2 USING 'You can customize the button message!' CAPTION 'PAUSE Command' ICON INFO BUTTON 'Your customized message here...'



-- Standard PAUSE Command with ICON

```

PAUSE 2 USING 'Message Text' +
CAPTION 'Caption Text' +
ICON HELP OPTION +
BUTTON 'Button Text' +
|BACK_COLOR WHITE +
|MESSAGE_COLOR WHITE +
|MESSAGE_FONT_COLOR GREEN +
|BUTTON_COLOR WHITE +
|BUTTON_FONT_COLOR GREEN +
|TRANSPARENCY 255

```

-- Standard PAUSE Command with custom ICON

```

PAUSE 2 USING 'Message Text' +
CAPTION 'Caption Text' +
BUTTON 'Button Text' +
OPTION ICON_FILE path\directory\filename.bmp +
|BACK_COLOR WHITE +
|MESSAGE_COLOR WHITE +
|MESSAGE_FONT_COLOR GREEN +
|BUTTON_COLOR WHITE +
|BUTTON_FONT_COLOR GREEN +
|TRANSPARENCY 255

```

Note: Custom icon to be used in PAUSE option must be 16x16 pixels BMP file.

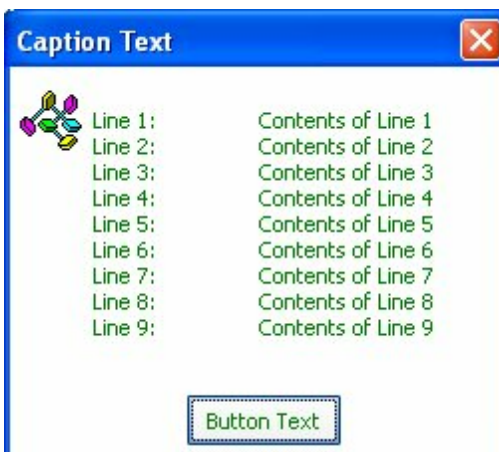
-- Multi-Line PAUSE Command

```

-- (CHAR(009)) = Tab Key (Indent)
-- (CHAR(013)) = Carriage Return
CLS
SET VAR vMsg = +
('Line 1:'+(CHAR(009))+(CHAR(009))&'Contents of Line 1'+(CHAR(009))+(CHAR(013))+ +
'Line 2:'+(CHAR(009))+(CHAR(009))&'Contents of Line 2'+(CHAR(009))+(CHAR(013))+ +
'Line 3:'+(CHAR(009))+(CHAR(009))&'Contents of Line 3'+(CHAR(009))+(CHAR(013))+ +
'Line 4:'+(CHAR(009))+(CHAR(009))&'Contents of Line 4'+(CHAR(009))+(CHAR(013))+ +
'Line 5:'+(CHAR(009))+(CHAR(009))&'Contents of Line 5'+(CHAR(009))+(CHAR(013))+ +
'Line 6:'+(CHAR(009))+(CHAR(009))&'Contents of Line 6'+(CHAR(009))+(CHAR(013))+ +
'Line 7:'+(CHAR(009))+(CHAR(009))&'Contents of Line 7'+(CHAR(009))+(CHAR(013))+ +
'Line 8:'+(CHAR(009))+(CHAR(009))&'Contents of Line 8'+(CHAR(009))+(CHAR(013))+ +
'Line 9:'+(CHAR(009))+(CHAR(009))&'Contents of Line 9'+(CHAR(009))+(CHAR(013)))

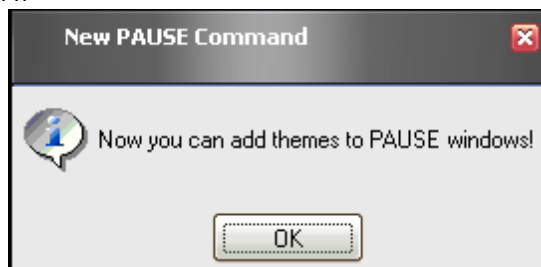
PAUSE 2 USING .vMsg +
CAPTION 'Caption Text' +
ICON APP +
BUTTON 'Button Text' +
OPTION BACK_COLOR WHITE +
|MESSAGE_COLOR WHITE +
|MESSAGE_FONT_COLOR GREEN +
|BUTTON_COLOR WHITE +
|BUTTON_FONT_COLOR GREEN +
|TRANSPARENCY 255

```



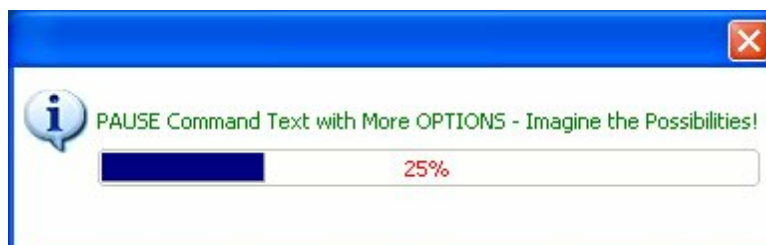
-- Standard PAUSE Command with custom ICON with Themes

```
PAUSE 2 USING +
'Now you can add themes to PAUSE windows!' +
CAPTION 'New PAUSE Command' ICON INFO +
OPTION themename Longhorn
```



-- PAUSE Command with Meter Progress Bar

```
PAUSE 3 USING +
'PAUSE Command Text with More OPTIONS - Imagine the Possibilities!' +
CAPTION ' ' +
OPTION METER_VISIBLE ON +
|METER_VALUE .vProgress +
|METER_TYPE BAR3D +
|METER_BACK_COLOR WHITE +
|METER_BAR_COLOR NAVY +
|METER_FONT_COLOR RED +
|BACK_COLOR WHITE +
|MESSAGE_COLOR WHITE +
|MESSAGE_FONT_COLOR GREEN
```



-- PAUSE Command with GAUGE Progress Bar

```
PAUSE 3 USING 'Calculating ... Please Stand By ...' +  
CAPTION '                Pause 3 with Gauge ' +  
ICON WINDOWS OPTION GAUGE_VISIBLE ON +  
|GAUGE_COLOR RED +  
|GAUGE_INTERVAL 10 +  
|MESSAGE_FONT_NAME VERDANA +  
|MESSAGE_FONT_SIZE 10 +  
|MESSAGE_FONT_COLOR WHITE +  
|THEMENAME Steel Blue
```

**1.16.2.1 OPTION parameters**

These parameters can be used to change the "look and feel" of a PAUSE window. In addition to the parameters below, you can also alter the PAUSE [Message](#), as well as add a [Meter](#) or [Gauge Meter](#).

ICON_FILE *value*

Specifies the external path and image name for the icon (file size must be 16x16 pixels).

BACK_COLOR *value*

Changes the color of the WINDOW BACKGROUND area. User can specify the integer value or type one of the predefined values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL

- WHITE
- YELLOW

BUTTON_COLOR *value*

Changes the background color of the button. Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

BUTTON_FONT_COLOR *value*

Changes the font color of the button. Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

TRANSPARENCY *value*

Set the transparency to specify a layered window with a translucent color. The transparency value specifies the degree of translucency. Values:

0 - 255

TOP *value*

Moves the pause box location, in pixels, from the top of the screen down

LEFT *value*

Moves the pause box location, in pixels, from the left of the screen down

THEMENAME *value*

Specifies one of 86 pre-defined Themes, or a custom Theme loaded into R:BASE. See Themes.

NOTE: All previously defined PAUSE commands which are redefined to use Themes should be thoroughly checked prior to putting into production as objects and text may look substantially different.

1.16.2.1.1 Message

These parameters the message within the PAUSE window.

MESSAGE_COLOR *value*

Changes the color of the actual message. Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

MESSAGE_FONT_COLOR *value*

Changes the color of message font. Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON

- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

MESSAGE_FONT_NAME *value*

Specifies a font name for the message. Values would consist of the fonts available on the computer.

MESSAGE_FONT_SIZE *value*

Specifies the font size

MESSAGE_FONT_BOLD *value*

Specifies if the message font is bold type. Values:

- ON
- OFF

MESSAGE_FONT_ITALIC *value*

Specifies if the message font is italic type. Values:

- ON
- OFF

MESSAGE_FONT_UNDERLINE *value*

Specifies if the message font is underline type. Values:

- ON
- OFF

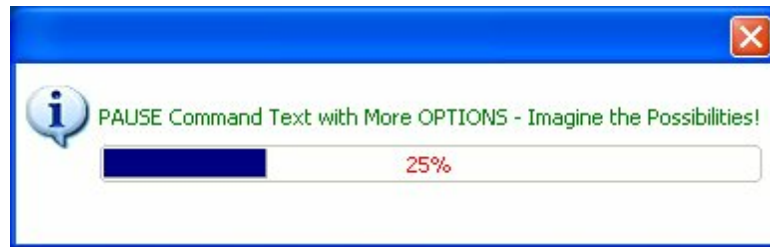
MESSAGE_FONT_STRIKEOUT *value*

Specifies if the message font is strikeout type. Values:

- ON
- OFF

1.16.2.1.2 Meter

These parameters will display a meter within the PAUSE window.

**METER_VISIBLE** *value*

Displays a meter to show a progress in percentage values. Values:

- ON
- OFF

METER_TYPE *value*

Three meter options are available. Values:

- NORMAL
- BAR3D
- LED

METER_VALUE *value*

Specifies the percentage to display the meter progress. Value:

0 to 100

METER_BAR_COLOR *value*

Specifies the color of the meter bar (color values listed below).

METER_BACK_COLOR *value*

Specifies the meter background color (color values listed below).

METER_FONT_COLOR *value*

Specifies the color of the meter font (color values listed below).

METER_MAX_VALUE *value*

Specifies the maximum percentage to display the meter progress. Values:

0 to 100

METER_COLOR1 *value*

Specifies the color of the meter when using three color segments (green, yellow, red) (color values listed below).

METER_COLOR2 *value*

Specifies the color of the meter when using three color segments (green, yellow, red) (color values listed below).

METER_COLOR3 *value*

Specifies the color of the meter when using three color segments (green, yellow, red) (color values listed below).

METER_COLOR_OFF *value*

Sets the color off when using three color segments (color values listed below).

Available Colors:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

1.16.2.1.3 Gauge Meter

These parameters will display an oscilating meter within the PAUSE window.



GAUGE_VISIBLE *value*

Displays an oscillating progress gauge. Values:

- ON
- OFF

GAUGE_COLOR *value*

Specifies the color of the progress gauge. Values:

- AQUA
- BLACK
- BLUE
- CREAM
- DARK GRAY
- FUCHSIA
- GRAY
- GREEN
- LIME
- LIGHT GRAY
- MAROON
- MEDIUM GRAY
- MINT GREEN
- NAVY
- OLIVE
- PURPLE
- RED
- SILVER
- SKY BLUE
- TEAL
- WHITE
- YELLOW

GAUGE_INTERVAL *value*

Specifies the speed interval of the oscillating progress gauge. Values:

10-100

-- PAUSE Command Example with GAUGE Progress Bar

```
PAUSE 3 USING 'Calculating ... Please Stand By ...' +  
CAPTION '                Pause 3 with Gauge ' +  
ICON WINDOWS OPTION GAUGE_VISIBLE ON +  
|GAUGE_COLOR RED +  
|GAUGE_INTERVAL 10 +  
|MESSAGE_FONT_NAME VERDANA +  
|MESSAGE_FONT_SIZE 10 +  
|MESSAGE_FONT_COLOR WHITE +  
|THEMENAME Steel Blue
```

1.16.3 PLUGINS

Use the PLUGINS command to all upon a file or internal library containing data used to alter, enhance, or extend the operation of R:BASE. R:BASE Plugins come in different versions specific to a particular version of R:BASE and are made available through different means. Some Plugins are included with the purchase of R:BASE. Other Plugins are included with [R:BASE Technologies, Inc. add-on products](#) such as R:Charts, RBZip, R:PDFMerge, and R:Fax. There is also an exclusive collection of plugins within the [R:BASE Plugin Power Pack](#). Others are supplied through the attendance of [R:BASE Training](#), the Developers' Conference Package, and others are distributed freely at the [R:BASE Community](#) Home

Page.

PLUGINS filename [parameters]

Options

filename

The plugin file name (extension for an R:BASE Plugin must be **.RBL**)

parameters

Parameters which are used internally by the plugin

Plugins in General

Many software products now used a technology called "PLUGIN". This technology is based on API, which is declared by the program creator. A sample of products which use plugins includes: WinAMP, Adobe Photoshop, ACDSee, etc. It means that the computer programmers can enhance and improve product possibilities using programming languages such as C/C++, Pascal, Basic.

Plugins can be the following type:

- **100** – General type of plugin, which appear in the R:BASE Main Menu under "Utilities" > "Plugins".
- **101** – These plugins are only used with the PLUGIN command, but return a value to R:BASE.

1.16.3.1 GUI Specific

The graphic user interface (GUI) Plugins will display a dialog window that accepts keyboard input and mouse clicks. These Plugins appear in the R:BASE Main Menu under "Utilities" > "Plugins".

1.16.3.1.1 R:Dialog Builder Plugin

The R:Dialog Builder Plugin is a syntax builder for the [DIALOG](#) command. It appears in the R:BASE for Windows Menu Bar under "Utilities" > "Plugins".

Using this plugin you can easily create dialog windows to prompt end users for information within your R:BASE application.

R:Dialog Builder Plugin

General Information Options Syntax

Message Text

Type Message Here ...

Variables

Response: vResponse End Key: vEndKey

☒ Lines 1 ☐ Password

Caption

Type Caption Here ...

PAUSE Icons

	☐ Application		☐ Help		☐ Stop
	☐ Attention		☐ Information		☐ Warning
	☐ Confirmation		☐ Question		☐ Windows
	☐ Error		☐ Serious	☒ No Icon	

Test Build Close

R:Dialog Builder Plugin

General Information Options Syntax

☒ Message

Font Color: Default

Background Color: Button Face

Font Name: MS Sans Serif Size: 10

☐ Bold ☐ Italic ☐ Underline ☐ Strikeout

☒ Buttons

OK Caption: &OK Cancel Caption: &Cancel

OK Button Color: Defal Cancel Button Color: Defal

OK Button Font Color: Defal Cancel Button Font Color: Defal

Background Color: Buttoi ☐ Show Glyph

☒ Window

☐ No Caption ☐ Small Caption ☒ Normal Caption

Background Color: Button Face

☐ Transparency

255

Icon File Name

☐ Use

☒ Other

Theme Name: Aqua

Test Build Close



1.16.3.1.2 R:Dialog with Buttons Builder Plugin

The R:Dialog with Buttons Builder Plugin is a syntax builder for the [DIALOG](#) command. It appears in the R:BASE for Windows Menu Bar under "Utilities" > "Plugins".

Using this plugin you can easily create dialog windows with buttons for end users to select within your R:BASE application.



R:Dialog with Buttons Builder Plugin

General Information Options Syntax

☒ Message

Font Color: Default

Background Color: Button Face

Font Name: MS Sans Serif Size: 10

☐ Bold ☐ Italic ☐ Underline ☐ Strikeout

☒ Window

☐ No Caption ☐ Small Caption ☒ Normal Caption

Background Color: Button Face

☒ Buttons

Yes Button Caption: &Yes No Button Caption: &No

Yes Button Color: Default

Yes Button Font Color: Default

No Button Color: Default

No Button Font Color: Default

☐ Transparency

255

Icon File Name

☐ Use

☒ Other

Theme Name: Aqua

Test Build Close



1.16.3.1.3 R:Pause Builder Plugin

The R:Pause Builder Plugin is a syntax builder for the [PAUSE](#) command. It appears in the R:BASE for Windows Menu Bar under "Utilities" > "Plugins".

Using this Plugin you can easily create message windows to display information within your R:BASE application.

R:Pause Builder Plugin

General Information Options Syntax

PAUSE Type

☐ 1 ☒ 2 ☐ 3 ☐ 4

Message Text

Message Text

Caption

Caption Text

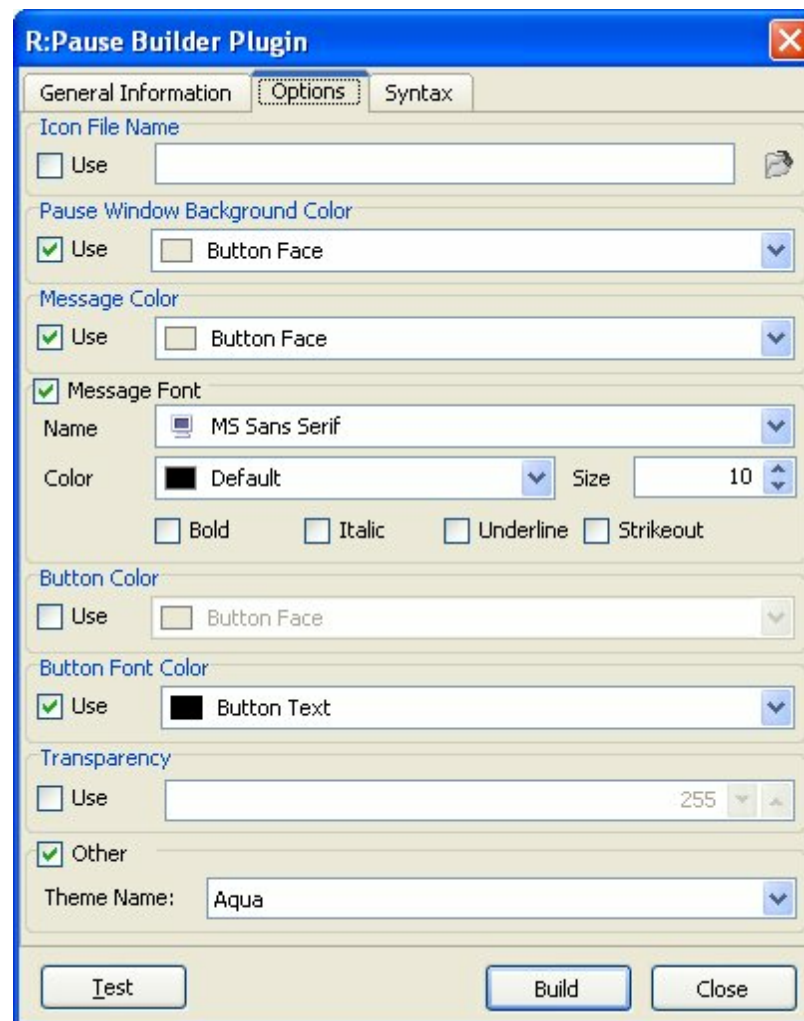
PAUSE Icons

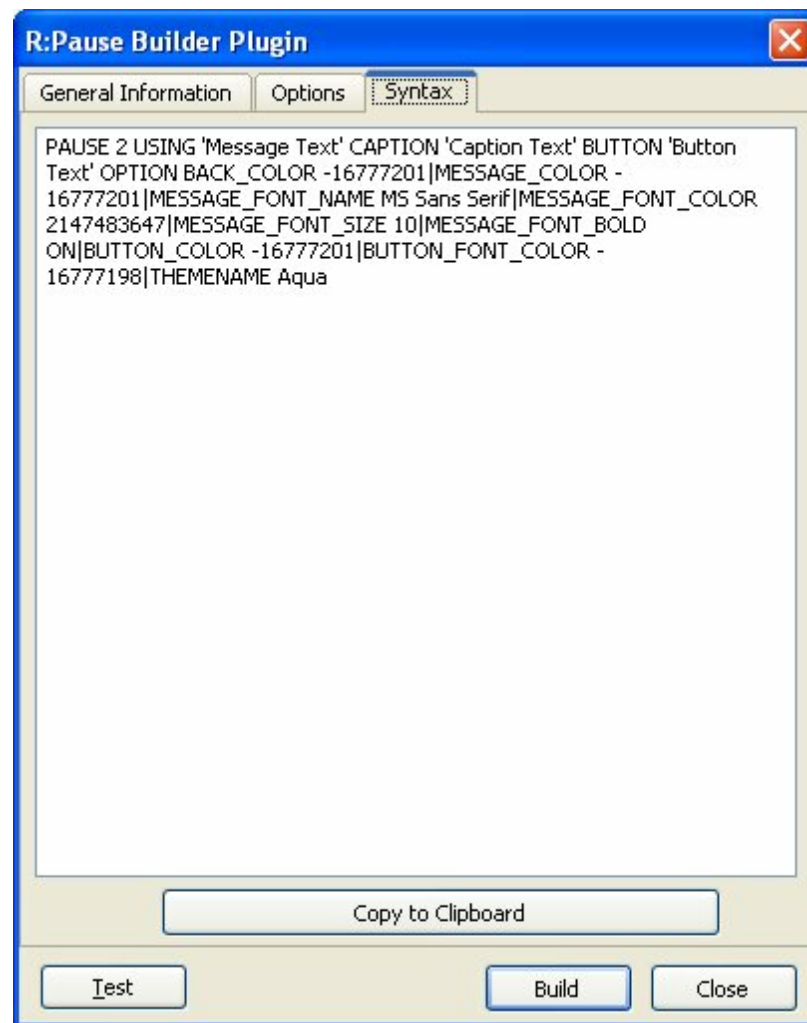
	☐ Application		☐ Help		☐ Stop
	☐ Attention		☐ Information		☐ Warning
	☐ Confirmation		☐ Question		☐ Windows
	☐ Error		☐ Serious	☒ No Icon	

Button Text

Button Text

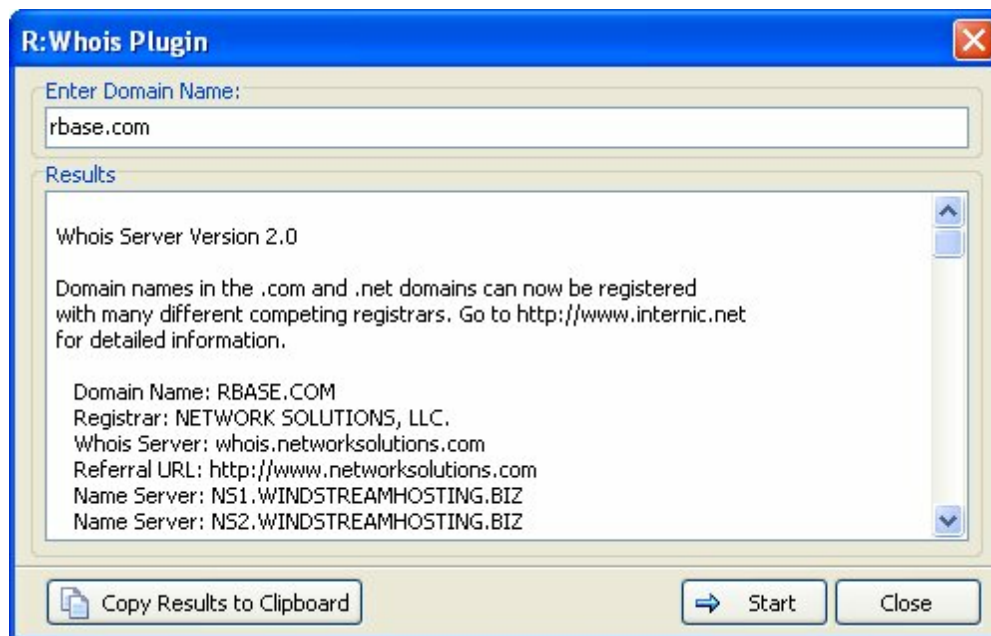
Test Build Close





1.16.3.1.4 R:Whois Plugin

The R:Whois Plugin will find and display domain name registration information and additional Web site data.



1.16.3.1.5 R:Choose Builder Plugin

The R:Choose Builder Plugin is a syntax builder for the [CHOOSE](#) command. It appears in the R:BASE for Windows Menu Bar under "Utilities" > "Plugins".

Using this plugin you can easily create choose windows to prompt end users for information within your R:BASE application.

R:Choose Builder Plugin

General Information

Options

Syntax

CHOOSE type

#COLUMNS in Table/View

Tables/Views

Table/View Name

BonusRate

Component

CompUsed

Contact

ContactCallNotes

Customer

Check

☒ None

☐ CHKBOX

☐ CHKSORT

Choices: 0

Lines: 5

☐ Formatted

Caption/Title

Caption: CHOOSE Caption Here...

Title: CHOOSE TITLE Here...

Variable

vChoose

Test

Build

Close

R:Choose Builder Plugin

General Information Options Syntax

☒ List Options

Font Color: AaBbYyZz Background Color: Button Face

☒ Title Options

Font Color: AaBbYyZz Background Color: Button Face

☒ Window

☐ No Caption ☐ Small Caption ☒ Normal Caption

☐ Show "Select All" Button ☐ Single Click

☐ Transparency 255

☒ Buttons

OK Caption: OK Background Color: Button Face

Cancel Caption: Cancel ☐ Show Glyph

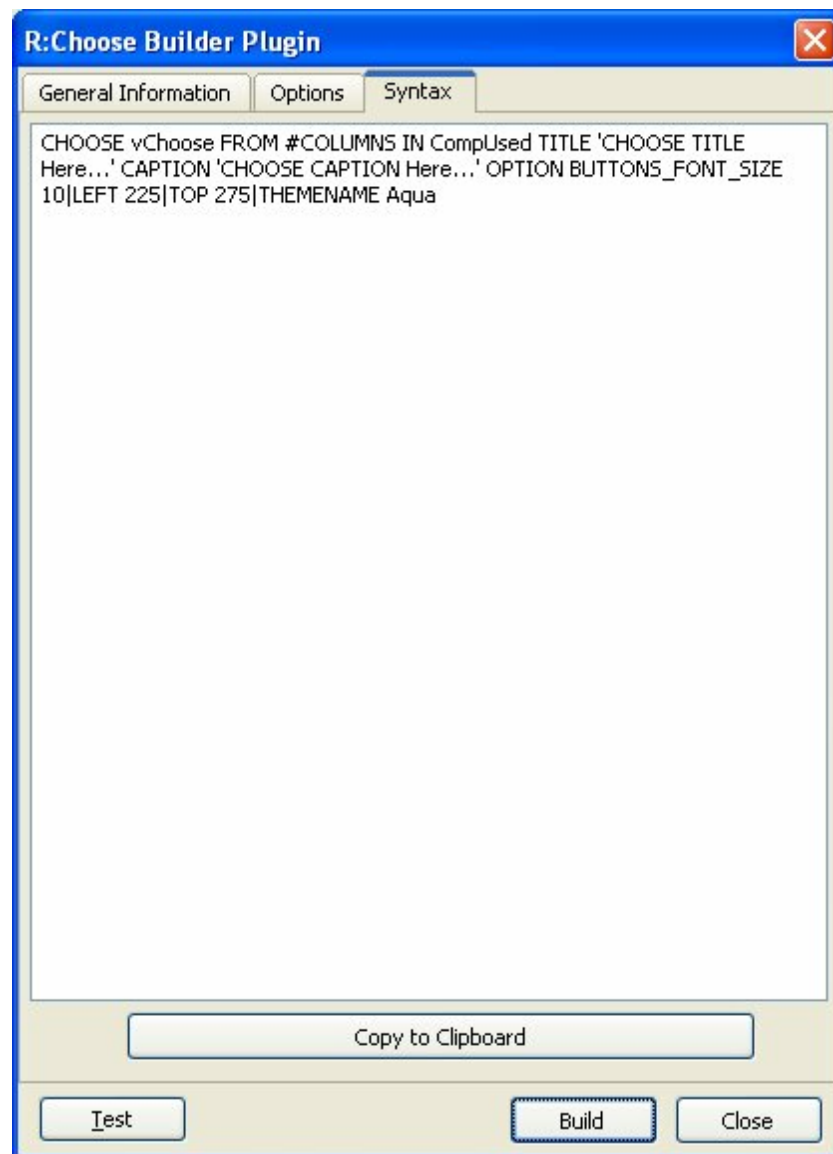
Font Size: 8 Font Color: Button Text

☒ Other

Left: 225 Top: 275

Theme Name: Aqua

Test Build Close



1.16.3.1.6 R:MAPI Mail Plugin

The R:MAPI Mail Plugin is a plugin to send a message through your email client program. It appears in the R:BASE for Windows Menu Bar under "Utilities" > "Plugins".

R:MAPI Mail Plugin

Account Settings

Account Name: support

Password: *****

Address Settings

To: support@rbase.com

Cc:

Bcc:

Subject: Additional Information

Message

This is the message body.

Attachments

C:\RBTI\RBG90_64\109ReasonstoUpgradeToRBASEeXtreme.

+ Attach File(s)

- Remove Attachment

☒ Send Directly ☐ Resolve Names

Send Cancel

1.16.3.2 Command Line Specific

The command line specific plugins are only used with the PLUGIN command and return a value to R:BASE.

1.16.3.2.1 LoadDirectoryName

The LoadDirectoryName Plugin is used to load a directory path into a variable by displaying a "Browse for Folder" dialog window for a user to browse the operating system.

Syntax:

PLUGINS LoadDirectoryName.rbl VarName|<options>

Options:

Option	Values	Description
TITLE	value	Specifies the title caption
PATH_NAME	value	Specifies the starting location
WINDOW_STATE	MAX/MIN/ NORMAL	Specifies the window state
HEIGHT	value	Specifies the window height
WIDTH	value	Specifies the window width
SHOW_BUTTONS	ON/OFF	Determines if the "Create Folder" and "Delete" buttons are

		displayed
CONTEXT MENUS	ON/OFF	Determines if a menu is displayed when user right clicks
READ_ONLY	ON/OFF	Determines if the folders names can be altered
INCLUDE_NON_FOLDERS	ON/OFF	Determines if only folders will be displayed
OLE_DRAG	ON/OFF	Supports drag and drop
OLE_DROP	ON/OFF	Supports drag and drop
CREATE_FOLDER_ICON	ON/OFF	Determines if the "Create Folder" icon is displayed on the button
DELETE_FOLDER_ICON	ON/OFF	Determines if the "Delete" icon is displayed on the button
VIRTUAL FOLDERS	ON/OFF	Determines if virtual folders can be seen
SHOW_HIDDEN	ON/OFF	Determines if hidden folders can be seen

Where:

VarName is the variable to return the text value of the selected directory

Notes:

- Each option must be separated by pipe "|" symbol.
- A value of '[Esc]' will be returned if the [Cancel] button or Close Window [x] button of the folder selection dialog is selected.

Example:

```

PLUGINS LoadDirectoryName.rbl vDirName+
|PATH_NAME C:\RBT\ +
|SHOW_BUTTONSON+
|CREATE_FOLDER_ICON OFF+
|DELETE_FOLDER_ICON OFF+
|WIDTH 500+
|HEIGHT 400+
|READ_ONLY ON+
|INCLUDE_NON_FOLDERS OFF+
|SHOW_HIDDEN OFF

```

1.16.3.2.2 LoadFileName

The LoadFileName Plugin is used to load a file name into a variable by displaying a dialog window for the user to browse the operating system.

Syntax:

```
PLUGINS LoadFileName.rbl VarName|<options>
```

Options:

Option	Value	Description
FULLPATH	ON/OFF	Determines if the full path for the selected file is returned with the file name
TITLE	value	Specifies the title caption
FILTER	value	Filters the display files (Example: Text Files (*.txt)*.txt)
INITIAL_DIR	value	Specifies the initial directory when launched
NO_CHANGE_DIR	ON/OFF	Determines if the directory changes when the file is selected
MULTISELECT	ON/OFF	Determines if multiple files can be selected
SHOW_HIDDEN	ON/OFF	Determines if hidden files can be seen
NO_LONG_NAMES	ON/OFF	Determines if files not using the 8.3 naming convention are

		displayed. This is only supported with the OLD_STYLE parameter.
NO_NETWORK_BUTTON	ON/OFF	Determines if the network button is displayed. This is only supported with the OLD_STYLE parameter.
HIDE_READ_ONLY	ON/OFF	Determines if the "Open as Read Only" check box is displayed
NO_DEREFERENCE_LINKS	ON/OFF	Determines if shortcut links are valid or are de-referenced
DONT_ADD_TO_RECENT	ON/OFF	Determines if file selection is added to recently opened documents
ENABLE_SIZING	ON/OFF	Determines if the dialog window can be resized
OLD_STYLE	ON/OFF	Determines if the "old style" window is displayed
NO_VALIDATE	ON/OFF	Determines if validation for invalid characters is used

Where:

VarName is the variable name to return the text value of the selected file

Notes:

- Each option must be separated by pipe "|" symbol.
- A value of '[Esc]' will be returned if the [Cancel] button or Close Window [x] button of the folder selection dialog is selected.

Example:

```

PLUGINS LoadFileName.rbl vFileName+
|INITIAL_DIR C:\RBT\|+
|SHOW_BUTTONSON+
|CREATE_FOLDER_ICON OFF+
|DELETE_FOLDER_ICON OFF+
|WIDTH 500+
|HEIGHT 400+
|FILTER PDF Files (*.pdf) #*.pdf+
|HIDE_READ_ONLY ON+
|INCLUDE_NON_FOLDERS OFF+
|SHOW_HIDDEN OFF

```

1.16.3.2.3 LoadFileNamePlus

The LoadFileNamePlus Plugin is used to load a file name into a variable by displaying a dialog window for the user to browse the operating system. The LoadFileNamePlus Plugin was added with additional options to load a file name.

Syntax:

```
PLUGINS LoadFileNamePlus.rbl VarName|<options>
```

Options:

Option	Value	Description
FULLPATH	ON/OFF	Determines if the full path for the selected file is returned with the file name
TITLE	value	Specifies the title caption
FILTER	value	Filters the display files (Example: Text Files (*.txt) #*.txt)
INITIAL_DIR	value	Specifies the initial directory when launched
NO_CHANGE_DIR	ON/OFF	Determines if the directory changes when the file is selected
MULTISELECT	ON/OFF	Determines if multiple files can be selected
SHOW_HIDDEN	ON/OFF	Determines if hidden files can be seen
NO_LONG_NAMES	ON/OFF	Determines if files not using the 8.3 naming convention are

		displayed. This is only supported with the OLD_STYLE parameter.
NO_NETWORK_BUTTON	ON/OFF	Determines if the network button is displayed. This is only supported with the OLD_STYLE parameter.
HIDE_READ_ONLY	ON/OFF	Determines if the "Open as Read Only" check box is displayed
NO_DEREFERENCE_LINKS	ON/OFF	Determines if shortcut links are valid or are de-referenced
DONT_ADD_TO_RECENT	ON/OFF	Determines if file selection is added to recently opened documents
ENABLE_SIZING	ON/OFF	Determines if the dialog window can be resized
OLD_STYLE	ON/OFF	Determines if the "old style" window is displayed
NO_VALIDATE	ON/OFF	Determines if validation for invalid characters is used
VIEW_MODE	THUMBNAI LS TILES ICONS (Default) LIST DETAILS	Determines the view mode for the dialog window

Where:

VarName is the variable name to return the text value of the selected file

Notes:

- Each option must be separated by pipe "|" symbol.
- A value of '[Esc]' will be returned if the [Cancel] button or Close Window [x] button of the folder selection dialog is selected.

Example:

```

PLUGINS LoadFileNamePlus.rbl vFileName+
|INITIAL_DIR C:\RBT\|+
|SHOW_BUTTONS ON+
|CREATE_FOLDER_ICON OFF+
|DELETE_FOLDER_ICON OFF+
|WIDTH 500+
|HEIGHT 400+
|FILTER PDF Files (*.pdf)#*.pdf+
|HIDE_READ_ONLY ON+
|INCLUDE_NON_FOLDERS OFF+
|SHOW_HIDDEN OFF+
|VIEW_MODE LIST

```

1.16.3.2.4 LoadFileNamePlusPlus

The LoadFileNamePlusPlus Plugin is used to load a file name into a variable by displaying a dialog window for the user to browse the operating system. The LoadFileNamePlusPlus Plugin was added with additional options to load a file name.

Syntax:

```
PLUGINS LoadFileNamePlusPlus.rbl VarName|<options>
```

Options:

Option	Value	Description
FULLPATH	ON/OFF	Determines if the full path for the selected file is returned with

		the file name
TITLE	value	Specifies the title caption
FILTER	value	Filters the display files (Example: Text Files (*.txt)*.txt)
INITIAL_DIR	value	Specifies the initial directory when launched
NO_CHANGE_DIR	ON/OFF	Determines if the directory changes when the file is selected
MULTISELECT	ON/OFF	Determines if multiple files can be selected
SHOW_HIDDEN	ON/OFF	Determines if hidden files can be seen
NO_LONG_NAMES	ON/OFF	Determines if files not using the 8.3 naming convention are displayed. This is only supported with the OLD_STYLE parameter.
NO_NETWORK_BUTTON	ON/OFF	Determines if the network button is displayed. This is only supported with the OLD_STYLE parameter.
HIDE_READ_ONLY	ON/OFF	Determines if the "Open as Read Only" check box is displayed
NO_DEREFERENCE_LINKS	ON/OFF	Determines if shortcut links are valid or are de-referenced
DONT_ADD_TO_RECENT	ON/OFF	Determines if file selection is added to recently opened documents
ENABLE_SIZING	ON/OFF	Determines if the dialog window can be resized
OLD_STYLE	ON/OFF	Determines if the "old style" window is displayed
NO_VALIDATE	ON/OFF	Determines if validation for invalid characters is used
VIEW_MODE	THUMBNAILS TILES ICONS (Default) LIST DETAILS	Determines the view mode for the dialog window
OK_BUTTON_CAPTION	value	Specifies the caption for the "OK" button
CANCEL_BUTTON_CAPTION	value	Specifies the caption for the "Cancel" button

Where:

VarName is the variable name to return the text value of the selected file

Notes:

- Each option must be separated by pipe "|" symbol.
- A value of '[Esc]' will be returned if the [Cancel] button or Close Window [x] button of the folder selection dialog is selected.

Example:

```

PLUGINS LoadFileNamePlusPlus.rbl vFileName+
|INITIAL_DIR C:\RBT\|+
|SHOW_BUTTONS ON+
|CREATE_FOLDER_ICON OFF+
|DELETE_FOLDER_ICON OFF+
|WIDTH 500+
|HEIGHT 400+
|FILTER PDF Files (*.pdf)*.pdf+
|HIDE_READ_ONLY ON+
|INCLUDE_NON_FOLDERS OFF+
|SHOW_HIDDEN OFF+
|VIEW_MODE LIST+
|OK_BUTTON_CAPTION Load

```

1.16.3.2.5 RCalculator

The RCalculator Plugin is used to launch a calculator to perform any of the standard operations for which you would normally use a handheld calculator. It does not appear in the R:BASE for Windows Menu.

Syntax:

```
PLUGIN 'RCalculator.rbl' 'vResult'|THEMENAME Razzmatazz
```

Where:

vResult is the variable to return the value of the calculated result

Razzmatazz will display the calculator using the theme below

Notes:

The returned variable value will be the correct data type of the calculated result (INTEGER, DOUBLE, REAL).

1.16.3.3 Create Custom Plugins

Using a programming language such as C/C++, Pascal, Delphi, Basic, etc., you can create your own R:BASE Plugins.

These plugins are DLL files with 4 exported functions:

GetRBPluginName

Syntax:

```
procedure GetRBPluginName(Value : PChar); stdcall;
```

Value – result value which you can see in Plugins menu.

A sample of implementing this function in RBL:

```
procedure GetRBPluginName(Value : PChar); stdcall;  
var Name : string;  
begin  
    Name := 'My Sample Plugin';  
    StrLCopy(Value, Pchar(Name), Length(Name));  
end;
```

GetRBPluginType

Syntax:

```
function GetRBPluginType : integer; stdcall;
```

This function returns the type of a plugin to R:BASE.

A sample of implementing this function in RBL:

```
function GetRBPluginType : integer; stdcall;  
begin  
    Result := 100;  
end;
```

GetBitmap

Syntax:

```
function GetGetBitmap : THandle; stdcall;
```

This function returns to R:BASE a handle for a Bitmap. This is the bitmap you can see in the plugins menu.

A sample of implementing this function in RBL:

```
function GetGetBitmap : THandle; stdcall;  
begin  
  Bitmap.LoadFromResourceName(Hinstance, 'MAINBMP');  
  Result := Bitmap.Handle;  
end;
```

StartRBPlugin

Syntax:

```
procedure StartRBPlugin(Params : PChar); stdcall;
```

This function starts your plugin.

101 – Same as in type 100 except:

```
function StartRBPlugin(Params : PChar) : PChar; stdcall;
```

In the result of this function, you need to put a value which will go to the R:BASE variable.

LoadDirectoryName Plugin schema:

LoadDirectoryName	TITLE	[value]
	PATH_NAME - value	[value]
WINDOW_STATE		[MAX MIN NORMAL]
HEIGHT - value		
WIDTH - value		
DELETE_BUTTON		[ON OFF]
CONTEXT_MENUS		[ON OFF]
READ_ONLY		[ON OFF]
INCLUDE_NON_FOLDERS		[ON OFF]
OLE_DRAG		[ON OFF]
OLE_DROP		[ON OFF]
CREATE_FOLDER_ICON		[ON OFF]
DELETE_FOLDER_ICON		[ON OFF]
VIRTUAL_FOLDERS		[ON OFF]
SHOW_HIDDEN		[ON OFF]

Example:

```

////////////////////////////////////
// Example to presentation:
//
//
// "Designing and Implementing R:BASE Plugins"
//
//
// Author: Aleksey V. Chuyasov.
//
//
// Load user-selected folder name to R:BASE variable
//
////////////////////////////////////

```

library LoadDirectoryName;

uses

```

    SysUtils,
    RzShellDialogs,
    Forms,
    Classes;

```

```
{ $E rbl }
```

```

{$R *.res}

procedure GetRBPluginName(ResName : PChar); stdcall;
var Name : string;
begin
    Name := 'Get File Name';
    StrLCopy(ResName, PChar(Name), Length(Name));
end;

function GetRBPluginType : Integer; stdcall;
begin
    Result := 101;
end;

function StartRBPlugin(Params : PChar) : PChar; stdcall;

    procedure ParseParams(Strings : TStringList);
    var st, s : string;
        i : integer;
    begin
        Strings.Clear;
        st := string(Params);
        s := '';
        for i := 1 to Length(Params) do
            begin
                if st[i] <> '|' then
                    begin
                        s := s + st[i];
                        if i = Length(Params) then Strings.Add(s);
                    end else
                        begin
                            Strings.Add(s);
                            s := '';
                        end;
                    end;
            end;
        end;

var FolderDLG : TRzSelectFolderDialog;
    FileName : string;
    st, st2 : string;
    List : TStringList;
    CurrDir : string;
    i, j : integer;
    fullpath : boolean;
    Ans : array[0..65535] of char;
begin
    FillChar(Ans, 65536, #0);
    CurrDir := GetCurrentDir;
    try
        FolderDLG := TRzSelectFolderDialog.Create(nil);
    try
        List := TStringList.Create;
    try
        ParseParams(List);
        for i := 0 to List.Count - 1 do
            begin
                st := trim(UpperCase(List[i]));
                st2 := trim(List[i]);
                if pos('TITLE', st) = 1 then
                    begin

```

```

Delete(st2, 1, Length('TITLE'));
st := trim(st2);
FolderDLG.Title := st;
end else
if pos('PATH_NAME', st) = 1 then
begin
Delete(st2, 1, Length('PATH_NAME'));
st := trim(st2);
FolderDLG.SelectedPathName := st;
end else
if pos('WINDOW_STATE', st) = 1 then
begin
Delete(st, 1, Length('WINDOW_STATE'));
st := trim(st);
if st = 'MAX' then
FolderDLG.FormWindowState := wsMaximized else
if st = 'MIN' then
FolderDLG.FormWindowState := wsMinimized else
if st = 'NORMAL' then
FolderDLG.FormWindowState := wsNormal;
end else
if pos('HEIGHT', st) = 1 then
begin
Delete(st, 1, Length('HEIGHT'));
st := trim(st);
try
FolderDLG.FormHeight := StrToInt(st);
except
//if user typing non-integer value here.
end;
end else
if pos('WIDTH', st) = 1 then
begin
Delete(st, 1, Length('WIDTH'));
st := trim(st);
try
FolderDLG.FormWidth := StrToInt(st);
except
//if user typing non-integer value here.
end;
end else
if pos('DELETE_BUTTON', st) = 1 then
begin
Delete(st, 1, Length('DELETE_BUTTON'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
FolderDLG.Options + [sfdoCreateDeleteButtons] else
if st = 'OFF' then FolderDLG.Options :=
FolderDLG.Options - [sfdoCreateDeleteButtons];
end else
if pos('CONTEXT_MENUS', st) = 1 then
begin
Delete(st, 1, Length('CONTEXT_MENUS'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
FolderDLG.Options + [sfdoContextMenus] else
if st = 'OFF' then FolderDLG.Options :=
FolderDLG.Options - [sfdoContextMenus];
end else
if pos('READ_ONLY', st) = 1 then
begin

```

```

Delete(st, 1, Length('READ_ONLY'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoReadOnly] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoReadOnly];
end else
if pos('INCLUDE_NON_FOLDERS', st) = 1 then
begin
Delete(st, 1, Length('INCLUDE_NON_FOLDERS'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoIncludeNonFolders] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoIncludeNonFolders];
end else
if pos('OLE_DRAG', st) = 1 then
begin
Delete(st, 1, Length('OLE_DRAG'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoOleDrag] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoOleDrag];
end else
if pos('OLE_DROP', st) = 1 then
begin
Delete(st, 1, Length('OLE_DROP'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoOleDrop] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoOleDrop];
end else
if pos('CREATE_FOLDER_ICON', st) = 1 then
begin
Delete(st, 1, Length('CREATE_FOLDER_ICON'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoCreateFolderIcon] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoCreateFolderIcon];
end else
if pos('DELETE_FOLDER_ICON', st) = 1 then
begin
Delete(st, 1, Length('DELETE_FOLDER_ICON'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoDeleteFolderIcon] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoDeleteFolderIcon];
end else
if pos('VIRTUAL_FOLDERS', st) = 1 then
begin
Delete(st, 1, Length('VIRTUAL_FOLDERS'));
st := trim(st);
if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoVirtualFolders] else
if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoVirtualFolders];
end else

```

```
if pos('SHOW_HIDDEN', st) = 1 then
begin
  Delete(st, 1, Length('SHOW_HIDDEN'));
  st := trim(st);
  if st = 'ON' then FolderDLG.Options :=
    FolderDLG.Options + [sfdoShowHidden] else
  if st = 'OFF' then FolderDLG.Options :=
    FolderDLG.Options - [sfdoShowHidden];
  end;
end;
if FolderDLG.Execute then
begin
  FileName := FolderDLG.SelectedPathName;
end;
finally
  List.Free;
end;
finally
  FolderDLG.Free;
end;
finally
  StrPCopy(Ans, FileName);
  Result := Ans;
  SetCurrentDir(CurrDir);
end;
end;

Exports GetRBPluginName, GetRBPluginType, StartRBPlugin;
begin

end.
```

SYNTAX:

```
PLUGIN LoadDirectoryName.rbl vFolderName|CREATE_FOLDER_ICON ON|
DELETE_FOLDER_ICON ON
```

1.16.4 PREVROW

Use the PREVROW command in an entry/exit procedure (EEP) to move to the previous row.

PREVROW

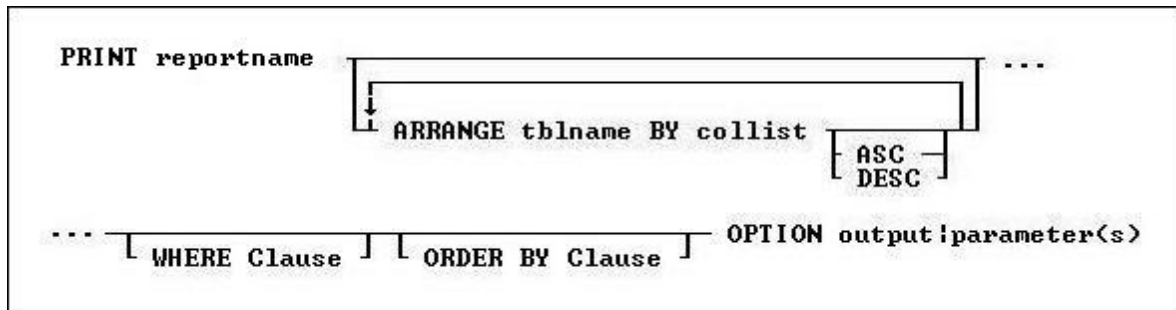
1.16.5 PREVTAB

Use the PREVTAB command in an entry/exit procedure (EEP) to move to the previous table.

PREVTAB

1.16.6 PRINT

Use the PRINT command to send a report to the screen, printer, or file.



Options

rptname

Specifies the name of the report you want to print.

ARRANGE tblname BY collist

Sorts the rows displayed by a report's lower-level tables. Each lower-level table in the report that you want to sort requires its own ARRANGE clause. Now you can specify up to twenty-five tables to ARRANGE and up to five columns in each table on which R:BASE will sort.

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

OPTION

specifies output options of the report along with additional parameters for each output such as:

- [SCREEN](#)
- [PRINTER](#)
- [BMP](#)
- [DIF](#)
- [EMF](#)
- [ETXT](#) (Report Emulation Text)
- [GIF](#)
- [HTML](#)
- [JPG](#)
- [LOTUS](#)
- [PDF](#)
- [PNG](#)
- [QPRO](#)
- [RTF](#)
- [SVG](#)
- [SYLK](#)
- [TIFF](#)
- [TXT](#)
- [WMF](#)
- [XHTML](#)
- [XLS](#)

Working with Breakpoints

When a report has breakpoints, R:BASE automatically sorts by the defined breakpoint columns. You can sort additional columns other than breakpoint columns if the breakpoint columns are listed in the ORDER BY clause first. If an ORDER BY clause that overrides the breakpoint sort is included in the PRINT command, the report might not produce the results defined by the breakpoints.

1.16.6.1 SCREEN

PRINT reportname **ARRANGE** clause **WHERE** clause **.. ORDER BY** clause **.. OPTION SCREEN**

Note: Each additional **OPTION** parameter must be separated by the pipe | symbol.

OPTION Parameters:

WINDOW_STATE specifies the state of preview window. By default, the window state is normalized.

Values:

MAXIMIZED
MINIMIZED
NORMAL

TOOLBAR_VISIBLE specifies the option to display preview toolbar with all buttons such as, Print, Whole Page, Page Width, 100%, Zoom Level, First, Page Previous Page, Next Page, Last Page and Cancel/Close.

Values:

TRUE
FALSE



BUTTONBAR_VISIBLE specifies the option to display button bar with Whole Page, Page Width and 100%. If set to FALSE, all three buttons (Whole Page, Page Width and 100%) will not be visible.

Values:

TRUE
FALSE

BTNPRINT_VISIBLE specifies the option to display Printer button on the button toolbar.

Values:

TRUE
FALSE

BTNWHOLEPAGE_VISIBLE specifies the option to display Whole Page button on the button toolbar.

Values:

TRUE
FALSE

BTNPAGEWIDTH_VISIBLE specifies the option to display Page Width button on the button toolbar.

Values:

TRUE
FALSE

BTNP100PERCENT_VISIBLE specifies the option to display 100% button on the button toolbar.

Values:

TRUE
FALSE

ZOOMPERCENT_VISIBLE specifies the option to display Zoom Level button on the button toolbar.

Values:

TRUE
FALSE

BTNFIRST_VISIBLE specifies the option to display First Page button on the button toolbar.

Values:

TRUE
FALSE

BTNPRIOR_VISIBLE specifies the option to display Previous Page button on the button toolbar.

Values:

TRUE
FALSE

PAGENO_VISIBLE specifies the option to display Page Number button on the button toolbar.

Values:

TRUE
FALSE

BTNNEXT_VISIBLE specifies the option to display Next Page button on the button toolbar.

Values:

TRUE
FALSE

BTNLAST_VISIBLE specifies the option to display Last Page button on the button toolbar.

Values:

TRUE
FALSE

BTNCANCEL_VISIBLE specifies the option to display Cancel/Close button on the button toolbar.

Values:

TRUE
FALSE

ZOOM_TYPE specifies the zoom type, such as PAGE_WIDTH, WHOLE_PAGE, 100_PERCENT or PERCENTAGE.

Values:

PAGE_WIDTH
WHOLE_PAGE
100_PERCENT
PERCENTAGE (to be used when specifying the ZOOMPERCENTAGE option)

ZOOMPERCENTAGE specifies the zoom level percentage, such as 70, 100, 120, 200, 250.

Note:

You must use the ZOOM_TYPE value as PERCENTAGE when specifying the ZOOMPERCENTAGE value.
See examples below for a more practical use of this option.

PREVIEW_CAPTION specifies the caption for the preview window

MDI specifies the option to display the preview window in a modeless window, to access other windows without closing it first.

LEFT *nnn* - specifies the left coordinate of the print preview window

TOP *nnn* - specifies the top coordinate of the print preview window

WIDTH *nnn* - specifies the width of the print preview window

HEIGHT*nnn* - specifies the height of the print preview window

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

THEMENAME - specifies one of 86 pre-defined Themes, or a custom Theme loaded into R:BASE. See Themes.

Examples:

Example 01 (Typical Print Preview on SCREEN with Toolbar):

PRINT Invoice **WHERE** TransID = 1002 **OPTION** SCREEN

Example 02 (Typical Print Preview on SCREEN with maximized preview window):

PRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED

Example 03 (Typical Print Preview on SCREEN with zoom type as page width):

PRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**ZOOM_TYPE** PAGE_WIDTH

Example 04 (Typical Print Preview on SCREEN with zoom type as whole page):

PRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**ZOOM_TYPE** WHOLE_PAGE

Example 05 (Typical Print Preview on SCREEN with zoom type as 100%):

PRINT Invoice **WHERE** TransID = 1002 +
OPTION SCREEN|**ZOOM_TYPE** 100_PERCENT

Example 06 (Typical Print Preview on SCREEN with zoom type as 90%):

PRINT Invoice **WHERE** CustID = 1002 +
OPTION SCREEN|**WINDOW_STATE** MAXIMIZED +
|**ZOOM_TYPE** PERCENTAGE|**ZOOMPERCENT** 90

Example 07 (Print Preview Tool Bar without Printer Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNPRINT_VISIBLE FALSE
```

Example 07 (Print Preview Tool Bar without Whole Page Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNWHOLEPAGE_VISIBLE FALSE
```

Example 07 (Print Preview Tool Bar without Page Width Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNPAGEWIDTH_VISIBLE FALSE
```

Example 08 (Print Preview Tool Bar without 100% Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTN100PERCENT_VISIBLE FALSE
```

Example 09 (Print Preview Tool Bar without Zoom Percentage Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|ZOOMPERCENT_VISIBLE FALSE
```

Example 10 (Print Preview Tool Bar without First Page Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNFIRST_VISIBLE FALSE
```

Example 11 (Print Preview Tool Bar without Previous Page Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNPRIOR_VISIBLE FALSE
```

Example 12 (Print Preview Tool Bar without Page Number Icon):

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|PAGENO_VISIBLE FALSE
```

Example 13 (Print Preview Tool Bar without Next Page Icon)

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNNEXT_VISIBLE FALSE
```

Example 14 (Print Preview Tool Bar without Last Page Icon)

```
PRINT Invoice WHERE TransID = 1002 +  
OPTION SCREEN|WINDOW_STATE MAXIMIZED +  
|BTNLAST_VISIBLE FALSE
```

Example 15 (Print Preview Tool Bar without Cancel/Close Icon)

```
PRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN|WINDOW_STATE MAXIMIZED +
|BTNCANCEL_VISIBLE FALSE
```

Example 16 (Print Preview without the entire Tool Bar)

```
PRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN|WINDOW_STATE MAXIMIZED +
|TOOLBAR_VISIBLE FALSE
```

Example 17 (Print Preview with a MDI window using specified coordinates)

```
PRINT Invoice WHERE TransID = 1002 +
OPTION SCREEN MDI|LEFT 20|TOP 20|WIDTH 800|HEIGHT 500
```

1.16.6.2 PRINTER

PRINT reportname **ARRANGE** clause **WHERE** clause .. **ORDER BY** clause .. **OPTION PRINTER**

Note: Each additional **OPTION** parameter must be separated by the pipe | symbol.

OPTION Parameters:

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

TRAY specifies the name of the bin (paper tray) containing the paper on which the report will be printed. A common use of this property is to set it to Manual Feed on mailing label reports. That way you can provide a means to load the special label paper without manually setting the printer.

COLLATION indicates whether multiple copies will print in pre-sorted order.

Values:

ON
OFF

COPIES specifies the number of copies to print. If the Collation property is set to ON, the copies will print in sorted order.

DOCUMENT_NAME When a report is printing, this is the name of the document as it should appear when listed in the Print Manager and on network header pages. This value can also be displayed in reports using the DocumentName type of the SystemVariable component.

DUPLEX This property indicates what type of two-sided printing should occur. Not all printers support duplex printing.

Values:

NONE - Duplex printing is disabled.
HORIZONTAL - Duplex printing should be performed across the page, from left to right, so the text will read as in a book
VERTICAL - Duplex printing should be performed down the page, from top to bottom, so the text will read as in tablet

ORIENTATION The value of the Orientation property determines whether a report prints vertically or

horizontally on a page.

Values:

- PORTRAIT - The print job prints vertically on the page.
- LANDSCAPE - The print job prints horizontally on the page.

PAPER_NAME The name of the paper on which the report will be printed. When you specify a value for the PAPER_NAME property, the PaperWidth and PaperHeight are automatically set to the proper values. For example, if Legal is specified, the PAPER_WIDTH will be set to 8.5" and the PAPER_HEIGHT will be set to 14" when ORIENTATION is PORTRAIT (if the ORIENTATION is LANDSCAPE, the values will be reversed).

PAPER_WIDTH The value of the PAPER_WIDTH are always expressed in the units specified by the Report.

PAPER_HEIGHT The value of the PAPER_HEIGHT are always expressed in the units specified by the Report.

PRINTER_NAME The name of the printer that will be used to print the report.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

- ALL
- FIRST
- LAST
- PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

- ALL
- FIRST
- LAST
- Page List i.e. 1,4-6,10-12,20

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

Examples:

Example 01 (To print Customer List on your default Windows printer):

PRINT CustomerList **OPTION** PRINTER

Example 02 (To print Customer List on your network printer):

```
PRINT CustomerList +  
OPTION PRINTER +  
| COPIES 1 +  
| ORIENTATION PORTRAIT +  
| PRINTER_NAME \\MyServer\hp LaserJet 1320 PCL 6
```

Example 03 (To print Customer Shipping Labels on your label printer):

```
LBLPRINT CustShippingLabel WHERE CustID = 127 OPTION PRINTER +  
| COPIES 2 +  
| PRINTER_NAME DYMO LabelWriter 2 +  
| ORIENTATION LANDSCAPE
```

Example 04 (To print first page of Customer List report only)

```
PRINT CustomerList OPTION PRINTER | PAGE_SETTINGS FIRST
```

Example 05 (To print last page of Customer List Report only)

```
PRINT CustomerList OPTION PRINTER | PAGE_SETTINGS LAST
```

Example 06 (To PRINT all pages)

```
PRINT ReportName OPTION PRINTER | PAGE_SETTINGS ALL
```

Example 07 (To print pages 3, 5, and 8 of your big report)

```
PRINT ReportName OPTION PRINTER +  
| PAGE_SETTINGS PAGE_LIST +  
| PAGE_LIST_VALUES 3,5,8
```

Example 08 (To print pages 3 and 8 of your big report)

```
PRINT ReportName OPTION PRINTER +  
| PAGE_SETTINGS PAGE_LIST +  
| PAGE_LIST_VALUES 3,8
```

Example 09 (To print page 3 only)

```
PRINT ReportName OPTION PRINTER +  
| PAGE_SETTINGS PAGE_LIST +  
| PAGE_LIST_VALUES 3
```

Example 10 (To print pages 1 through 3 and 8 of your big report)

```
PRINT ReportName OPTION PRINTER +  
| PAGE_SETTINGS PAGE_LIST +  
| PAGE_LIST_VALUES 1-3,8
```

Example 11 (To print pages 3, 5, and 9 through 15 of your big report)

```
PRINT ReportName OPTION PRINTER +  
| PAGE_SETTINGS PAGE_LIST +  
| PAGE_LIST_VALUES 3,5,9-15
```

1.16.6.3 BMP

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION BMP

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the BMP file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

MONOCHROME specifies whether bmp-file produced are color or monochrome images.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the bmp-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as BMP file)

```
PRINT Invoice WHERE TransID = 1068 +
OPTION BMP +
|FILENAME Invoice1068.BMP +
|SHOW_CANCEL_DIALOG OFF +
|BACKGROUND_COLOR WHITE +
|MONOCHROME OFF +
|PIXELFORMAT 32 +
|SCALEX 1 +
|SCALE 1 +
|OPEN ON
RETURN
```

1.16.6.4 DIF

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION DIF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

SCALEX - scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY - scaling factor for the height of the exported page. Use 1 for actual height.

COLUMN_SPACING - specifies the column spacing between data. Use -1 for auto-spacing.

ROW_SPACING - specifies the row spacing between data. Use -1 for auto-spacing.

AUTO_FORMAT_CELLS - specifies if the format of the rendered text (Integer, Double etc) is detected automatically. Otherwise the content is just inserted as text without any format considerations.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL - specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

--To print Invoice as DIF file

```
PRINT Invoice WHERE TransID = 1068 +
OPTION DIF +
|FILENAME Invoice1068.DIF +
|SHOW_CANCEL_DIALOG OFF +
|OPEN OFF +
|ROW_SPACING 1
RETURN
```

1.16.6.5 EMF

PRINT reportname **ARRANGE** clause **WHERE** clause .. **ORDER BY** clause .. **OPTION** EMF

Note Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the EMF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

SCALEX scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON

OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST

PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as EMF file)

```
PRINT Invoice WHERE TransID = 1068 +  
OPTION EMF +  
| FILENAME Invoice1068.EMF +  
| SHOW_CANCEL_DIALOG OFF +  
| BACKGROUND_COLOR WHITE +  
| SCALEX 1 +  
| SCALE 1 +  
| OPEN ON  
RETURN
```

1.16.6.6 ETXT

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION ETXT

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

The **Report Emulation Text File (ETXT)** option allows you to print the exact report as plain readable text including all bands, such as Report Header, Page Header, Break Headers, Break Footers, Page Footers and Report Footers, if any.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print columnar report as Emulation Text - ETXT file)

```
PRINT ColumnarReport +
OPTION ETXT +
|FILENAME ColumnarReport.TXT +
|SHOW_CANCEL_DIALOG OFF +
|OPEN ON
RETURN
```

1.16.6.7 GIF

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION GIF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED

SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the GIF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

MONOCHROME specifies whether bmp-file produced are color or monochrome images.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the bmp-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as GIF file)

```
PRINT Invoice WHERE TransID = 1068 +
OPTION GIF +
|FILENAME Invoice1068.GIF +
|SHOW_CANCEL_DIALOG OFF +
|BACKGROUND_COLOR WHITE +
|MONOCHROME OFF +
|PIXELFORMAT 32 +
|SCALEX 1 +
|SCALE 1 +
|OPEN ON
RETURN
```

1.16.6.8 HTML

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION HTML

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the HTML file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the Rich Text objects on the report should be included in the exported file.

Values:

ON
OFF

RICHTEXT_ENCODING_TYPE specifies the way in which Rich Text contents should be encoded in exported HTML document when INCLUDE_RICHTEXT option is set to ON.

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode Rich Text contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode Rich Text contents as unformatted text. The font used when encoding Rich Text as plain text is the one set in the Font property of the Rich Text object.

Values:

IMAGE
PLAINTEXT

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as "dots per inch" to enhance the image. By increasing this value, the file size will increase. An example would be 200. The Default Value is -1.

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

INCLUDE_HYPERLINKS specifies whether hyperlinks in your report should be included in the exported file.

Values:

ON
OFF

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords Meta content.

OPTIMIZE_FOR_IE specifies whether the entire report output as HTML document should be optimized for MS Internet Explorer browser.

Values:

ON
OFF

PAGE_END_LINES indicates whether a line should be drawn at the end of each page. Setting PAGE_END_LINES to ON draws a line after each page. PAGE_END_LINES is considered only when SEP_FILE_PER_PAGE is set to OFF.

Values:

ON
OFF

OUTPUT_STYLES_TO_CSS_FILE specifies whether to output styles to a CSS (Cascade Style Sheet) file.

Values:

ON
OFF

CSSFILE specify the CSS file name if OUTPUT_STYLES_TO_CSS_FILE is set to ON.

OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE specifies whether to output scripts to a Java Script File.

Values:

ON
OFF

JAVA_SCRIPT_FILE_NAME specify the file name if the OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE is set to ON.

GENERATE_TOC specifies whether to generate a **Table of Contents** based on all report Break Header (s), if defined.

Values:

ON
OFF

TOC_FILE_NAME specify the file name for the Table of Contents, if the GENERATE_TOC option is set to ON.

IMAGE_FOLDER specify the folder path and name for HTML document images.

DEFAULT_FONT_NAME specify the default font name for HTML document.

DEFAULT_FONT_SIZE specify the default font size for HTML document.

DEFAULT_FONT_COLOR specify the default font color for HTML document.

DEFAULT_FONT_BOLD specifies whether the font is bold faced.

Values:

ON
OFF

DEFAULT_FONT_ITALIC specifies whether the font is italicized.

Values:

ON
OFF

DEFAULT_FONT_UNDERLINE specifies whether the font is underlined.

Values:

ON

OFF

DEFAULT_FONT_STRIKEOUT specifies whether the font is displayed with a horizontal line through it.

Values:

ON
OFF

SEP_FILE_PER_PAGE specifies whether to create a separate file for each page of the report.

Values:

ON
OFF

NAV_SHOW Set to ON to show a collection of links that enable easy navigation between pages of the exported HTML. The links function much like the navigator buttons in the standard preview of Report Preview: one link to jump to the first page; one to the last page; one to the next page and another to get back to the previous page. NAV_SHOW is only effective when the option for SEP_FILE_PER_PAGE is set to ON.

Values:

ON
OFF

LINK_BACK_COLOR specifies the background color of navigator. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

LINK_H_BACK_COLOR specifies the background color of navigator when the mouse is hovered over it. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

LINK_H_FORE_COLOR specifies the color of navigator when the mouse is hovered over it. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

USE_TEXT_LINKS specifies the type of the navigator links that should be encoded for a multi-file HTML output. Setting USE_TEXT_LINKS to ON uses the text provided in **LINK_TEXT_FIRST**, **LINK_TEXT_PREV**, **LINK_TEXT_NEXT** and **LINK_TEXT_LAST** as the hyperlink's caption. Setting to OFF uses Images specified by **LINK_IMG_FILE_FIRST**, **LINK_IMG_FILE_PREV**, **LINK_IMG_FILE_NEXT**, **LINK_IMG_FILE_LAST**. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

ON
OFF

LINK_TEXT_FIRST specifies the hyperlink's text caption for first page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_PREV specifies the hyperlink's text caption for previous page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_NEXT specifies the hyperlink's text caption for next page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_LAST specifies the hyperlink's text caption for last page when USE_TEXT_LINKS option is set to ON.

LINK_FONT_NAME specify the font name for navigator links.

LINK_FONT_SIZE specify the font size for navigator links.

LINK_FONT_COLOR specify the font color for navigator links.

LINK_FONT_BOLD specifies whether the font is bold faced for navigator links.

Values:

ON
OFF

LINK_FONT_ITALIC specifies whether the font is italicized for navigator links.

Values:

ON
OFF

LINK_FONT_UNDERLINE specifies whether the font is underlined for navigator links.

Values:

ON
OFF

LINK_FONT_STRIKEOUT specifies whether the font for navigator link is displayed with a horizontal line through it.

Values:

ON
OFF

LINK_IMG_FILE_FIRST specifies the hyperlink's image file for first page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_PREV specifies the hyperlink's image file for previous page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_NEXT specifies the hyperlink's image file for next page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_LAST specifies the hyperlink's image file for last page when USE_TEXT_LINKS option is set to OFF.

NAV_TYPE specifies the type of the navigator. If NAV_TYPE is set to SCREEN then the navigator will be fixed to screen, else the navigator will be fixed to page. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

PAGE
SCREEN

NAV_ORIENT specifies the orientation of the navigator. if NAV_ORIENT is set to HORIZONTAL then, a Horizontal navigator is encoded, else a Vertical navigator is encoded. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

VERTICAL
HORIZONTAL

NAV_POS specifies the position of the navigator. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

DISABLE_RIGHT_CLICK specifies whether to disable the typical speed menu options when you right

click on your mouse button.

Values:

ON
OFF

DISABLE_COPYING specifies whether to disable the option to highlight and copy the text.

Values:

ON
OFF

AUTO_MAXIMIZE_WINDOW specifies whether to automatically maximize the window when HTML document is opened.

Values:

ON
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST

PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST

Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print SalesSumAndSubTot report as HTML file)

```
PRINT SalesSumAndSubTot +
OPTION HTML +
|FILENAME SalesSumAndSubTot.htm +
|SHOW_CANCEL_DIALOG ON +
|BACKGROUND_FILE NONE +
|BACKGROUND_TYPE CENTER +
|BACKGROUND_COLOR WHITE +
|INCLUDE_LINES ON +
|INCLUDE_SHAPES ON +
|INCLUDE_RICHTEXT OFF +
|RICHTEXT_ENCODING_TYPE PLAINTEXT +
|INCLUDE_IMAGES ON +
|IMAGE_FORMAT JPG +
|JPEG_QUALITY 100 +
|IMAGE_DPI -1 +
|PIXELFORMAT 24 +
|INCLUDE_HYPERLINKS ON +
|TITLE Customer Sales Sub-Totals and Totals +
|SUBJECT ConComp Sales Summary Report +
|AUTHOR Accounting Team of ConComp +
|KEYWORDS SubTotals Totals Sales Summary +
|OPTIMIZE_FOR_IE ON +
|PAGE_END_LINES ON +
|OUTPUT_STYLES_TO_CSS_FILE ON +
|CSSFILE +
|OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE ON +
|JAVA_SCRIPT_FILE_NAME +
|GENERATE_TOC ON +
|TOC_FILE_NAME Index.htm +
|IMAGE_FOLDER Images +
|DEFAULT_FONT_NAME Verdana +
|DEFAULT_FONT_SIZE 9 +
|DEFAULT_FONT_COLOR BLACK +
|DEFAULT_FONT_BOLD OFF +
|DEFAULT_FONT_ITALIC OFF +
|DEFAULT_FONT_UNDERLINE OFF +
|DEFAULT_FONT_STRIKEOUT OFF +
|SEP_FILE_PER_PAGE ON +
|NAV_SHOW ON +
|LINK_BACK_COLOR WHITE +
|LINK_H_BACK_COLOR BLUE +
|LINK_H_FORE_COLOR WHITE +
|USE_TEXT_LINKS OFF +
|LINK_TEXT_FIRST First +
```

```

|LINK_TEXT_PREV Previous +
|LINK_TEXT_NEXT Next +
|LINK_TEXT_LAST Last +
|LINK_FONT_NAME Wingdings +
|LINK_FONT_SIZE 10 +
|LINK_FONT_COLOR BLUE +
|LINK_FONT_BOLD OFF +
|LINK_FONT_ITALIC OFF +
|LINK_FONT_UNDERLINE OFF +
|LINK_FONT_STRIKEOUT OFF +
|LINK_IMG_FILE_FIRST Images\FirstOff.gif +
|LINK_IMG_FILE_PREV Images\PrevOff.gif +
|LINK_IMG_FILE_NEXT Images\NextOff.gif +
|LINK_IMG_FILE_LAST Images\LastOff.gif +
|NAV_TYPE PAGE +
|NAV_ORIENT HORIZONTAL +
|NAV_POS BOTTOMCENTER +
|DISABLE_RIGHT_CLICK OFF +
|DISABLE_COPYING OFF +
|AUTO_MAXIMIZE_WINDOW ON +
|OPEN ON
RETURN

```

1.16.6.9 JPG

PRINT reportname **ARRANGE** clause **WHERE** clause .. **ORDER BY** clause .. **OPTION** JPG

Note: Each additional **OPTION** parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the GIF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

GRAYSCALE determines whether the image output of a JPEG image is black and white, or color. Use GrayScale for reading in and writing out, that is, for decompressing and compressing, a JPEG image. This property affects how the image is displayed.

GrayScale is used for speed in output optimization. When GrayScale is set to ON, the color is separated from luminosity. Color takes the most time to decompress, so for previewing the image, GrayScale can be set to True for speed. The output then contains 255 shades of gray.

Values:

ON
OFF

PROGRESSIVE_ENCODING determines whether an image can be progressively displayed when it is decompressed.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the jpg-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

QUALITY determines the closeness of the saved, JPEG image to the original. As a side effect it also matter to the size of the saved file. Setting a larger value improves image quality but also increases file size; a smaller value decreases image quality and also reduces file size. Alter it according to the need.

Values:

1-100

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as JPG file)

```
PRINT Invoice WHERE TransID = 1068 +  
OPTION JPG +  
| FILENAME Invoice1068.JPG +  
| SHOW_CANCEL_DIALOG OFF +  
| BACKGROUND_COLOR WHITE +
```

```

| GRAYSCALE OFF +
| PIXELFORMAT 32 +
| QUALITY 100 +
| SCALEX 1 +
| SCALE 1 +
| OPEN ON
RETURN

```

1.16.6.10 LOTUS

PRINT reportname **ARRANGE** clause **WHERE** clause **.. ORDER BY** clause **.. OPTION** LOTUS

Note: Each additional **OPTION** parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

```

    ON (default)
    OFF

```

OPEN specifies whether the report should be opened after it is generated.

Values:

```

    ON
    OFF

```

SCALEX - scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY - scaling factor for the height of the exported page. Use 1 for actual height.

COLUMN_SPACING - specifies the column spacing between data. Use -1 for auto-spacing.

ROW_SPACING - specifies the row spacing between data. Use -1 for auto-spacing.

AUTO_FORMAT_CELLS - specifies if the format of the rendered text (Integer, Double etc) is detected automatically. Otherwise the content is just inserted as text without any format considerations.

Values:

```

    ON
    OFF

```

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the **MARGIN_LEFT** parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the **MARGIN_TOP** parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST

PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL - specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

--To print Invoice as Lotus file

```
PRINT Invoice WHERE TransID = 1068 +
OPTION LOTUS +
|FILENAME Invoice1068.WK1 +
|SHOW_CANCEL_DIALOG ON +
|OPEN OFF +
|ROW_SPACING 1
RETURN
```

1.16.6.11 PDF

PRINT reportname ARRANGE clause .. WHERE clause .. ORDER BY clause .. OPTION PDF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

MARGIN_LEFT nnn - specifies the horizontal position on the page where printing should begin. All report

component positions are relative to the margin. In other words, if the **MARGIN_LEFT** parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the **MARGIN_TOP** parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The **PAGE_LIST_VALUES** parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the **PAGE_SETTINGS** parameter when **PAGE_LIST** is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the PDF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY

MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_HYPERLINKS specifies whether to convert all Internet addresses in your report to hyperlinks.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

RICHTEXT_ENCODING_TYPE specifies the way in which RichText contents should be encoded in exported HTML document when INCLUDE_RICHTEXT option is set to ON.

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object.

Values:

IMAGE
PLAINTEXT

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as "dots per inch" to enhance the image. By increasing this value, the file size will increase. An example would be 200. The Default Value is -1.

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

GENERATE_TOC specifies whether to generate a **T**able **o**f **C**ontents based on all report Break Header (s), if defined.

Values:

ON
OFF

Set GENERATE_TOC to ON to generate a table of contents based on report headers, if defined, and encode it as outlines in PDF document.

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords in PDF document summary properties.

USE_COMPRESSION specifies whether compression should be used.

Values:

ON
OFF

COMPRESSION_METHOD specifies the type of compression to be used to compress text and Image.

Values:

FASTEST
NORMAL
MAXCOMPRESS

FONT_ENCODING specifies the type of font encoding to be used to encode font.

Values:

WIN_ANSI
MAC_ROMAN
MAC_EXPERT
STANDART
PDF_DOC

EMBED_USED_FONTS specifies whether all the used fonts should be embedded in the PDF document.

Values:

ON

OFF
FULL (same as "ON" for backward compatibility)
SUBSET

Set EMBED_USED_FONTS to ON to embed all the fonts used in the report. Fonts are embedded only if embedding is permitted depending on the Embedding Flags of the font.

ENCRYPT_FILE specifies whether the PDF document should be encrypted.

Values:

ON
OFF (default)

Set ENCRYPT_FILE to ON to encrypt the PDF document.

USER_PASSWORD specifies the user password of the encrypted PDF document. Opening the document with the correct user password allows additional operations to be performed according to the user access permissions specified by the permissions properties such as, ENABLE_PRINTING (ON/OFF), ENABLE_COPYING (ON/OFF), and ENABLE_MODIFYING (ON/OFF). These permissions are considered by the viewer applications if the document is opened with the User password. If ENABLE_PRINTING is OFF then printing is not enabled. If ENABLE_COPYING is OFF then page contents cannot be selected and copied. If ENABLE_MODIFYING is OFF then the PDF document cannot be modified.

OWNER_PASSWORD specifies the owner password of the encrypted PDF document. Opening the document with the correct owner password (assuming it is not same as the user password) allows full (owner) access to the document. This unlimited access includes the ability to change the document's passwords and access permissions.

ENCRYPTION_LEVEL specifies the level of encryption of the PDF document.

Values:

40BIT
128BIT

Set ENCRYPTION_LEVEL to 40BIT for a 40 bit encryption of the PDF document, i.e a 40 bit key is used to encrypt the PDF document. Set ENCRYPTION_LEVEL to 128BIT for a 128 bit encryption of the PDF document, i.e a 128 bit key is used to encrypt the PDF document. ENCRYPTION_LEVEL is considered only if ENCRYPT_FILE is set to ON.

ENABLE_PRINTING specifies whether to allow printing when USER_PASSWORD is set.

Values:

ON (default)
OFF

If ENABLE_PRINTING is OFF then the printing is not enabled.

ENABLE_COPYING specifies whether to allow page contents to be selected and copied when USER_PASSWORD is set.

Values:

ON (default)
OFF

If ENABLE_COPYING is OFF then the page contents cannot be selected and copied.

ENABLE_MODIFYING specifies whether the PDF document can be modified when USER_PASSWORD is set.

Values:

ON (default)
OFF

If `ENABLE_MODIFYING` is OFF then the PDF document cannot be modified.

ENABLE_ANNOTATION specifies whether the PDF document supports annotation when `USER_PASSWORD` is set.

Values:

ON
OFF

ENABLE_FORM_FILL specifies whether the PDF document supports form fill when `USER_PASSWORD` is set.

Values:

ON
OFF

ENABLE_ACCESSIBILITY specifies whether the PDF document supports tagging accessibility when `USER_PASSWORD` is set.

Values:

ON
OFF

ENABLE_DOC_ASSEMBLY specifies whether the PDF document supports document assembly when `USER_PASSWORD` is set.

Values:

ON
OFF

ENABLE_HIRES_PRINT specifies whether the PDF document supports high resolution printing when `USER_PASSWORD` is set.

Values:

ON
OFF

HIDE_TOOLBAR specifies whether to hide the viewer application's toolbars, such as Acrobat Reader or R:PDF Viewer, when the document is active. If the value is ON then the viewer applications toolbars are hidden.

Values:

ON
OFF (default)

HIDE_MENUBAR specifies whether to hide the viewer application's menu bar, such as Acrobat Reader or R:PDF Viewer, when the document is active. If the value is ON then the viewer applications menu bar is hidden.

Values:

ON
OFF (default)

HIDE_WINDOW_UI specifies whether to hide user interface elements in the document's window (such as scroll bars and navigation controls), leaving only the document's contents displayed. If the value is ON then documents user interface elements are hidden.

Values:

ON
OFF (default)

FIT_WINDOW specifies whether to resize the document's window to fit the size of the first displayed page. If the value is ON then the first page displayed is resized to fit the document's window.

Values:

ON
OFF (default)

CENTER_WINDOW specifies whether to position the document's window in the center of the screen. If the value is ON then the document is positioned at the center of the screen when opened.

Values:

ON
OFF (default)

PAGE_LAYOUT specifies the page layout to be used when the PDF document is opened.

Values:

SINGLE_PAGE
ONE_COLUMN
TWO_COLUMN_LEFT
TWO_COLUMN_RIGHT

If the value is set to SINGLE_PAGE then one page is displayed at a time. If the value is set to ONE_COLUMN then pages are displayed in one column. If the value is set to TWO_COLUMN_LEFT then pages are displayed in two columns with odd numbered pages on the left. If the value is set to TWO_COLUMN_RIGHT then pages are displayed in two columns with odd numbered pages on the right.

PAGE_MODE specifies how the document should be displayed when opened.

Values:

ONE
OUTLINES
THUMBS
FULLSCREEN

If the value is set to ONE then neither document outlines nor thumbnail images are visible. If the value is set to OUTLINES then document outline is visible. If the value is set to THUMBS then thumbnail images are visible. If the value is set to FULLSCREEN then document is opened in Full Screen mode with no menu bar, window controls, or any other window visible.

NON_FULL_SCREEN_PAGE_MODE specifies how to display the document on exiting full screen mode. This is meaningful only if the PAGE_MODE value is set to FULLSCREEN.

Values:

NONE
USE_OUTLINES
USE_THUMB

If the value is set to NONE then neither document outlines nor thumbnail images are visible. If the value is set to USE_OUTLINES then document outline is visible. If the value is set to USE_THUMB then thumbnail images are visible.

PRESENTATION_MODE similar to transition effects specifies how the viewer application, such as Acrobat Reader or R:PDF Viewer, should display/present the document. Some viewer applications allow a document to be displayed in the form of a presentation or "slide show," advancing from one page to the next either automatically or under user control.

Values:

SPLIT_HOR_INWARD
SPLIT_HOR_OUTWARD
SPLIT_VERT_INWARD

SPLIT_VERT_OUTWARD
BLINDS_HOR
BLINDS_VERT
BOX_INWARD
BOX_OUTWARD
WIPE_LEFT_TO_RIGHT
WIPE_BOTTOM_TO_TOP
WIPE_RIGHT_TO_LEFT
WIPE_TOP_TO_BOTTOM
DISSOLVE
GLITTER_LEFT_TO_RIGHT
GLITTER_TOP_TO_BOTTOM
GLITTER_TOPLEFT_TO_BOTTOMRIGHT
NONE

SPLIT_HOR_INWARD: Two lines sweep across the screen, revealing the new page. The lines are horizontal and move inward from the edges of the page.

SPLIT_HOR_OUTWARD: Two lines sweep across the screen, revealing the new page. The lines are horizontal and move outward from the center of the page.

SPLIT_VERT_INWARD: Two lines sweep across the screen, revealing the new page. The lines are vertical and move inward from the edges of the page.

SPLIT_VERT_OUTWARD: Two lines sweep across the screen, revealing the new page. The lines are vertical and move outward from the center of the page.

BLINDS_HOR: Multiple lines, evenly spaced across the screen, synchronously sweep in the same direction to reveal the new page. The lines are horizontal and move downward.

BLINDS_VERT: Multiple lines, evenly spaced across the screen, synchronously sweep in the same direction to reveal the new page. The lines are vertical and move to the right.

BOX_INWARD: A rectangular box sweeps inward from the edges of the page revealing the new page.

BOX_OUTWARD: A rectangular box sweeps outward from the center of the page revealing the new page.

WIPE_LEFT_TO_RIGHT: A single line sweeps across the screen from Left to Right edge of the page

WIPE_BOTTOM_TO_TOP: A single line sweeps across the screen from Bottom to the Top edge of the page

WIPE_RIGHT_TO_LEFT: A single line sweeps across the screen from Right to the Left edge of the page

WIPE_TOP_TO_BOTTOM: A single line sweeps across the screen from Top to Bottom edge of the page

DISSOLVE: The old page "dissolves" gradually to reveal the new one.

GLITTER_LEFT_TO_RIGHT: Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from Left side of the screen to the Right side.

GLITTER_TOP_TO_BOTTOM: Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from Top side of the screen to the Bottom side.

GLITTER_TOPLEFT_TO_BOTTOMRIGHT: Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from TopLeft corner of the screen to the BottomRight corner.

NONE: The new page simply replaces the old one with no special transition effect.

DURATION specifies the duration of the presentation mode (transition effect) in seconds.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print SalesSumAndSubTot report as secure PDF document)

```
PRINT SalesSumAndSubTot +
OPTION PDF +
|FILENAME SalesSumAndSubTotals.PDF +
|SHOW_CANCEL_DIALOG ON +
|BACKGROUND_FILE NONE +
|BACKGROUND_TYPE CENTER +
|BACKGROUND_COLOR WHITE +
|INCLUDE_LINES ON +
|INCLUDE_SHAPES ON +
|INCLUDE_RICHTEXT ON +
|RICHTEXT_ENCODING_TYPE PLAINTEXT +
|INCLUDE_IMAGES ON +
|IMAGE_FORMAT JPG +
|PIXELFORMAT 32 +
|JPEG_QUALITY 100 +
|IMAGE_DPI -1 +
|INCLUDE_HYPERLINKS ON +
|GENERATE_TOC ON +
|TITLE Customer Sales Sub-Totals and Totals +
|SUBJECT Sales Summary Report +
|AUTHOR Accounting Team of Consolidated Computer Company +
|KEYWORDS SubTotals Totals Sales Summary +
|USE_COMPRESSION ON +
|COMPRESSION_METHOD MAXCOMPRESS +
|FONT_ENCODING WIN_ANSI +
|EMBED_USED_FONTS ON +
|ENCRYPT_FILE ON +
|USER_PASSWORD RRBYW10 +
|OWNER_PASSWORD RRBYW10owner +
|ENCRYPTION_LEVEL 128BIT +
|ENABLE_PRINTING OFF +
|ENABLE_COPYING OFF +
|ENABLE_MODIFYING OFF +
|HIDE_TOOLBAR OFF +
|HIDE_MENUBAR OFF +
|HIDE_WINDOW_UI OFF +
|FIT_WINDOW OFF +
|CENTER_WINDOW OFF +
```

```

| PAGE_LAYOUT SINGLE_PAGE +
| PAGE_MODE NONE +
| NON_FULL_SCREEN_PAGE_MODE NONE +
| PRESENTATION_MODE NONE +
| DURATION 0 +
| OPEN ON
RETURN

```

1.16.6.12 PNG

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION PNG

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

```

ON (default)
OFF

```

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report.

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

INCLUDE_TEXT specifies whether text on the report should be included in the exported file.

Values:

```

ON
OFF

```

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

```

ON
OFF

```

INCLUDE_IMAGES specifies whether images on the report should be included in the exported file.

Values:

```

ON
OFF

```

IMAGE_DPI specifies the depth of image as "dots per inch" to enhance the image. By increasing this value, the file size will increase. An example would be 200. The Default Value is -1.

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

```

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit

```

16 : 16bit
 24 : 24bit
 32 : 32bit

COLOR_TYPE specifies the color type of the image

GRAYSCALE
 RGB (default)
 RGB_ALPHA
 GRAY_ALPHA

PIXEL_DIMENSIONS enables the ability to set PNG pixel dimension information

Values:

ON
 OFF

UNIT_SPECIFIER specifies the units for setting the pixel dimension information. **PIXEL_DIMENSIONS** must be set ON in order to use the parameter.

Values:

NONE - X_PIXELS_PER_UNIT and Y_PIXELS_PER_UNIT defines pixel aspect ratio only, the actual size of the pixels remains unspecified.
 METER - the actual size of the pixels is specified

X_PIXELS_PER_UNIT specifies the number of pixels per unit in the horizontal direction. **PIXEL_DIMENSIONS** must be set ON in order to use the parameter. Default: 1

Y_PIXELS_PER_UNIT specifies the number of pixels per unit in the vertical direction. **PIXEL_DIMENSIONS** must be set ON in order to use the parameter. Default: 1

TRANSPARENCY sets the fields related to transparency of a PNG output

Values:

ON
 OFF

IMAGE_ALPHA specifies a percentage of transparency for the PNG output. This is applicable only when **TRANSPARENCY** is ON and **COLOR_TYPE** is either **RGB_ALPHA** or **GRAY_ALPHA**. If **IMAGE_ALPHA** is 0, then the image is completely transparent. If **IMAGE_ALPHA** is 100, then the image is completely opaque. Default: 0

Values:

0 - 100

TRANSPARENT_COLOR specifies the color that is transparent. This is applicable only when **TRANSPARENCY** is ON and **COLOR_TYPE** is either **RGB** or **GRAYSCALE**.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
 OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the **MARGIN_LEFT** parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the **MARGIN_TOP** parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT*nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

--To print Invoice as PNG file

```
PRINT Invoice WHERE TransID = 1068 +
OPTION PNG +
|FILENAME Invoice1068.PNG +
|SHOW_CANCEL_DIALOG OFF +
|INCLUDE_SHAPES OFF +
|INCLUDE_IMAGES ON +
|OPEN ON
RETURN
```

1.16.6.13 QPRO

PRINT reportname **ARRANGE** clause **WHERE** clause **.. ORDER BY** clause **.. OPTION QPRO**

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

SCALEX - scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY - scaling factor for the height of the exported page. Use 1 for actual height.

COLUMN_SPACING - specifies the column spacing between data. Use -1 for auto-spacing.

ROW_SPACING - specifies the row spacing between data. Use -1 for auto-spacing.

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

AUTO_FORMAT_CELLS - specifies if the format of the rendered text (Integer, Double etc) is detected automatically. Otherwise the content is just inserted as text without any format considerations.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM*nn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL - specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

--To print Invoice as Quattro Pro file

```
PRINT Invoice WHERE TransID = 1068 +
OPTION QPRO +
|FILENAME Invoice1068.WB1 +
|SHOW_CANCEL_DIALOG OFF +
|OPEN ON +
|COLUMN_SPACING 1
RETURN
```

1.16.6.14 RTF

PRINT reportname **ARRANGE** clause **WHERE** clause .. **ORDER BY** clause .. **OPTION RTF**

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)

OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the RTF file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as "dots per inch" to enhance the image. By increasing this value, the file size will increase. An example would be 200. The Default Value is -1.

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

GRAPHIC_DATA_IN_BINARY specifies whether graphic data is to be encoded in binary format.

Values:

ON (default)
OFF

Set to ON to encode graphic data in binary format. If OFF, graphic data is encoded in hexadecimal format.

OPTIMIZATION specifies the type of encoding of the RTF document.

Values:

POSITIONAL_ACCURACY (default)
EASE_OF_EDITING

Resulting RTF document can be either optimized for positional accuracy (meaning that the placement of the objects will be as close as possible to a printed report), or for ease of editing (meaning that everything is treated as simple text and it is much easier to edit the file at the expense of aesthetics).

Set OPTIMIZATION EASE_OF_EDITING if you wish to easily modify the resulting RTF document.

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords Meta content in RTF document property summary.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print invoice report as RTF document)

PRINT Invoice WHERE TransID = 1068 +

OPTION RTF +

| **FILENAME** Invoice1068.RTF +

| **SHOW_CANCEL_DIALOG** ON +

| **BACKGROUND_FILE** NONE +

| **BACKGROUND_TYPE** CENTER +

| **INCLUDE_LINES** ON +

| **INCLUDE_SHAPES** ON +

```

| INCLUDE_RICHTEXT ON +
| INCLUDE_IMAGES ON +
| IMAGE_FORMAT JPG +
| JPEG_QUALITY 100 +
| IMAGE_DPI -1 +
| PIXELFORMAT 32 +
| GRAPHIC_DATA_IN_BINARY ON +
| OPTIMIZATION POSITIONAL_ACCURACY +
| TITLE Customer Invoice +
| SUBJECT Sales Order +
| AUTHOR Accounting Team of Consolidated Computer Company +
| KEYWORDS Invoice +
| OPEN ON
RETURN

```

1.16.6.15 SVG

PRINT reportname **ARRANGE** clause **WHERE** clause **.. ORDER BY** clause **.. OPTION** SVG

Note: Each additional **OPTION** parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

```

ON (default)
OFF

```

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report.

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

INCLUDE_TEXT specifies whether text on the report should be included in the exported file.

Values:

```

ON
OFF

```

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

```

ON
OFF

```

INCLUDE_IMAGES specifies whether images on the report should be included in the exported file.

Values:

```

ON
OFF

```

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when

INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as "dots per inch" to enhance the image. By increasing this value, the file size will increase. An example would be 200. The Default Value is -1.

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL

FIRST
 LAST
 Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
 OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

--To print Invoice as SVG file

```
PRINT Invoice WHERE TransID = 1068 +
OPTION PNG +
| FILENAME Invoice1068.SVG +
| SHOW_CANCEL_DIALOG OFF +
| IMAGE_FORMAT JPG +
| JPEG_QUALITY 100 +
| OPEN ON
RETURN
```

1.16.6.16 SYLK

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION SYLK

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
 OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
 OFF

SCALEX - scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY - scaling factor for the height of the exported page. Use 1 for actual height.

COLUMN_SPACING - specifies the column spacing between data. Use -1 for auto-spacing.

ROW_SPACING - specifies the row spacing between data. Use -1 for auto-spacing.

AUTO_FORMAT_CELLS - specifies if the format of the rendered text (Integer, Double etc) is detected automatically. Otherwise the content is just inserted as text without any format considerations.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL - specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

--To print Invoice as Symbolik Link format file

```
PRINT Invoice WHERE TransID = 1070 +  
OPTION SYLK +  
|FILENAME Invoice1070.sylk +  
|SHOW_CANCEL_DIALOG ON +  
|OPEN ON +  
|ROW_SPACING 1  
RETURN
```

1.16.6.17 TIFF

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION TIFF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

BACKGROUND_TYPE specifies the way background image has to be displayed in the BMP file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

MONOCHROME specifies whether tiff-file produced are color or monochrome images.

Values:

ON
OFF

MULTIPAGE specifies the output report to one multi-page file or one file per page.

Values:

ON
OFF

PIXELFORMAT specifies the pixel format of the tiff-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit
32 : 32bit

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

COMPRESSION When an image is loaded, this parameter indicates the compression used by the image. When saving in image, the parameter may be set to specify which compression method to use.

Values:

NONE
PACKBITS
GROUP3_1D
GROUP3_2D
GROUP4
ZLIB
JPEG

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST

LAST

PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as TIF file)

```
PRINT Invoice WHERE TransID = 1068 +
OPTION TIFF +
| FILENAME Invoice1068.TIF +
| SHOW_CANCEL_DIALOG OFF +
| BACKGROUND_COLOR WHITE +
| MONOCHROME OFF +
| MULTIPAGE OFF +
| PIXELFORMAT 32 +
| SCALEX 1 +
| SCALE 1 +
| COMPRESSION NONE +
| OPEN ON
RETURN
```

1.16.6.18 TXT

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION TXT

Notes:

- Each additional OPTION parameter must be separated by the pipe | symbol.
- The TXT "Text File" option allows you to define a particular band to print the data as Comma Delimited, Tab Delimited or Fixed Length file. This is the perfect option to export selective and/or summarized data used in a report band.

Caution:

This option requires the report setup to define the exact band to be print as text file.

Report Designer | File | Print to Text File Setup ...

If a user attempts to PRINT the report as Text File (OPTION TXT) without first having defined the Text File Setup in Report Designer, the program will return an -ERROR- message and not just write a zero length file.

The same rule applies when printing the report as Text File using the Print to a File option from the Database Explorer.

Database Explorer | Reports | Run Report | Print Target .. File

For step-by-step instructions on how to set up a report to print as Text File, please refer to the article on From The Edge: Printing Report as Text File (May 6, 2005).

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST

PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL

FIRST
 LAST
 Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
 OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print columnar report as Text file)

```
PRINT ColumnarReport +
OPTION TXT +
|FILENAME ColumnarReport.TXT +
|SHOW_CANCEL_DIALOG OFF +
|OPEN ON
RETURN
```

1.16.6.19 WMF

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION WMF

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
 OFF

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the WMF file.

Values:

TILE
 TOPLEFT
 TOPCENTER
 TOPRIGHT
 CENTERLEFT
 CENTER
 CENTERRIGHT
 BOTTOMLEFT
 BOTTOMCENTER
 BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)
YELLOW

SCALEX double; Scaling factor for the width of the exported page. Use 1 for actual width.

SCALEY double; Scaling factor for the height of the exported page. Use 1 for actual height.

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT *nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print Invoice as WMF file)

```
PRINT Invoice WHERE TransID = 1068 +
OPTION WMF +
| FILENAME Invoice1068.WMF +
| SHOW_CANCEL_DIALOG OFF +
| BACKGROUND_COLOR WHITE +
| SCALEX 1 +
| SCALE 1 +
| OPEN ON
RETURN
```

1.16.6.20 XHTML

PRINT reportname ARRANGE clause WHERE clause .. ORDER BY clause .. OPTION XHTML

Note: Each additional OPTION parameter must be separated by the pipe | symbol.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

MARGIN_LEFT nnn - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP nnn - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT nnn - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM*nn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

BACKGROUND_FILE stores the image file name that is to be used as background image of all pages of report.

BACKGROUND_TYPE specifies the way background image has to be displayed in the XHTML file.

Values:

TILE
TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

BACKGROUND_COLOR indicates the color to be used as background color of all pages of report.

Values:

AQUA
BLACK
BLUE
CREAM
DARK GRAY
FUCHSIA
GRAY
GREEN
LIME
LIGHT GRAY
MAROON
MEDIUM GRAY
MINT GREEN
NAVY
OLIVE
PURPLE
RED
SILVER
SKY BLUE
TEAL
WHITE (default)

YELLOW

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF

INCLUDE_RICHTEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON
OFF

RICHTEXT_ENCODING_TYPE specifies the way in which RichText contents should be encoded in exported HTML document when INCLUDE_RICHTEXT option is set to ON.

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object.

Values:

IMAGE
PLAINTEXT

INCLUDE_IMAGES specifies whether Images on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report when INCLUDE_IMAGES option is set to ON.

Values:

GIF
JPG
BMP

JPEG_QUALITY specifies the quality of the JPEG Image when INCLUDE_IMAGES is set to ON and IMAGE_FORMAT is set to JPG.

Values:

1-100

IMAGE_DPI specifies the depth of image as "dots per inch" to enhance the image. By increasing this value, the file size will increase. An example would be 200. The Default Value is -1.

PIXELFORMAT specifies the pixel format of the image-file (bits/pixel).

Values:

1 : 1bit
4 : 4bit
8 : 8bit
15 : 15bit
16 : 16bit
24 : 24bit

32 : 32bit

INCLUDE_HYPERLINKS specifies whether hyperlinks in your report should be included in the exported file.

Values:

ON
OFF

TITLE specifies the title of the report.

SUBJECT specifies the subject of the report.

AUTHOR specifies the author of the report.

KEYWORDS stores the list of keywords used in the report. Specify a list of keywords for the document that should be encoded as the Keywords Meta content.

PAGE_END_LINES indicates whether a line should be drawn at the end of each page. Setting **PAGE_END_LINES** to ON draws a line after each page. **PAGE_END_LINES** is considered only when **SEP_FILE_PER_PAGE** is set to OFF.

Values:

ON
OFF

OUTPUT_STYLES_TO_CSS_FILE specifies whether to output styles to a CSS (Cascade Style Sheet) file.

Values:

ON
OFF

CSSFILE specify the CSS file name if **OUTPUT_STYLES_TO_CSS_FILE** is set to ON.

OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE specifies whether to output scripts to a Java Script File.

Values:

ON
OFF

JAVA_SCRIPT_FILE_NAME specify the file name if the **OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE** is set to ON.

GENERATE_TOC specifies whether to generate a **Table of Contents** based on all report Break Header (s), if defined.

Values:

ON
OFF

TOC_FILE_NAME specify the file name for the Table of Contents, if the **GENERATE_TOC** option is set to ON.

IMAGE_FOLDER specify the folder path and name for XHTML document images.

DEFAULT_FONT_NAME specify the default font name for XHTML document.

DEFAULT_FONT_SIZE specify the default font size for XHTML document.

DEFAULT_FONT_COLOR specify the default font color for XHTML document.

DEFAULT_FONT_BOLD specifies whether the font is bold faced.

Values:

ON
OFF

DEFAULT_FONT_ITALIC specifies whether the font is italicized.

Values:

ON
OFF

DEFAULT_FONT_UNDERLINE specifies whether the font is underlined.

Values:

ON
OFF

DEFAULT_FONT_STRIKEOUT specifies whether the font is displayed with a horizontal line through it.

Values:

ON
OFF

SEP_FILE_PER_PAGE specifies whether to create a separate file for each page of the report.

Values:

ON
OFF

NAV_SHOW Set to ON to show a collection of links that enable easy navigation between pages of the exported XHTML. The links function much like the navigator buttons in the standard preview of Report Preview: one link to jump to the first page; one to the last page; one to the next page and another to get back to the previous page. NAV_SHOW is only effective when the option for SEP_FILE_PER_PAGE is set to ON.

Values:

ON
OFF

LINK_BACK_COLOR specifies the background color of navigator. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

LINK_H_BACK_COLOR specifies the background color of navigator when the mouse is hovered over it. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

LINK_H_FORE_COLOR specifies the color of navigator when the mouse is hovered over it. This option will only be used if the SEP_FILE_PER_PAGE and NAV_SHOW options are set to ON.

USE_TEXT_LINKS specifies the type of the navigator links that should be encoded for a multi-file HTML output. Setting USE_TEXT_LINKS to ON uses the text provided in **LINK_TEXT_FIRST**, **LINK_TEXT_PREV**, **LINK_TEXT_NEXT** and **LINK_TEXT_LAST** as the hyperlink's caption. Setting to OFF uses Images specified by **LINK_IMG_FILE_FIRST**, **LINK_IMG_FILE_PREV**, **LINK_IMG_FILE_NEXT**, **LINK_IMG_FILE_LAST**. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

ON
OFF

LINK_TEXT_FIRST specifies the hyperlink's text caption for first page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_PREV specifies the hyperlink's text caption for previous page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_NEXT specifies the hyperlink's text caption for next page when USE_TEXT_LINKS option is set to ON.

LINK_TEXT_LAST specifies the hyperlink's text caption for last page when USE_TEXT_LINKS option is set to ON.

LINK_FONT_NAME specify the font name for navigator links.

LINK_FONT_SIZE specify the font size for navigator links.

LINK_FONT_COLOR specify the font color for navigator links.

LINK_FONT_BOLD specifies whether the font is bold faced for navigator links.

Values:

ON
OFF

LINK_FONT_ITALIC specifies whether the font is italicized for navigator links.

Values:

ON
OFF

LINK_FONT_UNDERLINE specifies whether the font is underlined for navigator links.

Values:

ON
OFF

LINK_FONT_STRIKEOUT specifies whether the font for navigator link is displayed with a horizontal line through it.

Values:

ON
OFF

LINK_IMG_FILE_FIRST specifies the hyperlink's image file for first page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_PREV specifies the hyperlink's image file for previous page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_NEXT specifies the hyperlink's image file for next page when USE_TEXT_LINKS option is set to OFF.

LINK_IMG_FILE_LAST specifies the hyperlink's image file for last page when USE_TEXT_LINKS option is set to OFF.

NAV_TYPE specifies the type of the navigator. If NAV_TYPE is set to SCREEN then the navigator will be fixed to screen, else the navigator will be fixed to page. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

PAGE
SCREEN

NAV_ORIENT specifies the orientation of the navigator. if NAV_ORIENT is set to HORIZONTAL then, a Horizontal navigator is encoded, else a Vertical navigator is encoded. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

VERTICAL
HORIZONTAL

NAV_POS specifies the position of the navigator. The property is effective only when the options for SEP_FILE_PER_PAGE and NAV_SHOW are set to ON.

Values:

TOPLEFT
TOPCENTER
TOPRIGHT
CENTERLEFT
CENTER
CENTERRIGHT
BOTTOMLEFT
BOTTOMCENTER
BOTTOMRIGHT

DISABLE_RIGHT_CLICK specifies whether to disable the typical speed menu options when you right click on your mouse button.

Values:

ON
OFF

DISABLE_COPYING specifies whether to disable the option to highlight and copy the text.

Values:

ON
OFF

AUTO_MAXIMIZE_WINDOW specifies whether to automatically maximize the window when HTML document is opened.

Values:

ON
OFF

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

Example:

Example 01 (To print SalesSumAndSubTot report as XHTML file)

```
PRINT SalesSumAndSubTot +  
OPTION XHTML +  
|FILENAME SalesSumAndSubTot.htm +  
|SHOW_CANCEL_DIALOG ON +  
|BACKGROUND_FILE NONE +  
|BACKGROUND_TYPE CENTER +  
|BACKGROUND_COLOR WHITE +  
|INCLUDE_LINES ON +  
|INCLUDE_SHAPES ON +  
|INCLUDE_RICHTEXT OFF +
```

|**RICHTEXT_ENCODING_TYPE** PLAINTEXT +
|**INCLUDE_IMAGES** ON +
|**IMAGE_FORMAT** JPG +
|**JPEG_QUALITY** 100 +
|**IMAGE_DPI** -1 +
|**PIXELFORMAT** 24 +
|**INCLUDE_HYPERLINKS** ON +
|**TITLE** Customer Sales Sub-Totals and Totals +
|**SUBJECT** ConComp Sales Summary Report +
|**AUTHOR** Accounting Team of ConComp +
|**KEYWORDS** SubTotals Totals Sales Summary +
|**PAGE_END_LINES** ON +
|**OUTPUT_STYLES_TO_CSS_FILE** ON +
|**CSSFILE** +
|**OUTPUT_SCRIPTS_TO_JAVA_SCRIPT_FILE** ON +
|**JAVA_SCRIPT_FILE_NAME** +
|**GENERATE_TOC** ON +
|**TOC_FILE_NAME** Index.htm +
|**IMAGE_FOLDER** Images +
|**DEFAULT_FONT_NAME** Verdana +
|**DEFAULT_FONT_SIZE** 9 +
|**DEFAULT_FONT_COLOR** BLACK +
|**DEFAULT_FONT_BOLD** OFF +
|**DEFAULT_FONT_ITALIC** OFF +
|**DEFAULT_FONT_UNDERLINE** OFF +
|**DEFAULT_FONT_STRIKEOUT** OFF +
|**SEP_FILE_PER_PAGE** ON +
|**NAV_SHOW** ON +
|**LINK_BACK_COLOR** WHITE +
|**LINK_H_BACK_COLOR** BLUE +
|**LINK_H_FORE_COLOR** WHITE +
|**USE_TEXT_LINKS** OFF +
|**LINK_TEXT_FIRST** First +
|**LINK_TEXT_PREV** Previous +
|**LINK_TEXT_NEXT** Next +
|**LINK_TEXT_LAST** Last +
|**LINK_FONT_NAME** Wingdings +
|**LINK_FONT_SIZE** 10 +
|**LINK_FONT_COLOR** BLUE +
|**LINK_FONT_BOLD** OFF +
|**LINK_FONT_ITALIC** OFF +
|**LINK_FONT_UNDERLINE** OFF +
|**LINK_FONT_STRIKEOUT** OFF +
|**LINK_IMG_FILE_FIRST** Images\FirstOff.gif +
|**LINK_IMG_FILE_PREV** Images\PrevOff.gif +
|**LINK_IMG_FILE_NEXT** Images\NextOff.gif +
|**LINK_IMG_FILE_LAST** Images\LastOff.gif +
|**NAV_TYPE** PAGE +
|**NAV_ORIENT** HORIZONTAL +
|**NAV_POS** BOTTOMCENTER +
|**DISABLE_RIGHT_CLICK** OFF +
|**DISABLE_COPYING** OFF +
|**AUTO_MAXIMIZE_WINDOW** ON +

| OPEN ON
RETURN

1.16.6.21 XLS

PRINT reportname **ARRANGE** clause **WHERE** clause .. **ORDER BY** clause .. **OPTION XLS**

Notes:

- Each additional **OPTION** parameter must be separated by the pipe | symbol.
- When printing to the XLS format, the report objects must be spaced apart appropriately or "padded" in order for the report data to appear correctly in the Excel report output. Experimenting with vertical and horizontal will also allow you to attain your desired output. Also, please refrain from using elaborate fonts that may not be supported within Excel.

OPTION Parameters:

FILENAME specifies the full path and filename. If no path is defined, output will be saved in the current working directory.

SHOW_CANCEL_DIALOG specifies whether to suppress cancel dialog box during the print process.

Values:

ON (default)
OFF

ALL_TEXT_IN_GENERAL_FORMAT Specifies whether all the text in the report should be converted to general format or the format of the text should be detected and encoded accordingly.

Vales:

ON
OFF

Set **ALL_TEXT_IN_GENERAL_FORMAT** to ON to encode all the text contents in the report in general format. General format text have no specific number format. Set **ALL_TEXT_IN_GENERAL_FORMAT** to OFF to detect the format of the text and encode accordingly. In this case number and other formats are identified and encoded.

CELL_ATTR Cell attributes include shading of the cell whose color is not WHITE.

Values:

ON
OFF

INCLUDE_IMAGES specifies whether Image objects on the report should be included in the exported file.

Values:

ON
OFF

IMAGE_FORMAT specifies the Linked image file format for image controls used in the report.

Values:

BMP
WMF

INCLUDE_LINES specifies whether lines on the report should be included in the exported file.

Values:

ON
OFF (default)

INCLUDE_RICH_TEXT specifies whether contents of the RichText objects on the report should be included in the exported file.

Values:

ON (default)
OFF

RICHTEXT_ENCODING_TYPE indicates the way in which RichText contents should be encoded in Excel document.

Values:

IMAGE (default)
PLAINTEXT

Set RICHTEXT_ENCODING_TYPE to IMAGE to encode RichText contents as exact graphical representations. Set RICHTEXT_ENCODING_TYPE to PLAINTEXT to encode RichText contents as unformatted text. The font used when encoding RichText as plain text is the one set in the Font property of the RichText object. When RICHTEXT_ENCODING_TYPE is set to PLAINTEXT make sure you set the Default Font property of the RichText control in report designer to true and set the required font in the Font property.

INCLUDE_SHAPES specifies whether shapes on the report should be included in the exported file.

Values:

ON
OFF (default)

LINE_SPACE specifies the spacing to be used between lines of text.

Values:

ACTUAL
0
1
2
3
4
5

ONE_SHEET_PER_PAGE specifies whether each page in the report should be encoded into separate excel sheets or all the pages should be encoded in a single excel sheet.

Values:

ON
OFF (default)

OPEN specifies whether the report should be opened after it is generated.

Values:

ON
OFF

MARGIN_LEFT *nnn* - specifies the horizontal position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_LEFT parameter is set to 0.25 inches and you place a report component in a band and set the component's "left" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the left margin).

MARGIN_TOP *nnn* - specifies the vertical position on the page where printing should begin. All report component positions are relative to the margin. In other words, if the MARGIN_TOP parameter is set to 0.25 inches and you place a report component in a band and set the component's "top" parameter to 0, then that component will print 0.25 inches from the edge of page (or at the top margin).

MARGIN_RIGHT*nnn* - specifies the horizontal position on the page where printing should stop. All report component positions are relative to the margin.

MARGIN_BOTTOM *nnn* - specifies the vertical position on the page where printing should stop. All report component positions are relative to the margin.

PAGE_SETTINGS specifies the option to print All, First, Last or a given number of pages.

Values:

ALL
FIRST
LAST
PAGE_LIST - specifies the option to print a given number of pages. The PAGE_LIST_VALUES parameter must be used with option.

PAGE_LIST_VALUES specifies the values for number of pages, such as, 3,5,8 or 3,8 or 3, or 1-3,8 or 3,5,9-15. This option must be used with the PAGE_SETTINGS parameter when PAGE_LIST is specified.

PAGE_STYLE_SETTINGS - determines the pages in which the Page Style will appear on.

Values:

ALL
FIRST
LAST
Page List i.e. 1,4-6,10-12,20

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

See additional options when EMAIL value is set to ON. [EMAIL SET ON](#)

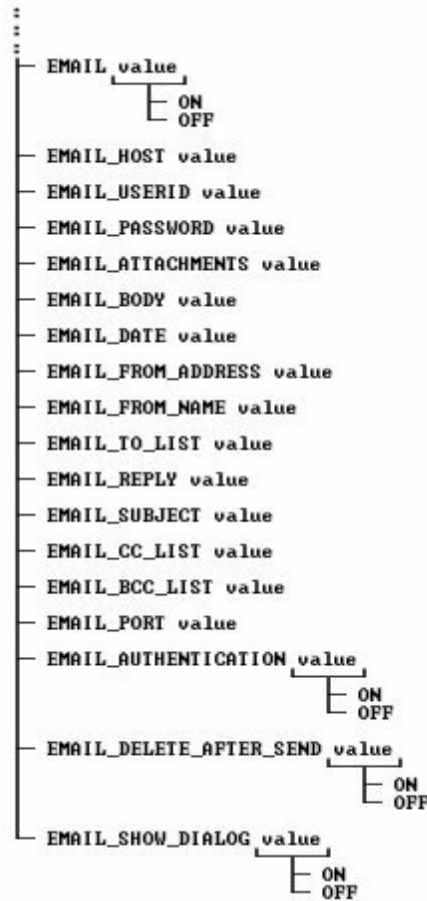
Example:

Example 01 (To print Invoices as XLS file)

```
PRINT Invoice WHERE CustID = 127 +
OPTION XLS +
|FILENAME Invoice.XLS +
|SHOW_CANCEL_DIALOG OFF +
|ALL_TEXT_IN_GENERAL_FORMAT OFF +
|CELL_ATTR ON +
|INCLUDE_IMAGES OFF +
|INCLUDE_LINES OFF +
|INCLUDE_RICH_TEXT ON +
|RICHTEXT_ENCODING_TYPE IMAGE +
|INCLUDE_SHAPES OFF +
|LINE_SPACE ACTUAL +
|ONE_SHEET_PER_PAGE ON +
|OPEN ON
RETURN
```


1.16.6.22 EMAIL SET ON

Additional options when the **EMAIL** value is set to **ON**:



Note: Each additional OPTION parameter must be separated by pipe | symbol

OPTION Parameters:

EMAIL specifies the report output (defined as FILENAME filename.ext) to be sent via e-mail as an attachment.

Values:

ON
OFF

EMAIL_HOST specifies Hostname/IP Address of SMTP Server (Outgoing). Also sometimes known as your 'Sending Mail Server', your SMTP server receives and processes all of your outgoing mail. For example, your Outgoing Mail Server might be called something like 'my-internet-provider.com' or 'outgoing-mail.mycompany.com'. The SMTP server name is sometimes the same as your incoming mail server name. If your ISP or your network administrator has not given this server name to you, you can leave the box blank. If you do not know what your outgoing mail server name is, you need to contact your Internet Service Provider or Network Administrator.

EMAIL_USERID specifies a valid User Name or Login Name for SMTP Server. This is the name you use to log into your email account. Your ISP or organization provides it. It is generally, but not necessarily, the name that appears before the at sign (@) in your return email address. For example, if your email address was <jim@my-server.com> - then your username would be 'jim' (no quotes). If you do not know what your username is, you need to contact your Internet Service Provider or Network

Administrator.

EMAIL_PASSWORD specifies valid SMTP Server Password for the user.

EMAIL_ATTACHMENTS specifies file list. Each file must be separated by # symbol.

EMAIL_BODY This option contains body of the mail.

EMAIL_DATE This option holds the date the message was sent. (Default = .#DATE)

EMAIL_FROM_ADDRESS specifies the address of the E-mail account from which the mail is to be sent. Sender E-Mail Address (From E-Mail Address:) Enter your e-mail address here. For example, your email address might be 'jim@my-internet-service.com' or 'molly@my-business-domain.com' Take care not to make a mistake here - or else it will make it difficult for others to reply to your emails. If you do not know what your e-mail address is, you need to contact your Internet Service Provider or Network Administrator.

EMAIL_FROM_NAME represents the person's name in an E-mail address. Your real name or what you want to appear as your name on messages that you send out.

EMAIL_TO_LIST identifies the recipient(s) of a message. Specify the list of valid e-mail address(s) to which the e-mail is to be sent.

EMAIL_REPLY identifies the delivery address for message replies. This is typically the same as Sender E-Mail Address. Enter your e-mail address here. For example, your email address might be 'jim@my-internet-service.com' or 'molly@my-business-domain.com' Take care not to make a mistake here - or else it will make it difficult for others to reply to your emails. If you do not know what your e-mail address is, you need to contact your Internet Service Provider or Network Administrator.

EMAIL_SUBJECT identifies the subject for the message. Subject is a text string that identifies the subject for the message.

EMAIL_CC_LIST specifies the list of e-mail addresses as Carbon Copies to be sent along with the e-mail.

EMAIL_BCC_LIST specifies the list of e-mail addresses as Blind Carbon Copies to be sent along with the mail.

EMAIL_PORT specifies the Integer value of SMTP Server Port. (Default = 25).

EMAIL_AUTHENTICATION Set the value for this option to ON if your SMTP Server requires user authentication.

Values:

ON
OFF

EMAIL_DELETE_AFTER_SEND Set the value for this option to ON if you wish to delete the report files after sending e-mail.

EMAIL_SHOW_DIALOG Set the value for this option to ON if you would like to show e-mail setup dialog.

Example:

```
CONNECT RRBYW10
PRINT Invoice WHERE TransID = 1031 +
OPTION PDF|FILENAME Invoice_1031.PDF +
|EMAIL ON +
|EMAIL_TO_LIST yourname@yourdomain.com, whoelse@yourdomain.com +
|EMAIL_FROM_NAME John D. Johnson +
|EMAIL_FROM_ADDRESS johndjohnson@yourdomain.com +
```

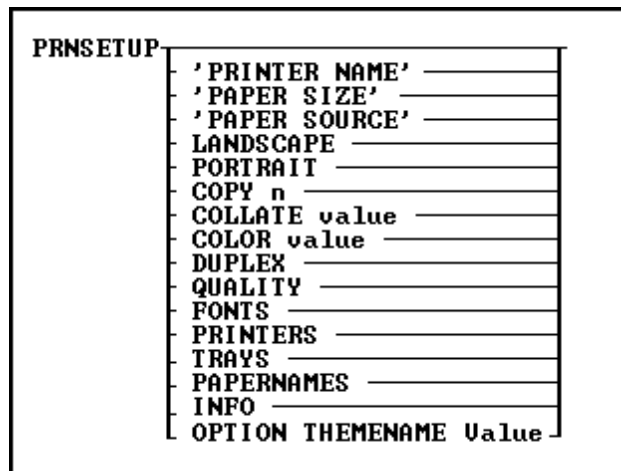
```

|EMAIL_SUBJECT Imagine The Possibilities! +
|EMAIL_BODY Attached is the invoice - sent via R:BASE. Because we Can! +
|EMAIL_ATTACHMENTS C:\Letters\otherstuff.pdf#C:\Letters\morestuff.pdf +
|EMAIL_DELETE_AFTER_SEND ON +
|EMAIL_SHOW_DIALOG OFF
RETURN

```

1.16.7 PRNSETUP

Use the PRNSETUP command to open the "Printer Setup" dialog box.



The PRNSETUP command allows you to control a Windows printer from within R:BASE. If you wish to use more than one option you will need to issue these as separate commands. If you change the printer using the 'printer name' your other print options will be reset so be sure to issue that command first.

It is worth noting that if you use a printer name that does not exist there will be no error and the current printer will not be changed.

Options

'PRINTER NAME'

Specify which printer to print to. This must match the name of the printer as it is displayed in the Windows Printer Control Panel Applet.

'PAPER SIZE'

Options:

LETTER	Letter 8 1/2 x 11 in
LETTERSMALL	Letter Small 8 1/2 x 11 in
TABLOID	Tabloid 11 x 17 in
LEDGER	Ledger 17 x 11 in
LEGAL	Legal 8 1/2 x 14 in
STATEMENT	Statement 5 1/2 x 8 1/2 in
EXECUTIVE	Executive 7 1/4 x 10 3/8 in
A3	A3 297 x 420 mm
A4	A4 210 x 297 mm
A4SMALL	A4 Small 210 x 297 mm
A5	A5 148 x 210 mm

B4	B4 (JIS) 250 x 354
B5	B5 (JIS) 182 x 257 mm
FOLIO	Folio 8 12 x 13 in
QUARTO	Quarto 215 x 275 mm
10X14	10x14 in
11X17	11x17 in
NOTE	Note 8 12 x 11 in
ENV9	Envelope #9 3 78 x 8 78
ENV10	Envelope #10 4 18 x 9 12
ENV11	Envelope #11 4 12 x 10 38
ENV12	Envelope #12 4 276 x 11
ENV14	Envelope #14 5 x 11 12
CSHEET	C size sheet
DSHEET	D size sheet
ESHEET	E size sheet
ENVDL	Envelope DL 110 x 220mm
ENVC3	Envelope C3 324 x 458 mm
ENVC4	Envelope C4 229 x 324 mm
ENVC5	Envelope C5 162 x 229 mm
ENVC6	Envelope C6 114 x 162 mm
ENVC65	Envelope C65 114 x 229 mm
ENVB4	Envelope B4 250 x 353 mm
ENVB5	Envelope B5 176 x 250 mm
ENVB6	Envelope B6 176 x 125 mm
ENVITALY	Envelope 110 x 230 mm
ENVMONARCH	Envelope Monarch 3.875 x eXtreme in
ENVPERSONAL	6 34 Envelope 3 58 x 6 12 in
FANFOLDUS	US Std Fanfold 14 78 x 11 in
FANFOLDSG	German Std Fanfold 8 12 x 12 in
FANFOLDLG	German Legal Fanfold 8 12 x 13 in
ISOB4	B4 (ISO) 250 x 353 mm
JPOSTCARD	Japanese Postcard 100 x 148 mm
9X11	9 x 11 in
10X11	10 x 11 in
15X11	15 x 11 in
ENVINITE	Envelope Invite 220 x 220 mm
LETTEREXTRA	Letter Extra 9 275 x 12 in
LEGALEXTRA	Legal Extra 9 275 x 15 in
TABLOIDEXTRA	Tabloid Extra 11.69 x 18 in
A4EXTRA	A4 Extra 9.27 x 12.69 in
LETTERTRANSVERSE	Letter Transverse 8 275 x 11 in
A4TRANSVERSE	A4 Transverse 210 x 297 mm
LETTEREXTRATRANSVERSE	Letter Extra Transverse 9 275
APLUS	SuperASuperAA4 227 x 356 mm
BPLUS	SuperBSuperBA3 305 x 487 mm
LETTER_PLUS	Letter Plus 8.5 x 12.69 in
A4PLUS	A4 Plus 210 x 330 mm
A5TRANSVERSE	A5 Transverse 148 x 210 mm
B5TRANSVERSE	B5 (JIS) Transverse 182 x 257 mm
A3EXTRA	A3 Extra 322 x 445 mm
A5EXTRA	A5 Extra 174 x 235 mm
B5EXTRA	B5 (ISO) Extra 201 x 276 mm
A2	A2 420 x 594 mm
A3TRANSVERSE	A3 Transverse 297 x 420 mm
A3EXTRATRANSVERSE	A3 Extra Transverse 322 x 445 mm

'PAPER SOURCE'

Specifies the paper source.

Options:

- UPPER
- LOWER
- MIDDLE
- MANUAL
- ENVELOPE
- ENVMANUAL
- AUTO
- TRACTOR
- SMALLFMT
- LARGEfmt
- LARGEcapacity
- CASSETTE

LANDSCAPE

Specifies to the landscape for the type of the report you want to print.

PORTRAIT

Specifies to the portrait for the type of the report you want to print.

COPY n

Specifies the number of copies to print.

COLLATE value

Specifies whether collation should be used when printing multiple copies. This parameter can be one of the following values:

- ON - Collate when printing multiple copies.
- OFF - Do not collate when printing multiple copies.

COLOR value

Switches between color and monochrome on color printers. Following are the possible values:

- ON
- OFF

DUPLEX

Select duplex or double_sided printing for printers capable of duplex printing. Following are the possible:

- SIMPLEX - Normal(nonduplex) printing
- HORIZONTAL - Short_edge binding, that is, the long edge of the page is horizontal.
- VERTICAL - Short_edge binding, that is, the long edge of the page is vertical.

QUALITY

Specifies the printer resolution. There are four values:

- HIGH
- MEDIUM
- LOW
- DRAFT

FONTs

Read Fonts to get a list of the fonts supported by the printer.

PRINTERS

Lists all printers installed in Windows.

TRAYS

Retrieves a list of the names of the paper bins.

PAPERnames

Retrieves a list of supported paper names (for example, Letter or Legal).

INFO

Retrieves a list of current printer settings like:

- Current Printer :
- Paper Orientation :
- Paper Size :
- Paper Source :
- Print Quality (DPI) :
- Copies :
- Color Mode :
- Duplex Mode :
- Collation :

Note: Only use options specific to your printer.

TOP - will move the print dialog location in pixels from the top of the screen down

LEFT - will move the print dialog location in pixels from the left of the screen down

OPTION THEMENAME

Will dynamically provide over thirty pre-defined themes

Values:

- Deep Cyan
- Deep Green
- Deep Orange
- Diamond Blue
- Diamond Green
- Funny Theme
- Gnome
- Green Glass
- Longhorn
- Mac OS
- Matrix
- Media
- One Blue
- One Clear Blue
- One Cyan
- One Green
- One Orange
- Orange Glass
- R:BASE Rocks!
- Razzmatazz
- Real One
- Sports Black
- Sports Blue
- Sports Cyan
- Sports Green
- Sports Orange
- Steel Black
- Steel Blue
- Win Native
- XP Blue
- XP Egypt
- XP Green
- XP Grey Scale
- XP Orange
- XP Silver

1.16.8 ProcessMessage

Use the **ProcessMessage** command to process messages that are currently in the windows message queue.

ProcessMessage

--Example 01:

```
SET VAR vPauseMessage TEXT = ((CHAR(013))+ 'Computing - Please Wait .....')
CLS
PAUSE 3 USING .vPauseMessage +
CAPTION ' Pause 3 with ProcessMessage' +
OPTION ICON_FILE Hourglass.GIF +
|BACK_COLOR WHITE +
|MESSAGE_COLOR WHITE +
|MESSAGE_FONT_NAME VERDANA +
|MESSAGE_FONT_COLOR RED +
|MESSAGE_FONT_SIZE 10 +
|THEMENAME R:BASE Rocks!
SET VAR vCounter INT = 1
WHILE vCounter < 2500000 THEN
SET VAR vCounter = (.vCounter + 1)
ProcessMessage
ENDWHILE
CLEAR VARIABLE vPauseMessage,vCounter
CLS
RETURN
```

--Example 02:

```
CLS
PAUSE 3 USING ' Calculating ... Please Stand By ...' +
CAPTION ' Pause 3 with Gauge + ProcessMessage' ICON APP +
OPTION GAUGE_VISIBLE ON +
|GAUGE_COLOR [R218,G228,B246] +
|GAUGE_INTERVAL 10 +
|MESSAGE_FONT_NAME VERDANA +
|MESSAGE_FONT_SIZE 10 +
|MESSAGE_FONT_COLOR BLUE +
|THEMENAME Razzmatazz
SET VAR vCounter INTEGER = 1
WHILE vCounter < 2500000 THEN
SET VAR vCounter = (.vCounter + 1)
ProcessMessage
ENDWHILE
CLEAR VARIABLE vCounter
CLS
RETURN
```

--Example 03:

```
CLS
PAUSE 3 USING ' Calculating ... Please Stand By ...' +
CAPTION ' Pause 3 with Gauge + ProcessMessage' ICON APP +
OPTION GAUGE_VISIBLE ON +
|GAUGE_COLOR 2708091 +
|GAUGE_INTERVAL 10 +
|MESSAGE_FONT_NAME VERDANA +
```

```

|MESSAGE_FONT_SIZE 10 +
|MESSAGE_FONT_COLOR RED +
|THEMENAME R:BASE Rocks!
SET VAR vCounter INTEGER = 1
WHILE vCounter < 2500000 THEN
SET VAR vCounter = (.vCounter + 1)
ProcessMessage
ENDWHILE
CLEAR VARIABLE vCounter
CLS
RETURN

```

--Example 04:

```

CLS
PAUSE 3 USING ' Calculating ... Please Stand By ...' +
CAPTION ' Pause 3 with Gauge + PM' +
ICON WINDOWS +
OPTION GAUGE_VISIBLE ON +
|GAUGE_COLOR RED +
|GAUGE_INTERVAL 10 +
|MESSAGE_FONT_NAME VERDANA +
|MESSAGE_FONT_SIZE 10 +
|MESSAGE_FONT_COLOR WHITE +
|THEMENAME Steel Blue
SET VAR vCounter INTEGER = 1
WHILE vCounter < 2500000 THEN
SET VAR vCounter = (.vCounter + 1)
ProcessMessage
ENDWHILE
CLEAR VARIABLE vCounter
CLS
RETURN

```

1.16.9 PROJECT

Use the PROJECT command to create a new table from an existing table or view.

```

PROJECT [ TEMPORARY ] tblname1 FROM tblview USING [ collist ] ...
               [ WHERE clause ] [ ORDER BY clause ]

```

Options

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

tblname1 FROM tblview

Tblname1 is the name of the new table you want to create, and FROM *tblview* specifies the existing table or view you want to copy.

TEMPORARY

Allows you to create a Temporary Table with the PROJECT command.

USING *

Specifies to use all columns with the command.

USING collist

Specifies the column(s) to use with the command.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the PROJECT Command

The new table can be a copy of an existing table, a copy of an existing table with the rows sorted in a different order, a duplicate of a table structure without any data, or a table that contains specific rows and columns from an existing table.

You must include the USING clause with the PROJECT command. The USING clause specifies the columns to be included in the new table. If you want the new table to include all the columns from an existing table, use an asterisk (*) in the clause. If you want the new table to include only specific columns from the existing table, list them in the order you want them to appear in the new table. If you want the new table to include all columns in a different order, list them in the order you want them to appear.

Working with Computed Columns

R:BASE copies the data from each column into the new table. If a computed column is included, R:BASE transfers the current values in the computed column to the new table. In order to calculate computed values in the column in the new table, R:BASE needs the column names used in the computed column's expression. Therefore, include those column names in the USING clause before the computed column. When you do not include those column names in the USING clause before the computed column, R:BASE makes the computed column a regular column, assigns a data type, and displays a message suggesting you rename the column in the new table. If you do not rename the column, the new table has a column with the same name as the column in the original table, but does not have the designation COMPUTED. You will not be able to use the [BACKUP](#) and [UNLOAD](#) commands, because you cannot have a computed column and a regular column with the same name.

Working with Autonumbered Columns

In a new table, R:BASE does not update the value in a row for the autonumber column. The autonumber column becomes a regular column.

Removing Columns and Rows from a Table

PROJECT is also useful if you want to remove several columns or rows from a table. To delete columns from a table, create a new table that retains the columns you want to keep, or to delete rows, create a new table using a [WHERE](#) clause. Use the [DROP](#) command to remove the table you no longer want, then use the [RENAME](#) command to give the new table the original table's name.

Transferring Default Column Definitions

Default column definitions are not transferred to a new table. When rows are added to the new table with the [INSERT](#) command, they are given a null value. If you want a default column definition, define the default column again.

Examples

The following command creates a new table that is a duplicate of the *employee* table.

```
PROJECT reps FROM employee USING *
```

The following command creates a table named *empty* that has the same structure as the *prodlocation* table but contains no rows of data.

```
PROJECT empty FROM prodlocation USING * WHERE LIMIT=0
```

The example below creates a table named *gt5year*. The order of the columns in the *gt5year* table are specified in the USING clause. The [WHERE](#) clause specifies that only the information for employees hired before January 1, 1984 will be selected. The [ORDER BY](#) clause sorts the rows in alphabetical order by the employees' last names.

```
PROJECT gt5year FROM employee USING empfname, emplname, +
empid, empext, hiredate WHERE hiredate < '01/01/84' +
ORDER BY emplname
```

1.16.10 PROPERTY

Use the PROPERTY command to dynamically change the properties of the form control object(s), the form itself, and the form table.

PROPERTY	ComponentID	ControlProperty	Value	_____
	RBASE_FORM	FormProperty	Value	_____
	RBASE_FORM_ACTION	CustomFormActionName	Parameters	_____
	TABLE	FormTableName	Value	_____
	APPLICATION	ApplicationProperty	Value	_____
	RBA_FORM	ApplicationFormProperty	Value	_____
	LOADWINDOW	WindowProperty	Value	_____
	ThemeParameter	ThemeName	Value	_____

Options

APPLICATION ApplicationProperty

To alter an application property such as the CAPS lock status, if it is compiled, or the title

Component ID

The unique identifier that you assign to the control (i.e. DB Edit). The unique identifier within the PROPERTY command must match the unique identifier listed in the control properties. A "Component ID" field is located in the "Object Properties" for all forms and form controls. A "Component ID" option is located in the speed menu list when you right click on a report/label control. **Wildcards are supported for altering the properties of multiple components.**

ControlProperty

The name of the form/report/label object's property you wish to change

LOADWINDOW WindowProperty

To alter the load window property such as the caption or title. The LOADWINDOW parameter is only supported in R:Compiler

Parameters

The optional value(s) you can pass onto the calling command, if required. Use ' ' blank space with single or double quotes (depending on your database QUOTE settings), if parameter is not required in calling command block

RBA_FORM ApplicationFormProperty

To alter an application form property such as the current theme, the enabled status for an action, or a caption

RBASE_FORM FormProperty

To alter a form property such as height, width, etc.

RBASE_FORM_ACTION CustomFormActionName

The actual name of the Command Name assigned when designing Custom Form Actions in the Form Designer

REPORT ReportProperty

To alter a report property such as the report name, the total page count, or the number of tables using

in the report

TABLE FormTableName

The driving table name of the form

ThemeName

Specifies the external theme name to load, release, or change. For more information on using the PROPERTY command to manage external themes, see External Themes.

ThemeProperty

Specifies the theme property to manage

Value

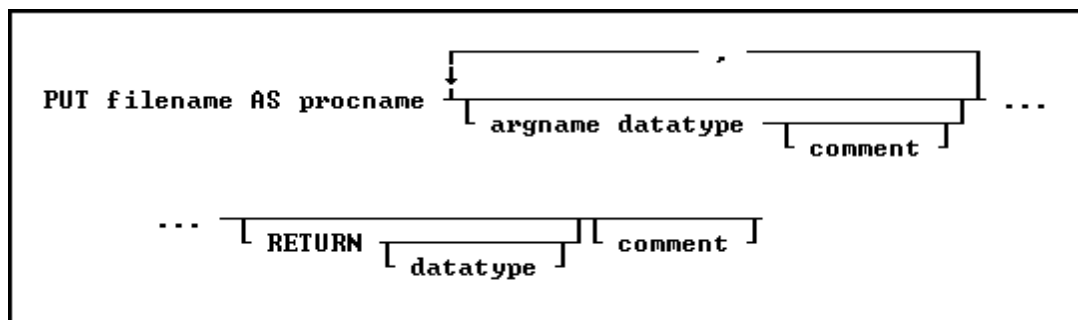
The new value to assign to the property of the object specified

Notes:

- Always use the current [QUOTES](#) character around the *Value* parameter, if the value is text.
- Use the following Web URL to create PROPERTY command syntax with the **PROPERTY Command Syntax Builder**:
<http://www.rbase.com/property/>
- An optional add-on product containing all PROPERTY/GETPROPERTY commands and parameters constructed in an R:BASE application is also available. The product is called [R:Docs](#). Please contact RBTI Sales Staff at sales@rbase.com if you wish to acquire the annual subscription to stand-alone R:Docs database and application.

1.16.11 PUT

Creates or replaces Stored Procedures into the database.



Options

argname datatype

The parameter name and datatype. This portion may be reported.

comment

An optional comment for the parameter or, if placed after RETURN, an optional comment for the entire procedure. The comment must be enclosed in the current quote setting.

filename

The filename in ASCII text format, with full path, to load as the Stored Procedure.

procname

Specifies the procedure name. If a procedure by this name already exists in the database, an error is generated. In R:BASE eXtreme 9.0 (32), the procedure name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the procedure name is limited to 128 characters.

RETURN datatype

Determines the datatype of the value returned by the procedure.

About the PUT Command

Argument List

When you load a Stored Procedure into a database, you specify arguments to be passed to it. These arguments are used within the procedure. When the procedure is called, the number and type of arguments passed must match the number and type specified when the procedure was stored in the database. When an argument name is referenced in the Stored Procedure code, the argument name must be preceded by a period unless it is a table or column name, then it must be preceded by an ampersand (&). For example:

```
UPDATE &p1 SET col = 99 WHERE col = .p2
```

The arguments names are specified when the procedure is stored in the database with the PUT command.

Return Values

The value to be returned by a Stored Procedure is specified in the procedure code following the keyword RETURN. For example, RETURN 'Los Angeles'. The value returned must match the datatype specified when the procedure was stored.

Replacing a Procedure

If you are replacing an existing procedure, you must LOCK the procedure first either with the [GET LOCK](#) or the [SET PROCEDURE](#) command. Once the procedure is locked, it is replaced by an updated file using the PUT command. A procedure cannot be replaced unless it is locked. A procedure is automatically unlocked when replaced with the PUT command.

Example

Use the PUT command as follows to store a command file as a Stored Procedure:

```
PUT INS.RMD AS SP_ContCheck p1 INT, p2 TEXT RETURN INTEGER
```

The contents of INS.RMD could be something like:

```
--INS.RMD
IF (.p1 > 105) THEN
  INSERT INTO contact (custid, contlname) VALUES (.p1, .p2)
  RETURN 1
ELSE
  RETURN 0
ENDIF
```

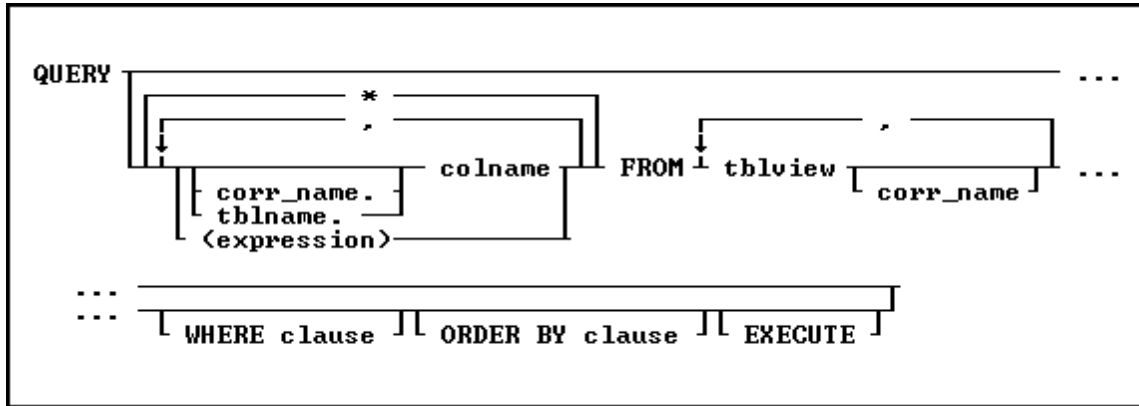
See Also:

Stored Procedures & Triggers

1.17 Q

1.17.1 QUERY

Use the QUERY command to select, combine, manipulate, and retrieve data from tables.



Options

*

Specifies all columns.

,

Indicates that this part of the command is repeatable.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter *#c*, where *#c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*). You can enter *tblname.** to specify all columns in the table.

corr_name

Correlation name. A nickname or alias for a table or view name. Use *corr_name* to refer to the same table twice within the command, or to more briefly specify a column in more than one table.

EXECUTE

Retrieves columns and their data, then displays the columns with their data in the Data Browser.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

FROM tblview

Specifies the name of the table(s) or view(s) to query.

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

tblname.

Specifies the table containing the columns. You can use *tblname.** to specify all columns in the table.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the QUERY Command

You construct a query to get information from your database's tables and columns. The process of selecting, retrieving, and working with data is referred to as creating and executing a query. R:BASE processes the query and displays what you request.

You can specify a maximum of ten tables and correlation names.

You can save the query as a view, or you can create a new table from the columns.

In R:BASE for Windows QUERY starts the **Query By Example** option from the **Tools** menu. After you enter the command QUERY, what R:BASE displays depends on the options you specified.

To display...	Do the following...
The "Query By Example" window	Use the QUERY command without specifying any options.
The "Query By Example" window and the query contained in the QUERY command	Use the QUERY command with any option except EXECUTE.
The Data Browser with the data retrieved by the QUERY command	Use the QUERY command with EXECUTE and any other option.

Examples

The following command chooses all the columns from the *customer* table and displays the columns' names and data on the screen.

```
QUERY * FROM customer EXECUTE
```

The next command selects the *empid*, *empfname*, and *emplname* columns from the *employee* table, and displays the column names on the screen.

```
QUERY empid, empfname, emplname FROM employee
```

The following command selects the data from the *custid* column in the *customer* table and the matching *netamount* columns in the *transmaster* table. *T1* and *t2* are correlation names. In the WHERE clause, the link is established between the two tables by comparing the values of the *custid* column between the two tables. The linking columns can have different names; however, they must have the same data type.

```
QUERY t1.custid, t2.netamount FROM customer t1, transmaster t2 +
WHERE t1.custid = t2.custid EXECUTE
```

1.17.2 QUIT

In a program, use QUIT to close all open program files, [IF/ENDIF](#) structures, [SWITCH/ENDSWITCH](#) structures, and [WHILE/ENDWHILE](#) loops.

QUIT TO filespec

Option

TO filespec

Specifies the command file that is to run after the QUIT command executes. QUIT TO another codelocked binary command file is also supported.

About the QUIT Command

When the QUIT command runs, it clears the nesting levels R:BASE maintains to control the following two items:

- Nesting command files called by the [RUN](#) commands.
- Levels for nesting [IF](#) structures, [SWITCH](#) structures, and [WHILE](#) loops.

Examples

The following command closes all open command files, IF structures, SWITCH structures, and WHILE loops, then you are returned to the R> Prompt.

```
QUIT
```

The command below closes all open command files, IF structures, SWITCH structures, and WHILE loops, and runs the command file named GETVAL.INV.

```
QUIT TO getval.inv
```

1.18 R

1.18.1 RBAPP

Use the RBAPP command to start the Application Designer with which you can create or edit RBA application files.

RBAPP

1.18.2 RBBEDIT

Use the RBBEDIT command to start the R:BASE BLOB Editor, with which you can create or edit binary files.

RBBEDIT [**filename.ext**]

You can use the RBBEDIT command to launch the R:BASE BLOB Editor for managing external files at the R> Prompt, in a command file, or in an EEP. Depending on the file extension, the BLOB Editor will display the available options accordingly.

Options

filename.ext

Specifies an external ASCII text, RTF, or image file. The R:BASE BLOB Editor recognizes the following image formats:

GIF	BMP
DCX	PCC
PCX	AFI
VST	TGA
PNG	JPEG
JPG	TIFF
TIF	ICO
EMF	WMF

Examples:

01.

The following will display the "Images" and "Multipage Images" tabs of the BLOB Editor for you to load an image.

RBBEDIT

02.

The following will display a bmp image in the "Images" tab of the BLOB Editor.

```
RBBEDIT myphoto.bmp
```

03.

The following will display the RTF file in the "RTF" tab of the BLOB Editor.

```
RBBEDIT myfile.rtf
```

04.

The following will display the text file in the "Note/VarChar" tab of the BLOB Editor.

```
RBBEDIT mytext.txt
```

You can also use the RBBEDIT command to launch the BLOB Editor to create a "new" RTF file.

05.

The following will display the "RTF" tab of the BLOB Editor for you to create a new RTF file.

```
RBBEDIT myfile.RTF
```

1.18.3 RBDEFINE

Use the RBDEFINE command to start the Database Designer.

RBDEFINE

About the RBDEFINE Command

Use the Database Designer to create or modify an existing table and its constraints, rules, indexes, and keys.

Alternative R:BASE Commands

Use the following commands as an alternative to the Database Designer:

- [CREATE SCHEMA](#)
- [CREATE TABLE](#)

Use the following R:BASE commands to enhance or modify a database:

- [ALTER TABLE](#)
- [COMMENT ON](#)
- [CREATE INDEX](#)
- [CREATE VIEW](#)
- [DELETE](#)
- [DROP](#)
- [LIST](#)
- [RENAME](#)
- [RULES](#)

1.18.4 RBEDIT

Use the RBEDIT command to start the R:BASE Editor with which you can create or edit text files and command files.

```
RBEDIT [ filespec [ SELECT VARCHAR clause ] ]
```

Options

filespec

Displays the contents of the file on the screen for editing. If the file does not exist, a blank screen is displayed. If you specify a wildcard with the *filespec* option, RBEDIT displays a list of all files that meet the condition. You can then choose a file from the list.

SELECT VARCHAR clause

Specifies a column defined with the VARCHAR data type from a table, then you can edit the contents. The [SELECT](#) clause must limit the data to only one row; otherwise, an error is returned.

About the RBEDIT Command

You can start the R:BASE editor by entering RBEDIT at the R> Prompt, or by choosing **R:BASE Editor** from the **Tools** menu in R:BASE.

Example

The following command starts REdit and displays the file named OLDFILE for editing. If OLDFILE does not exist, the editor creates a new file and names it OLDFILE.

```
RBEDIT oldfile
```

Use the following link to launch the complete documentation for the R:BASE Editor module. To launch the help from within the R:BASE Editor, press [Shift]+[F1].

1.18.5 RBLABELS

Use the RBLABELS command to create and edit labels from the R> Prompt.

```
RBLABELS [ labelname [ tblview ] ]
```

Options

labelname

Specifies the name of an existing label to modify.

tblview

This option is for DOS only; names the first table or view that a label serves. Use this option when you are creating a new label. This option is available only if you start to build a label from the R> Prompt with a database open.

Specify both *labelname* and *tblview* when you want to build a new label. R:BASE displays the Label Designer" screen where you can begin to define the new label. Use *labelname* alone to specify the name of an existing label you want to modify.

About the RBLABELS Command

You can display the Label Designer window by entering RBLABELS at the R> Prompt.

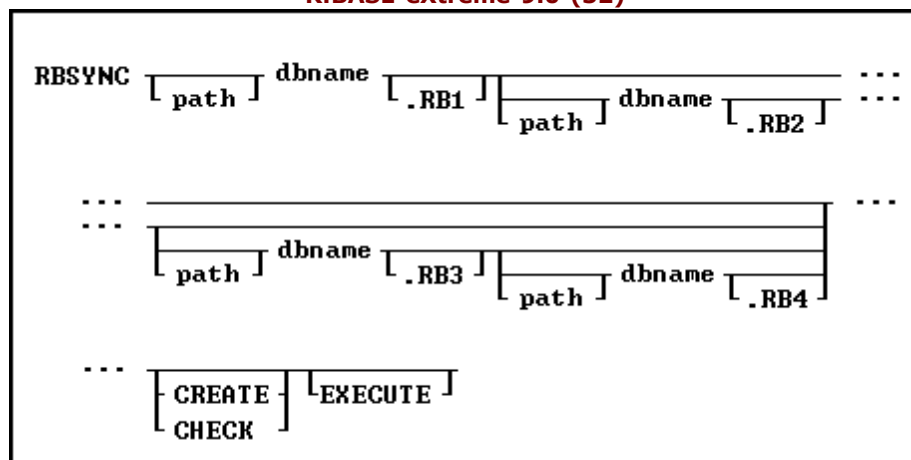
When creating labels, you can do the following:

- Build a label definition to create labels from data in a table or view.
- Set dimensions of a label definition to fit all popular label stock.
- Define variables and include them on your labels.
- Print labels on a printer, display them on screen, or send them to a file.

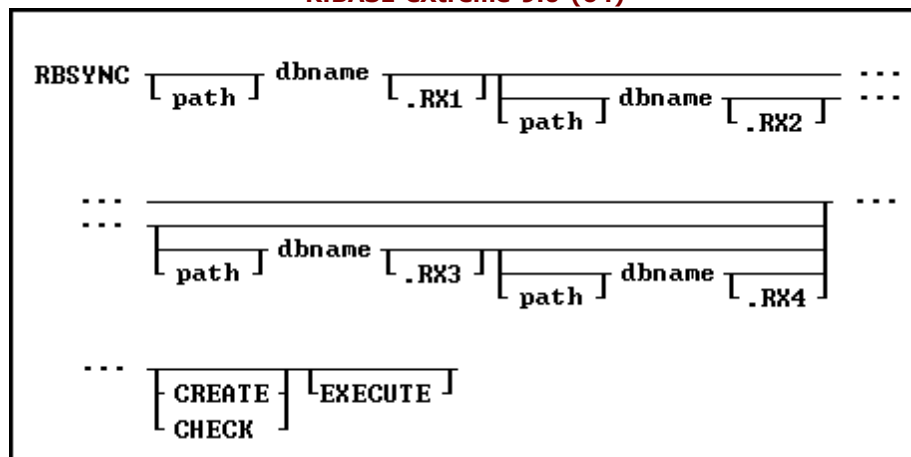
1.18.6 RBSYNC

Use the RBSYNC command to synchronize the timestamps for database files. Based upon your R:BASE eXtreme 9.0 version (32) or (64), you would use one of the follow syntax diagrams to synchronize your database files.

R:BASE eXtreme 9.0 (32)



R:BASE eXtreme 9.0 (64)



Options

dbname

Specifies the name of the database to synch.

path

Specifies the location of the database file(s).

.RB1

Specifies the .RB1 database file.

.RB2

Specifies the .RB2 database file.

.RB3

Specifies the .RB3 database file.

.RB4

Specifies the .RB4 database file.

.RX1

Specifies the .RX1 database file.

.RX2

Specifies the .RX2 database file.

.RX3

Specifies the .RX3 database file.

.RX4

Specifies the .RX4 database file.

CREATE

Creates a skeleton index file if it can't be found.

CHECK

Shows current synch status only. This is the default option unless you specify otherwise.

EXECUTE

Executes the command to synch the database.

About the RBSYNC Command

Use the RBSYNC command to synchronize the date and timestamps for the database files. The database files can become out of synch when R:BASE is not exited normally. You cannot open a database if the database files are out of sync.

With RBSYNC, a database owner can update the encoded timestamp values in the four database files to all have the same timestamp values. This might be necessary if R:BASE was aborted or if the computer or operating system crashes.

When a database is created or opened, a timestamp is encoded in each of the four files. Encoding the timestamp in each of the four database files enables the database files to be stored in separate directories apart from one another. To ensure that the correct database files are accessed, R:BASE evaluates the timestamp values of the files as it searches for them on the path statement. R:BASE must first try to connect to File 1, and then it searches the path for the other database files.

When you start RBSYNC, you can choose the location of File 1, enter the database owner's user identifier if one exists, then RBSYNC displays the location and timestamp for all four database files.

RBSYNC displays error messages if the database is already open or if any of the three files can't be located; however, if File 1 and File 2 are located but not File 3, the user has the option to create a new (empty) File 3 or abort. If creating a new File 3, RBSYNC displays a warning message stating that a PACK operation should be performed.

Notes:

- RBSYNC requires the EXECUTE parameter to actually perform the synchronization.
- You can implement the [AUTOSYNC](#) operating condition automatically synchronize the database files, if necessary.
- The *path*, *dbname*, and *.RX** options are rarely used except when the database files are in

different locations.

Example:

RBSYNC dbname EXECUTE

1.18.7 RECALC

Use the RECALC command in an entry/exit procedure (EEP) to recalculate expressions in a form.



Options

TABLES

Finds, recalculates, and redraws the lower-level tables to reflect the actions of the entry/exit procedure (EEP).

VARIABLES

Redisplays all variables for the current table.

About the RECALC Command

Without using the RECALC Command in EEPs-A recalculation takes place when you move to the field following the field that triggered the EEP. R:BASE automatically redisplay the value in a form field (variable) that calls the EEP. Redisplaying the value in a form field ensures that the field always contains the correct value.

To automatically redisplay a field, the following statements must be true:

- The EEP must be started from the field you want redisplayed.
- The field you want redisplayed must be a variable.
- The EEP loads a new value into a variable that is the same as the variable that started the EEP.

If you want to load the value into a column, create a form expression that equates the variable and the column, for example, *colname=.varname*.

Using the RECALC Command in EEPs-Use RECALC to ensure that expressions based on variables modified by an EEP are always current. RECALC only updates expressions when the following conditions apply:

- The field containing the EEP is a variable.
- The EEP changes the value of the variable.
- The EEP runs the RECALC command before returning to the form.
- At least one form expression has the variable to the right of the equal sign, which signifies a dependent expression.

When you use the RECALC command without any options, RECALC redisplay only the variable in the field that started the EEP, and recalculates only expressions using that variable. Use the VARIABLES option to force redisplay of all variables placed on the form and recalculation of all table expressions. Use the TABLES option to force recalculation of expressions in the other tables used in a multi-table form. The TABLES option recalculates lower tables only, not the current table.

Example

If you have an EEP in a form that calculates a value for a variable that you want to load into a column, follow the procedure below:

To calculate a value for a variable to load into a column:

1. Place the variable on the form.
2. Include the EEP in the field settings for the variable.
3. Create an expression that equates the column to the variable so that the variable is loaded into the column.
For example: colname1 = (.varname1)
4. Include RECALC or RECALC VARIABLES in the EEP.

For example, a procedure that calculates the sum from a column could be:

```
SELECT SUM colname2 INTO varname1 FROM tblname WHERE +  
colname3 = .varname2  
RECALC VARIABLES  
RETURN
```

1.18.8 RECOVER

Use the RECOVER command to start the Recover utility, which corrects errors that occur during transaction processing, for instance from a network or power failure.

RECOVER [**dbname**]

Options**dbname**

Specifies the database to recover.

About the RECOVER Command

You can use the RECOVER command to restabilize a database after a transaction is interrupted. The following conditions can interrupt a transaction:

- R:BASE discovers minor inconsistencies or unresolvable resource conflicts within a transaction. R:BASE automatically rolls back the transaction and displays an error message.
- Transaction processing has been interrupted unexpectedly. When you try to connect to the database, R:BASE detects either an existing Before Image file (a log of the commands issued in a transaction) or major inconsistencies. An error message tells you to use the RECOVER command.

Other users connected to the same database might get the same message, but not until they disconnect from the database and try to reconnect to it. The transactions entered by other users still connected to the database are executed properly unless they try to access system resources (such as tables and views) that are directly affected by the transaction error.

When transaction processing is on, R:BASE creates Before Image files for the current database. When you use the RECOVER command, it rolls back the interrupted transaction by using each Before Image file, clears all table and database locks, and resolves any other internal inconsistencies.

Before you use the RECOVER command, do the following:

- Back up your database and Before Image files. The Before Image files are in the same directory as the database and have extensions that begin with "B," such as DATABASE.B01.
- Make sure all users have exited from the affected database.

1.18.9 REFF

Use the REFF command to run the External Form Files.

REFF filename.rff

1.18.10 RELOAD

Use the RELOAD command to copy an open database without copying any unusable space.

RELOAD dbspec [WITH USER CASE]

Options

dbspec

Specifies the new database name.

WITH USER CASE

Incorporates case folding and collating tables defined in the user's configuration file (RBENGINE9.CFG) into the reloaded database.

About the RELOAD Command

Disk space becomes unavailable in a database when the following actions are performed:

- Deleting rows or indexes.
- Removing tables or columns.
- Adding columns or modifying tables with the [ALTER TABLE](#) command.

RELOAD copies a database table by table, and places all rows for each table in a single area on the disk, which improves database-response time.

When you use RELOAD to reload a database on the same disk and directory as the original database, enter a different name for the new database. When you reload a database from a different disk or directory onto the current disk, you can use the same database name for the copy. Be sure to specify the new drive or directory when you enter the command.

If there is not enough available disk space to copy a database using the RELOAD command, use the [PACK](#) command instead. Back up the database before packing. PACK eliminates unused space in a database; however, PACK does not rearrange the rows-only the RELOAD command rearranges rows.

RELOAD is available when [MULTI](#) is set on and a user has not set any locks on the database.

RELOAD is unavailable when RELOAD is unavailable when is on.

When you reload data that has a NOTE data type, the rows are adjusted according to the current setting of the [SET NOTE PAD](#) command.

RELOAD is unavailable when [STATICDB](#) is set on, which activates a read-only schema mode.

Database Access Rights with RELOAD

When access rights for a table have been assigned using the [GRANT](#) command, RELOAD requires the database owner's user identifier to RELOAD a database.

Example

The following command reloads an open database and gives it the name *newbase* in the RBASE directory

of drive C:.

RELOAD c:\rbase/newbase

1.18.11 RENAME

Use the RENAME command to change a form, report, label, table, view, or column name, and the database owner's user identifier. You can also use RENAME to change the name of an existing file.

```

      COLUMN colname1 TO colname2 [ IN tblname ]
      TABLE tblname1 TO tblname2
      VIEW viewname1 TO viewname2
      FORM formname1 TO formname2
      LABEL labelname1 TO labelname2
      PROCEDURE procname1 TO procname2
      REPORT rptname1 TO rptname2
      OWNER ownername1 TO ownername2
      NOCHECK

RENAME filespec filename
  
```

Options

COLUMN colname1 TO colname2

Renames a column in one table or in all tables in the open database.

filename

Specifies the new name of the file.

filespec

Specifies the file you want to rename. Optionally, include a drive and path specification in the form D:\PATHNAME/FILENAME.EXT.

FORM formname1 TO formname2

Renames a form in the open database.

IN tblname

Specifies the table in which you want to rename a column.

LABEL labelname1 TO labelname2

Renames a label in the open database.

NOCHECK

Does not update references to views, tables, and columns in forms, reports, labels, access rights, and rules.

OWNER ownername1 TO ownername2

Renames an owner in the open database.

PROCEDURE procname1 TO procname2

Renames a Stored Procedure in the open database.

REPORT rptname1 TO rptname2

Renames a report in the open database.

TABLE tblname1 TO tblname2

Renames a table in the open database.

VIEW viewname1 TO viewname2

Renames a view in the open database.

About the RENAME Command

If you do not want R:BASE to update references to views, tables, and columns when you rename them, include the NOCHECK option with the command.

Renaming Columns

You can rename a column in an entire database or in a single table. R:BASE does not update column references in rules. When you rename a column, R:BASE automatically updates references to the column in the following instances:

- If the column has a description.
- If it is used in a form, report, label, computed column, [UPDATE access right](#), or autonumbered column. However, because of possible size problems, R:BASE does not change column references inside an expression in a form, report, or label. These column references must be modified manually through the Form, Report, or Label Designer.

Updating Views and Tables

R:BASE automatically updates references to views and tables in the following instances:

- If you rename a view used in a report, label, or access right.
- If you rename a table that has a description, or is used in a form, report, label, or access right.

R:BASE does not update table or column references in views.

To update a view, delete it with the [DROP](#) command and define it again with the [CREATE VIEW](#) command, or QBE. To update a rule, you can use the [RULES](#) command, or the **Database Designer**. If you use the RULES command, you must first delete the rule with the DROP command and then add it again with the RULES command.

Renaming Tables

When you rename a table that is used as the rule table in a data-entry rule, R:BASE updates the rule definition. However, if you rename a table used in the WHERE clause of a rule definition, you must update the rule yourself. R:BASE does not update table references in views.

Renaming Files

When you use RENAME to change the name of a disk file, only the name of the file is changed. The file remains in the same directory on the same drive. You can include a file specification for the file you are renaming but not for the new file name. If you want the file to reside in a different drive or directory, use the [COPY](#) command. This command is similar to the operating system command RENAME.

On a workstation with multiple drives (local or mapped), especially when the files are on the different drive, it is always the best practice to define a drive letter when copying, deleting, renaming or running files, unless the specified files are located in the working directory. You will not need to specify the drive letter if all of the files are located in the default directory when using the copy, delete, rename or run commands.

Updating Command Files

R:BASE also does not update column, table, view, form, report, or label references in command files or applications. To update command files, use REdit or another text editor. To update applications, use the Application Designer.

Assigning User Identifiers

You can assign or change the database owner's user identifier with RENAME. The default user identifier is PUBLIC. Until this default is changed, any user can modify the database structure, read, enter, change, or delete data.

An owner's user identifier can be a maximum of eighteen characters. It must begin with a letter, and can contain letters, numbers, and the symbols #, \$, _, and %, and must be unique among all user identifiers.

Examples

The following command renames a column from *transid* to *transxno* in the *transmaster* table.

```
RENAME COLUMN transid TO transxno IN transmaster
```

The following command changes the database owner's user identifier from the default PUBLIC to *june*.


```
RENAME OWNER PUBLIC TO june
```

The following command changes the name of the CUSTOMER file to CUSTOMER.DAT.

```
RENAME customer customer.dat
```

The following command renames all four R:BASE eXtreme 9.0 (32) database files to NYC.RB1, NYC.RB2, NYC.RB3, and NYC.RB4.

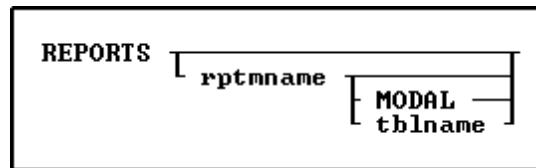
```
RENAME newyork.rb? nyc.rb?
```

The following command renames all four R:BASE eXtreme 9.0 (64) database files to NYC.RX1, NYC.RX2, NYC.RX3, and NYC.RX4.

```
RENAME newyork.rx? nyc.rx?
```

1.18.12 REPORTS

Use the REPORTS command to start the Report Designer.



Options

MODAL

Allows an EEP command block to launch a Form, Label or Report in a designer. After making appropriate changes, you will have to save the changes and close the designer in order to continue the next command in your EEP.

rptname

Specifies an existing report whose definition you want to modify.

tblview

This options is for DOS only; specify both *rptname* and *tblview* when you want to build a new report. R:BASE displays the "Report Designer" window where you can begin to define the new report. Use *rptname* alone to specify the name of an existing report you want to modify.

About the REPORTS Command

When designing a report, you can do the following:

- Define a Quick or a Custom report that prints or displays data from a table or view.
- Group data by unique column values.
- Define variables and include them in the report.
- Look up values from columns in other tables.
- Draw lines, borders, 3D boxes, and standard boxes on the report.
- Print a report to a printer, the screen, or a disk file.
- Print summary information, such as subtotals.
- Create display masks for data.
- Place a graphic on a report.

1.18.13 RESETROW

Use the RESETROW command to reset the current row of data currently being edited or entered in a form. This will cause the data to revert to it's original format and content.

RESETROW

1.18.14 RESTORE

Use the RESTORE command to restore part or all of a database from a file created with [BACKUP](#).

RESTORE filespec

Option

filespec

Specifies a file name with a drive and path specification in the form D:\PATHNAME/FILENAME.EXT.

About the RESTORE Command

If you are restoring a backup copy from a floppy disk, insert the disk in the floppy drive. If there is more than one backup disk, insert the disk labeled 1 first. RESTORE prompts you for the appropriate disk and allows you to confirm each disk. If you backed up data and structure separately, you can choose not to restore any subsequent disks. RESTORE appends data to the table if it exists in the database.

If your database becomes damaged after your last backup and any of the .RB*n* files of the database remain, you can rename the database files, restore the database on a different directory, or erase them before you restore the database from the backup.

When [STATICDB](#) is set on-which activates a read-only schema mode-RESTORE is unavailable.

To use RESTORE, the RESTORE.RMD file must be in the directory containing the R:BASE program files. If you cannot locate the RESTORE.RMD file in your R:BASE program directory, then save the following as the file.

```
--RESTORE.RMD
CLEAR VARIABLES YesNo, OldQuote, Quote
SET VARIABLE Quote TEXT = QUOTES
SET VARIABLE OldQuote = (CVAL(.Quote))
SET QUOTES='
SET VARIABLE YesNo = 'Y'
WHILE YesNo <> 'E' THEN
    BEEP
    SET QUOTES='
    WRITE 'Please insert disk into drive...'
    FILLIN YesNo USING 'Press Enter to continue, or S to stop operation. '
    IF YesNo = 'S' THEN
        SET QUOTES = .OldQuote
        PAUSE 1
        RETURN
    ENDIF
    INPUT .%1
ENDWHILE
PAUSE 1
RETURN
```

Examples

This command restores the *concomp* database from the CONCOMP.BUP file on drive A:.

```
RESTORE a:concomp.bup
```

The following command lines open the database and restore the TBLNAME.BUP file on drive A: that contains a backup of a table. R:BASE generated the TBLNAME.BUP file when you used the [BACKUP](#) DATA command.

```
CONNECT dbname
RESTORE a:table.bup
```

1.18.15 RETURN

Use the RETURN command in a program to return control to a program or to the R> Prompt.

RETURN `[.varname]`

Options

.varname

Used ONLY within a Stored Procedure to return a value. The returned value will be stored in the STP_RETURN system variable. This option will return an error when used outside of a Stored Procedure. The default is text 8 characters, but if you want more, set it to a larger value.

[PUT](#) Command Example (sets the limit for the return value at 30 characters):

```
PUT test.prc AS TEXT test p1 integer RETURN TEXT (30)
```

About the RETURN Command

When you use RETURN to return to a command file or a block in a procedure file, the next command executed is the command following [RUN](#) in the command file or procedure file.

If you used the RUN commands to run a command file or a block in a procedure file from the R> Prompt, control returns to the R> Prompt when RETURN is executed.

Examples

Assume you have two command files named CMDFIL1 and CMDFIL2 with the commands shown below.

```
--CMDFIL1
WRITE 'This command file (cmdfil1) calls cmdfil2'
RUN cmdfil2
CLS
WRITE 'Cmdfil1 is in control.'
*(CMDFIL2)
LABEL top
FILLIN vend=0 USING 'Press Q to quit: '
IF vend = 'Q' THEN
    RETURN
ELSE
    GOTO top
ENDIF
```

When you run CMDFIL1 by entering RUN CMDFIL1, the following messages appear on the screen:

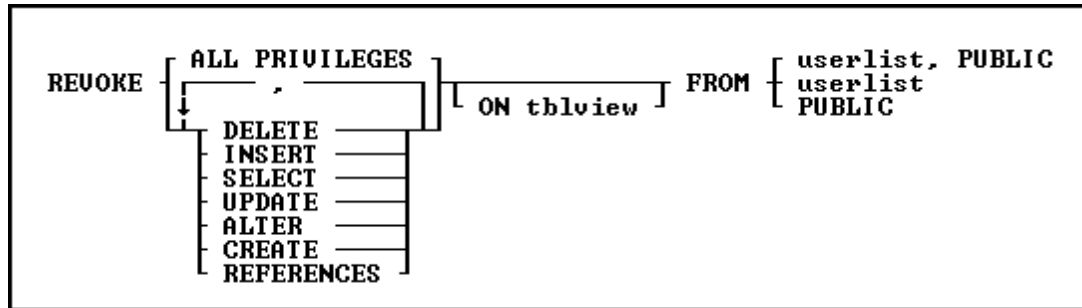
```
This command file (cmdfil1) calls cmdfil2
Press Q to quit:
```

If you press [Q] at the [FILLIN](#) command prompt, control returns to CMDFIL1. The following message appears on screen, processing stops, and you are returned to the R> Prompt:

Cmdfil1 is in control.

1.18.16 REVOKE

Use the REVOKE command to remove privileges provided to users with the [GRANT](#) command.



Options

,

Indicates that this part of the command is repeatable.

ALL PRIVILEGES

Removes all user privileges granted for all tables and views or for one table or view.

ALTER

Removes permission from users to modify the structure of all tables or specified tables.

CREATE

Removes permission from users to create new tables. Do not specify any tables or views when removing this permission.

DELETE

Removes permission to remove rows from all tables and views, or from a specified table or view.

FROM PUBLIC

Specifies PUBLIC. If, for example, *ralph*, *sam*, *jane*, and PUBLIC have been granted certain user privileges, revoking privileges from PUBLIC would not affect the three listed users.

FROM userlist

Specifies individual users whose access is to be revoked. You must separate user identifiers with a comma (or the current delimiter).

FROM userlist, PUBLIC

Specifies both individual users and PUBLIC, whose access is to be revoked. You must separate user identifiers with a comma (or the current delimiter).

INSERT

Removes permission to add rows to all tables and views or to a specified table or view.

ON tblview

Specifies a table or view from which to remove user privileges.

REFERENCES

Removes permission to create a table with a foreign key that references a table with a primary key.

SELECT

Removes permission to view and print data from all tables and views, or from a specified table or view.

UPDATE

Removes permission to change the value of all columns on all tables and views, or on a specified table

or view. You cannot specify columns when revoking UPDATE permission.

About the REVOKE Command

If you are the owner of a database, you can revoke any user privileges granted to users. If the database owner or other users have assigned you user privileges with the [WITH GRANT OPTION](#), you can revoke only the user privileges that you have granted to other users.

To remove the WITH GRANT OPTION, you must first revoke the privilege(s) to which the WITH GRANT OPTION has been assigned. Use the [LIST ACCESS](#) command to display a list of user privileges. An asterisk before an user privilege indicates that the WITH GRANT OPTION has been assigned to that user privilege, for example, *UPDATE.

REVOKE ALL PRIVILEGES revokes all user privileges that have been granted. However, REVOKE combined with ALTER, CREATE, DELETE, INSERT, REFERENCES, SELECT or UPDATE only applies to those privileges.

You can remove more than one user privilege in a REVOKE command. Separate the user privileges with a comma (or the current delimiter). You can also revoke user privileges using the **User Privileges** option from the **Utilities** menu in R:BASE for Windows. In R:BASE for DOS, choose **Databases: Create/modify**, choose **Modify: database name**, then choose **Access Rights: Revoke**.

Examples

Assume that the following sequence of [GRANT](#) commands represents all the user privileges granted for the *concomp* database.

```
GRANT INSERT ON employee TO ralph, sam
GRANT SELECT, INSERT ON transmaster TO jane WITH GRANT OPTION
GRANT UPDATE ON transmaster TO sam
GRANT UPDATE (company, custaddress, custcity) ON customer TO sam, PUBLIC
```

The following command revokes permission granted to *janeto* display or print data, or add rows to the *transmaster* table.

```
REVOKE SELECT, INSERT ON transmaster FROM jane
```

The following command revokes the UPDATE user privilege granted to *sam* for all tables and views in the database.

```
REVOKE UPDATE FROM sam
```

The following command revokes all user privileges granted to *sam*, except those granted to him as a member of PUBLIC.

```
REVOKE ALL PRIVILEGES FROM sam
```

The following command revokes all user privileges for all tables and views for all users.

```
REVOKE ALL PRIVILEGES FROM sam, jane, ralph, PUBLIC
```

1.18.17 RHELP

Included with R:BASE 2000 (version 6.5++) is the external file RHelp.EXE. This file can be called using the [ZIP](#) command and will display the contents of the file specified. Users will not be able to edit the file displayed. This window can be Minimized, Normalized and Maximized with an option to Close [x]. The RHelp.EXE is no longer provided with current versions of R:BASE.

ZIP RHELP filespec

Options

filespec

Displays the contents of the file on the screen. This may be either a plain text file or a Rich Text Format (RTF) file. If the file name includes spaces you must enclose the *filespec* in either single or double quotes.

Example

The following command starts RHelp and displays the file named RichFile.RTF for viewing.

```
ZIP RHELP C:\RBTI\RichFile.RTF
```

The following example displays the Long File Named file "This is a Test.RTF" from a network share "Users" on the server "InHouse".

```
ZIP RHELP "\\InHouse\Users\This Is a Test.RTF"
```

1.18.18 RHIDE

To hide the background R> Prompt window when using modal forms

RHIDE

The RHIDE command is also supported in EEPs.

1.18.19 RMDIR

Use the RMDIR command to remove a directory from a drive.

RMDIR d: **pathname**

Options

d:

Specifies a drive letter.

pathname

Specifies one or more directory names that lead from the root directory to a specified directory.

About the RMDIR Command

The R:BASE RMDIR command is similar to the operating system RMDIR command. For more information about the RMDIR command, refer to your operating system manual.

You cannot remove the root directory or the current directory. Before you can remove a directory, all files and directories in that directory must be removed. To delete files from a directory, use the [ERASE](#) command. When a file path includes more than one directory, the last directory listed is removed.

Examples

The following command removes the OLDDATA directory from the current drive.

```
RMDIR olddata
```

The following command removes the OLDDATA directory from the /SALESDEP/YTDSALES file path on

drive C:.

RD c:\salesdep\ytdsales\olddata

1.18.20 ROLLBACK

Use the ROLLBACK command to cancel the changes made to the data or database structure made since the beginning of the transaction, when using transaction processing.

ROLLBACK [**WORK**]

Options

WORK

WORK is optional and has no effect in R:BASE.

About the ROLLBACK Command

A transaction consists of all the commands executed since the last [COMMIT](#) or ROLLBACK command, or since you connected to the database if a COMMIT command has not been executed.

If you want to be able to process a group of commands, or transaction, without permanently affecting your data until you accept the changes, turn the [AUTOCOMMIT](#) setting off. R:BASE stores the information needed to reverse each modification made to the database in a Before Image file.

To reverse a transaction and undo the changes, enter the ROLLBACK command after you have entered the transaction. The ROLLBACK command also releases all automatic table and database locks. If you want to make the changes permanent, enter the COMMIT command.

This command is only valid when you open a database with [TRANSACTION](#) set on.

1.18.21 RPHONE

The RPHONE command uses the Windows Phone Dialer to dial a call. You must have a modem installed in your computer.

RPHONE 'phonenumber' ['nameforwindowslog' ['comment']]

Options

nameforwindowslog

Optionally pass in the name of the party you are calling to the Windows Dialer call log.

comment

Optionally pass a comment about the phone call into the Windows Dialer call log.

About the RPHONE Command

RPHONE command uses the Windows Phone Dialer to dial a call. You must have a modem installed in your computer.

RPHONE command requires the number that you want to call. You may optionally pass in the name of the party you are calling and a comment about the call. This information will be placed in the Windows Dialer call log if the call is successful.

The RPHONE command is also supported in EEPs.

Examples

Example 01:

```
RPHONE '(724) 733-0053'
```

Example 02:

```
RPHONE '(724) 733-0053' 'R:BASE Technologies' 'Pre-Order extreme'
```

Example 03:

```
SELECT CustPhone, Company INTO +
vCustPhone INDIC IvCustPhone, +
vCompany INDIC IvCompany +
FROM Customer WHERE CustID = .vCustID
RPHONE .vCustPhone .vCompany 'Follow-Up'
```

1.18.22 RSHOW

To bring back the background R> Prompt window when using modal forms

RSHOW

The RSHOW command is also supported in EEPs.

1.18.23 RULES

Use the RULES command to regulate data entry in a database.

```
RULES 'message' FOR tblname [DELETE] [SUCCEEDS] [FAILS] WHERE clause
```

Options

DELETE

- **DELETE SUCCEEDS** deletes a row from a database when the conditions in the [WHERE](#) clause are met.
- **DELETE FAILS** deletes a row from a database when the conditions in the WHERE clause are not met.

FOR tblname

Specifies the name of the table for which you are defining rules.

FAILS

Specifies that a row must not meet the conditions included in the WHERE clause in order to be added to the database.

'message'

Specifies a message to be displayed when a rule is violated.

SUCCEEDS

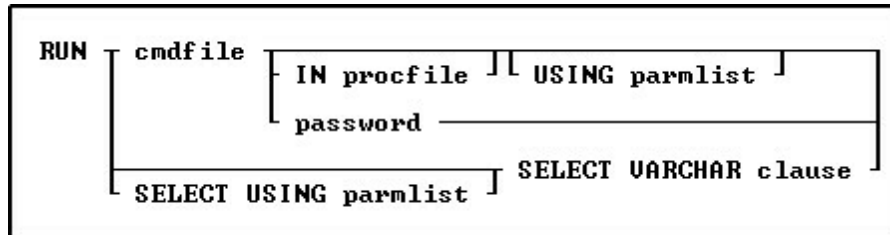
Specifies that a row must meet the conditions included in the WHERE clause in order to be added to the database.

WHERE clause

Limits rows of data. For more information, see the WHERE command.

1.18.24 RUN

Use the RUN command to run command blocks, command files, and command files requiring passed parameters.

**Options****cmdfile**

Specifies the name of the command block or command file to execute.

On a workstation with multiple drives (local or mapped), especially when the files are on the different drive, it is always the best practice to define a drive letter when copying, deleting, renaming or running files, unless the specified files are located in the working directory. You will not need to specify the drive letter if all of the files are located in the default directory when using the copy, delete, rename or run commands.

IN procfile

Specifies the name of a procedure file. A procedure file is a compiled binary file that contains stored menu, screen, and command blocks. Include a drive and path name if the procedure file is not on the current drive and directory.

password

Specifies the file password if encryption was specified when the file was saved or created. See [OUTPUT](#) for the command syntax to encrypt database information output. See R:BASE Editor for information on saving encrypted command files.

SELECT VARCHAR clause

Specifies a column defined with the VARCHAR data type from a table, then you can run the contents. The SELECT clause must limit the data to only one row; otherwise, an error is returned.

USING parmlist

Lists the values the command file uses when it runs. The parameter list can contain up to 18 values. The first value in the list is referenced in the executed file as %1, the second as %2, and so on through %9. They are treated just like other variables. To reference the contents of these variables, preface the variable name with a dot (.); for example, set v1 =.%1.

About the RUN Command

The RUN command must be on a line by itself and not combined with other commands.

Examples

The following command runs a file named MYCMD.COM in the current working directory.

```
RUN mycmd.cmd
```

The following command runs a command block named *mycmdfil* in the MYPROCFL.APX procedure file.

```
RUN mycmdfil IN myprocfl.apx
```

The following command executes the *mycmdfil* command block in the MYPROCFL.APX procedure file,

```

RUN SELECT CmdData FROM IntrnlCmd WHERE CmdName = 'NewKey'

```

If the foreign table has no primary or unique key, specifies the column(s) that uniquely identify the rows in the table.

About the SATTACH Command

Before you can attach a foreign data source table, an R:BASE database must be connected. Also, your workstation must be connected to the data source.

If you use SATTACH without the *tblname* option, R:BASE displays the "**Attach Table(s)**" dialog box with names of tables in the data source. You can then select a table to attach.

After you have exited R:BASE or disconnected the database to which the foreign table is attached, you don't need to reconnect to the table's data source when you open the database again. The data source is connected when you use the attached table. The data source table remains attached until you detach it with the [SDETACH](#) command, or use the **Utilities: Detach SQL Database Table** menu option in R:BASE for Windows.

When a foreign table is attached, R:BASE writes a table description that identifies the table as a data source table and names its data source. Use the [LIST TABLES](#) *tblname* command to review table descriptions.

Notes:

- SATTACH requires an owner password if one has been defined, or permission to create tables.
- When you attach a foreign table, R:BASE only includes the columns with legal names. For example, R:BASE does not include columns that have spaces in the name, or column names that exceed the character limit for the R:BASE version installed. R:BASE eXtreme 9.0 (32) table names and column names are limited to 18 characters. R:BASE eXtreme 9.0 (64) table names and column names are limited to 128 characters.
- When you attach a foreign table and select the columns that uniquely identify its rows in the "Select Column Set" dialog box, do not select columns that have LOB data types--unpredictable results might occur.
- When running applications that connect to foreign data sources, you should always disconnect from the R:BASE database before running the application again.

Notes for [Alias] Parameters:

- Syntax has been extended to allow you to specify only the changed columns. For example, if you only need to alias the second column you can use ...alias ,,location,,
- Any missing alias names will use the default name.
- If Qualkey defined as "using", it no longer prompts when you supplied already.
- If there is a conflicting column name, you will be prompted with a new -ERROR- message (3069). It basically states that: "column xxx will not be attached" When a name is too long it should be obvious why it is asking for an alias.
- If the name conflicts with another name then you get the error message first explaining the conflict, then the dialog box.
- If no qualkey is specified, automatic qualkeys will be assigned based on information from the ODBC source

Examples:

The Following scenarios will help you understand the use of enhanced SATTACH command in R:BASE for Windows.

Scenario A:

When the foreign data source database includes the same table name as in your database, you need to attach the table with a different name.

Commands to Use: [SCONNECT](#) and SATTACH

Syntax:

SCONNECT DataSourceName IDENTIFIED BY UserID

Where:

DataSourceName is the foreign data source to connect

IDENTIFIED BY UserID specifies the user identifier. When access rights have been assigned using the [GRANT](#) command, you must enter an appropriate user identifier with the SCONNECT command unless permission has been granted to PUBLIC.

Syntax:

SATTACH TableName AS AliasTableName USING collist

Where:

TableName is the table to attach

AS AliasTableName to specifies an alias, or temporary name, for the foreign table.

USING collist, if the foreign table has no primary or unique key, specifies the column(s) that uniquely identify the rows in the table.

Example 01.

```
SET ERROR MESSAGE 2038 OFF
SDISCONNECT DataSourceName
SCONNECT DataSourceName IDENTIFIED BY NONE
SDETACH tCustomers NOCHECK
SATTACH Customers AS tCustomers USING CustomerID
```

OR

```
SET ERROR MESSAGE 2038 OFF
SDISCONNECT DataSourceName
SCONNECT DataSourceName IDENTIFIED BY password
SDETACH tCustomers NOCHECK
SATTACH Customers AS tCustomers USING ALL
```

Scenario B:

When the foreign data source database includes the same table name as in your database, you will need to attach the table with a different name. The same applies conflicting column names that use data types where you must rename the column name to avoid any conflict with data types.

Example 02:

```
SET ERROR MESSAGE 2038 OFF
SDISCONNECT DataSourceName
SCONNECT DataSourceName IDENTIFIED BY NONE
SDETACH tOrders NOCHECK
SATTACH Orders AS tOrders USING +
OrderID ALIAS +
OrderID, +
CustomerID, +
EmployeeID, +
OrderDate, +
RequiredDate, +
ShippedDate, +
ShipVia, +
```

```
tFreight, +
ShipName, +
ShipAddress, +
ShipCity, +
ShipRegion, +
ShipPostalCode, +
ShipCountry
```

Notice that the table Orders will be attached as tOrders, column OrderID will be used AS Primary Key and the column Freight will be an alias tFreight to avoid any conflict.

If you know the exact column sequence and the column you would like to alias, you can use the following simple version of the same command:

Example 02A:

```
SET ERROR MESSAGE 2038 OFF
SDISCONNECT DataSourceName
SCONNECT DataSourceName IDENTIFIED BY NONE
SDETACH tOrders NOCHECK
SATTACH Orders AS tOrders USING +
OrderID ALIAS ,,,,,,,tFreight,,,,,
```

Notice that the table Orders will be attached as tOrders, column OrderID will be used as the Primary Key and instead of defining individual columns, you can start the list of columns with comma, add comma for each column and the AliasColumn, such a tFreight, to be aliased, and then continue with a comma for each additional column.

In the above example we have eight (8) commas, then AliasColumn and an additional six (6) commas.

There are a total of fourteen (14) columns in tOrders table.

So, start with first comma, seven commas for first seven columns, AliasName, and then additional six commas for columns nine through fourteen.

Scenario C:

When the foreign data source database includes the table name with spaces, you need to attach the table with a different name.

Example 03:

```
SET ERROR MESSAGE 2038 OFF
SDISCONNECT DataSourceName
SCONNECT DataSourceName IDENTIFIED BY NONE
SDETACH tOrderDetails NOCHECK
SATTACH `Order Details` AS tOrderDetails USING +
OrderID, +
ProductID
```

Notice that the table `Order Details` with spaces is surrounded by [IDQUOTES](#) and will be attached as an alias table tOrderDetails, column OrderID will be used as Primary Key.

1.19.2 SAVEROW

Use the SAVEROW command in an entry/exit procedure (EEP) to save a row.

SAVEROW

About the SAVEROW Command

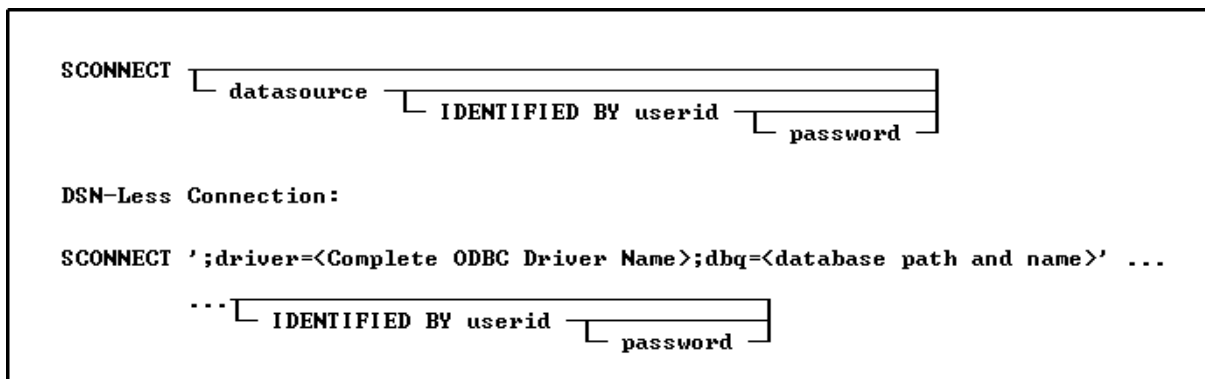
Use the SAVEROW command to save changes to the row of data currently being edited or entered in a form.

SAVEROW is not the same as **Save Changes** option on the **Forms** menu used in R:BASE for DOS, which saves the edits to the data then displays the next row. To perform the same functions as this menu option, use the SAVEROW command followed by the [NEXTROW](#) command.

The SAVEROW command, used in an exit procedure, leaves you in the field from which it was called. To move to the next field, add the command [SKIP 1](#) to the EEP.

1.19.3 SCONNECT

Use the SCONNECT command to connect R:BASE to a foreign data source.



Options

datasource

Specifies the name of the foreign data source that contains the table you want.

IDENTIFIED BY userid

Specifies your user account name for the foreign data source. You can use " (two single quotes) in this position if there is no User ID.

password

Specifies your password for the foreign data source. You can use " (two single quotes) in this position if there is no password.

About the SCONNECT Command

If you omit the *datasource* option, a dialog box opens, listing data sources from which to choose and continues with prompts for *username* and *password*.

About the DSN-Less Connection

A data source name (DSN) is a data structure that contains the information about a specific database that an Open Database Connectivity (ODBC) driver needs in order to connect to it. Included in the DSN, which resides either in the registry or as a separate text file, is information such as the name, directory and driver of the database, and, depending on the type of DSN, the ID and password of the user. The developer creates a separate DSN for each database. To connect to a particular database, the developer specifies its DSN within a program. In contrast, DSN-less connections require that all the necessary information be specified within the command. DSN-Less connection requires no server setup, just a carefully constructed connection string.

There are three kinds of DSN: user DSNs (sometimes called machine DSNs); system DSNs; and file DSNs. User and system DSNs are specific to a particular computer, and store DSN information in the registry. A user DSN allows database access for a single user on a single computer, and a system DSN

for any user of a particular computer. A file DSN contains the relevant information within a text file with a .DSN file extension, and can be shared by users of different computers who have the same drivers installed.

DSN-less connections demand that you know the name of the file (i.e. file based databases like R:BASE) or the address of the data server (SQL Server for example).

Armed with appropriate information you could open a data source without a DSN!

Normally on the `SCONNECT` command you specify the DSN you want to use. ODBC looks up this DSN and determines the driver to use and what connection it needs. In this version we let you specify the ODBC driver instead of an DSN. This is done as follows:

Assume you have a R:BASE database called RRBYW16 located in a folder called D:\SampleData. To access this database you need an R:BASE eXtreme 9.0 (64) or the Oterro 9.0 (64) ODBC driver installed. For the `SCONNECT` command use this:

Example 01 (Using R:BASE eXtreme 9.0 (32) ODBC Driver):

```
SCONNECT ";driver=R:BASE eXtreme 9.0 (32) Database Driver Database Driver (*.RB1);dbq=d:\SampleData\rrbyw16'
```

Example 02 (Using R:BASE eXtreme 9.0 (64) ODBC Driver):

```
SCONNECT ";driver=R:BASE eXtreme 9.0 (64) Database Driver Database Driver (*.RX1);dbq=d:\SampleData\rrbyw16'
```

Example 03 (Using Oterro 9.0 (64) ODBC Driver):

```
SCONNECT ";driver=Oterro 8.0 (64) Database Driver (*.RX1);dbq=d:\SampleData\rrbyw16'
```

Note the initial semi-colon. This is the flag that you are not specifying a DSN. The item after the 'driver' is the actual name of the driver as defined in ODBC. Make sure you spell and space this exactly as it is defined in ODBC Data Source Administrator Drivers. The database and the exact path of the database is specified in the 'dbq' part of the string. The entire string must be surrounded by your database [QUOTE](#) character. Single quotes are used in the example above.

Example 04 (To use an Access database in "dsnless" mode):

```
SCONNECT ";Driver={Microsoft Access Driver (*.mdb)};DBQ=c:\Documents and Settings\Administrator\My Documents\db1.mdb;'
```

This example uses a database called db1.mdb in the "My Documents" folder. The connection can specify additional items like user id with the "UID=" parameter and a password with the "PWD=" parameter.

Example 05 (To use a SQL Server database):

```
SCONNECT ";driver={SQL Server};server=corpseadb0d;uid=my_user_name;pwd=my_pw;database=JohnDoe;'
```

In this one note the server and database within the server get specified along with the uid and pwd.

Use [SATTACH](#) command to attach table(s).

With a connection like this you can use a linked R:BASE database without a pre-defined DSN. A table attached using such a connection remembers all this for later "DSN-less" access.

Notes:

To disconnect the DSN-less connection, use the exact same syntax using [SDISCONNECT](#) command.

1.19.4 SCREEN RESTORE

Use the SCREEN RESTORE command in forms when you want to redisplay the screen after returning from an entry/exit procedure (EEP). The SCREEN RESTORE command is only needed for R:BASE for DOS. R:BASE for Windows forms automatically refresh correctly.



Options

OFF

Set SCREEN RESTORE off to prevent the original form from displaying any changes after an entry/exit procedure executes in a form.

ON

Set SCREEN RESTORE on to redisplay the original form after an entry/exit procedure is executed in a form.

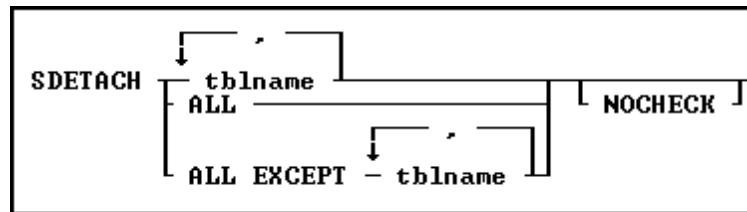
About the SCREEN RESTORE Command

The SCREEN RESTORE ON command, which is the default, redisplay the original screen. The SCREEN RESTORE OFF command does not redisplay the original screen, allowing for screen I/O during execution of the EEP to remain when control returns to the form.

In Windows, if one EEP uses the SCREEN RESTORE OFF command, all EEPs must also use it.

1.19.5 SDETACH

Use the SDETACH command to remove a foreign table from a connected R:BASE database.



Options

' Indicates that this part of the command is repeatable.

ALL

Specifies all tables.

ALL EXCEPT tblname

Specifies all tables except those specified.

NOCHECK

Eliminates the confirmation message.

tblname

Specifies the table to detach.

For information on how to attach foreign data source tables, see [SATTACH](#).

1.19.6 SDISCONNECT

Use the SDISCONNECT command to disconnect a foreign data source from an R:BASE database.

SDISCONNECT datasource

Options

datasource

Specifies the data source to disconnect.

About the SDISCONNECT Command

If you omit the *datasource* option, a dialog box opens listing data sources from which to choose.

Note:

To disconnect the DSN-less connection, use the exact same data source syntax as the [SCONNECT](#) command, only with SDISCONNECT.

Examples:

Example 01:

```
SDISCONNECT ';driver=R:BASE eXtreme 9.0 (32) Database Driver (*.RB1);dbq=d:\SampleData\rrbyw16'
```

Example 02:

```
SDISCONNECT ';driver=R:BASE eXtreme 9.0 (64) Database Driver (*.RX1);dbq=d:\SampleData\rrbyw16'
```

Example 03:

```
SDISCONNECT ';driver=Oterro 9.0 (64) Database Driver (*.RX1);dbq=d:\SampleData\rrbyw16'
```

Example 04:

```
SDISCONNECT ';Driver={Microsoft Access Driver (*.mdb)};DBQ=c:\Documents and Settings\Administrator\My Documents\db1.mdb;'
```

Example 05:

```
SDISCONNECT ';driver={SQLServer};server=corpseadb0d;uid=my_user_name;pwd=my_pw;database=JohnDoe;'
```

See also: [DSN-less connections](#)

1.19.7 SELECT

Use the SELECT command to display rows of data from a table or view. To display the data in the order you want, modify the SELECT command by using various clauses.

The SELECT command is a very powerful data retrieval command. By learning this command, and all of its parts you can greatly enhance your ability to work with any other R:BASE command that uses those same portions. For example, learning to use a WHERE clause with SELECT will help you work with WHERE clauses on other commands.

You can use the SELECT command to do the following:

- A SELECT command is essentially a process of elimination. A SELECT command can contain a number of clauses (two are required), each of which begins with a keyword, such as [FROM](#) or [WHERE](#).

SELECT... TOP... SELECT functions... INTO...
FROM... LIMIT... INNER JOIN... OUTER JOIN...
WHERE... Sub-SELECT... AS... GROUP BY...
HAVING... ORDER BY... UNION... HTML...

Note:

- ## SELECT Command Clause

- Select all columns by entering SELECT with an asterisk.
- Name the columns you want to select.
- Use expressions and SELECT functions to perform calculations whose results will also appear as a column in the final result.
- Select the column or expression values and load them into variables.

```

SELECT
  [ ALL | DISTINCT ]
  [ * | [ =S ] ]
  [ dbname. | tblview. | <expression> | USER ]
  [ colname | #c ]
  [ =w | =S ]
  ...
  INTO [ into_var ]
  [ INDICATOR ] ind_var
  ...
  FROM [ tblview ]
  [ corr_name ]
  [ WHERE clause ]
  [ ORDER BY clause ]

```

Options

Specifies all columns.

,

Indicates that this part of the command is repeatable.

ALL

Specifies all rows returned by the other clauses.

#c

Specifies a column, where #c is the column number shown in the output of the [LIST TABLES](#) command. You can enter a table or correlation name before the #c.

colname

Specifies a column name. In a command, you can enter #c, where #c is the column number shown when the columns are listed with the LIST TABLES command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*). You can enter *tblname.** to specify all columns in the table.

corr_name

Correlation name. A nickname or alias for a table or view name. Use *corr_name* to refer to the same table twice within the command, or to explicitly specify a column in more than one table.

dbname

Currently connected database name, plus the drive and directory if the database is not on the current directory. It has the form D:\PATHNAME/DBNAME where D: is the optional drive letter, /PATHNAME is the optional directory path, and /DBNAME is the database name.

DISTINCT

Eliminates duplicate rows from the resulting data set.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as #date, #time, and #pi.

FROM

Lists the tables from which data is to be displayed.

ind_var

Specifies a variable result indicator to be used with an [INTO](#) clause in a SELECT command. This variable stores the status of the variable: non-null (0) or null (-1).

INDICATOR

Indicates the following variable is an indicator variable, which is used to indicate if a null value is retrieved.

INTO

Selects information directly from a table and puts it into variables. You must include a WHERE clause so the SELECT command finds only one row.

into_var

Specifies a variable whose value is assigned with an INTO clause in a SELECT command.

ORDER BY clause

Sorts rows of data. See [ORDER BY](#).

=S

Calculates the sum of a column that has CURRENCY, DOUBLE, INTEGER, NUMERIC, or REAL data type values, or the results of an expression using CURRENCY, DOUBLE, INTEGER, NUMERIC, or REAL data type values.

tblview

Specifies a table or view name.

USER

Retrieves the current user as a constant.

=w

Specifies a display width.

WHERE clause

Limits rows of data. See "[WHERE](#)".

Examples

The following command selects the company name and ID for companies in Washington state:

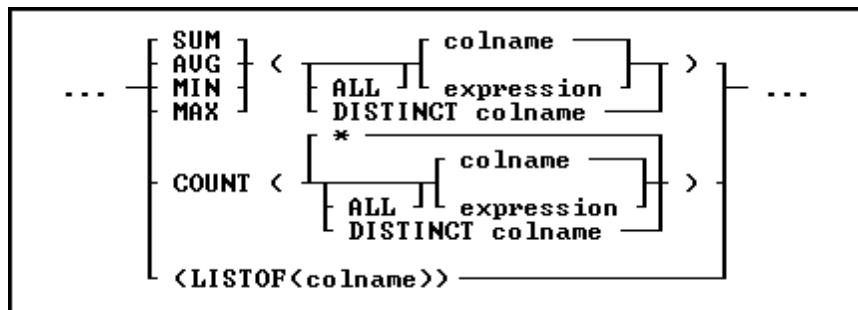
```
SELECT custid, company FROM customer +
WHERE custstate = 'WA' ORDER BY company
```

custid company

```
122    Data Solutions
119    Datacrafters Infosystems
130    MIS by Design
114    Softech Database Design
```

1.19.7.1 SELECT Functions

This clause, determines which columns to include.



Options

*

Specifies all rows.

AVG

Computes the numeric average of CURRENCY, DOUBLE, NUMERIC, REAL, INTEGER, DATE, or TIME data types. R:BASE rounds averages of integer values to the nearest integer value and currency values to their nearest unit.

COUNT

Determines how many non-null entries there are for a particular column item.

DISTINCT

Eliminates duplicate rows from the calculation.

LISTOF

Creates a text string of the values separated by the current comma delimiter character.

The LISTOF function can be used with the "SELECT ... INTO ..." to populate a variable with a list of values which can then be used in a [CHOOSE](#) command with the #LIST option. It can also be used in Forms, Reports or Labels to look up values from multiple rows.

MAX

Selects the maximum value in a column. This option accepts columns with any data type except BIT, VARBIT, VARCHAR, and BITNOTE.

MIN

Selects the minimum value in a column. This option accepts columns with any data type except BIT, VARBIT, VARCHAR, and BITNOTE.

SUM

Computes the numeric sum of CURRENCY, DOUBLE, REAL, NUMERIC, or INTEGER data types.

About SELECT Functions

A SELECT function can be used to provide summary data about a group of rows in a table or for all rows in a table. These functions may only be used with the GROUP BY clause or when only SELECT functions are specified.

Using MIN and MAX functions for NOTE data type

Selecting aggregate functions, such as MIN and MAX, requires that R:BASE keeps an accumulator and choose to only use the **first 80 characters** for NOTE values. This matches the fact that if you sort on NOTE fields, the sort will be based on the first 80 characters only.

Examples

The COUNT function works in two different ways, depending on its argument. COUNT(*) counts all rows in a table, but COUNT(*colname*) counts only rows in which the value in the specified column is not null. For example:

```
SELECT emptitle, COUNT(*), COUNT(emptitle) +
FROM employee GROUP BY emptitle
```

The final result contains both forms of the COUNT function.

emptitle	COUNT(*)	COUNT(emptitle)
Manager	2	2
Representative	3	3
Sales Clerk	1	1
-0-	2	0

If you wanted to compute the difference between each employee's average sales and the average sales for all employees, you would first use a SELECT function to calculate the average for all employees and store the result in a variable. For more information, see [INTO](#).

The following command assigns the value \$71,571.88 to the *vaverage* variable.

```
SELECT AVG(netamount) INTO vaverage FROM transmaster
```

Next, you can use the variable and another SELECT function in an expression to calculate the difference for each employee, and display the average net amount for each employee.

```
SELECT empid, AVG(netamount), +
    (.vaverage - (AVG(netamount)))=30 +
FROM transmaster GROUP BY empid
```

empid	AVG (netamount)	(.vaverage - AVG(netamount))
102	\$64,510.00	\$7,061.88
129	\$69,555.00	\$2,016.88
131	\$118,000.00	-\$46,428.12
133	\$44,380.00	\$27,191.88
160	\$114,850.00	-\$43,278.12

165	\$14,685.00	\$56,886.88
167	\$3,830.00	\$67,741.88

Examples Using the LISTOF Function

```
SELECT (LISTOF(ColumnName)) INTO VariableName INDICATOR IndVar +
FROM TableView WHERE ...
```

In a Form, Report or Label Expression:

```
VariableName = (LISTOF(ColumnName)) IN LookUpTableView WHERE +
KeyColumn = KeyColumn
```

Example 01:

```
SET VAR vValueList TEXT = NULL
SELECT (LISTOF(ColumnName)) INTO vValueList INDIC IvValueList +
FROM TableName WHERE ...
```

The variable vValueList will be a text string of the values separated by the current comma delimiter character.

If you would like to add a single space after each value, then:

```
SET VAR vValueList TEXT = NULL
SELECT (SRPL(LISTOF(ColumnName),', ',' ',0)) INTO +
vValueList INDIC IvValueList FROM TableName WHERE ...
```

Notice the additional space after comma in ReplaceString.

If you would like to use a carriage return after each value, then:

```
SET VAR vValueList TEXT = NULL
SELECT (SRPL(LISTOF(ColumnName),', ',(CHAR(10)),0)) INTO +
vValueList INDIC IvValueList FROM TableName WHERE ...
```

Example 02:

```
CONNECT Concomp IDENTIFIED BY NONE
SET CAPTION ' '
SET AUTODROP OFF
SET RBGSize CENTER CENTER 800 600
SET VAR vLines INTEGER = 0
SET VAR vValueList TEXT = NULL
SET VAR vLastName TEXT = NULL
SET VAR vTitle TEXT = 'List Created Using LISTOF Function'
SET VAR vCaption TEXT = 'Using #LIST Options in CHOOSE Command!'
CLS
PAUSE 3 USING 'Collecting Values ...' CAPTION .vCaption AT 16 30
SELECT (COUNT(*)), (LISTOF(EmpLName)) INTO +
vLines INDIC IvLines, vValueList INDIC IvValueList FROM Employee
IF vLines > 18 THEN
    SET VAR vLines = 18
ENDIF
CLS
CHOOSE vLastName FROM #LIST .vValueList AT 6 30 +
TITLE .vTitle CAPTION .vCaption LINES .vLines FORMATTED
IF vLastName IS NULL OR vLastName = '[Esc]' THEN
```

```

    GOTO Done
ELSE
    CLEAR ALL VAR EXCEPT vLastName
ENDIF
-- Do what you have to do here ...
LABEL Done
CLS
CLEAR ALL VAR
QUIT TO MainMenu.RMD
RETURN

```

Example 03:

If you would like to retrieve the list of DISTINCT values, then:

```

SET VAR vValueList TEXT = NULL
SELECT (LISTOF(DISTINCT ColumnName)) INTO +
vValueList INDIC IvValueList FROM TableName WHERE ...

```

1.19.7.2 TOP

This parameter includes the support to specify the TOP n qualifier for the [SELECT](#) command. The TOP n will retrieve the top number of records from the table.

```
SELECT TOP n * FROM tablename ...
```

The "TOP n" goes between the word SELECT and the column list for the command.

Example:

To show the top 5 bonuses where the bonus is under \$500 from the SalesBonus table in the ConComp:

```
SELECT TOP 5 * FROM SalesBonus WHERE Bonus < 500 ORDER BY Bonus=DESC
```

1.19.7.3 INNER JOIN

This clause is used to retrieve data from two tables.

```

SELECT ... FROM lefttblview [ corr_name ] ...
      ... INNER JOIN righttblview [ corr_name ] ...
      ... ON [ lefttblview ].column1 = [ righttblview ].column2 ...
      ... [ WHERE Clause ] ...

```

Options**.column1**

Defines the column on which to link.

.column2

Defines the column on which to link.

corr_name

A correlation name is an alias or nickname for a table. It lets you refer to the same table twice in one command, use a shorter name, and explicitly refer to a column when referring to the same column if that column appears in more than one table.

FROM lefttblview

Specifies the left table or view.

lefttblview

Explicitly defines the column on which to link the left table name or view.

INNER JOIN righttblview

Specifies the right table or view.

righttblview

Explicitly defines the column on which to link the right table name or view.

WHERE clause

Limits rows of data. See [WHERE](#).

About JOIN

When you perform a SQL JOIN, you specify one column from each table to join on. These two columns contain data that is shared across both tables. You can use multiple joins in the same SQL statement to query data from as many tables as you like.

JOIN Types

Depending on your requirements, you can do an "**INNER**" join or an "[OUTER](#)" join. The differences are:

- **INNER JOIN:** This will only return rows when there is at least one row in both tables that match the join condition.
- **LEFT OUTER JOIN:** This will return rows that have data in the left table (left of the JOIN keyword), even if there's no matching rows in the right table.
- **RIGHT OUTER JOIN:** This will return rows that have data in the right table (right of the JOIN keyword), even if there's no matching rows in the left table.
- **FULL OUTER JOIN:** This will return all rows, as long as there's matching data in one of the tables.

Example (INNER JOIN)

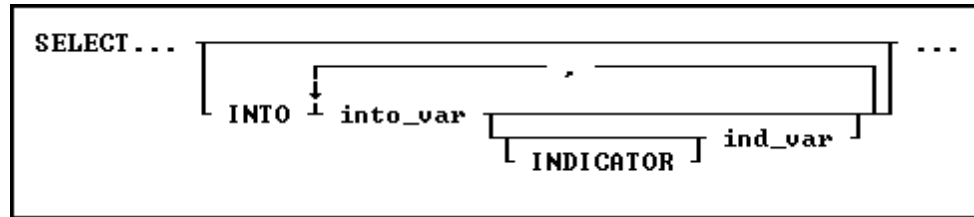
The following example lists all of the employees and their total sales, including those employees who have not yet completed a sale.

```
SELECT t1.empid, t2.netamount, t2.transdate FROM employee t1 +
INNER JOIN transmaster t2 ON t1.empid = t2.empid +
WHERE empid = 129
```

t1.empid	t2.netamount	t2.TransDate
129	\$3,080.00	07/02/2003
129	\$5,385.00	07/08/2003
129	\$6,160.00	07/11/2003
129	\$5,575.00	08/24/2003
129	\$10,445.00	08/24/2003
129	\$10,175.00	08/25/2003
129	\$2,195.00	08/27/2003

1.19.7.4 INTO

If the result consists of one row, this clause loads the data into one or more variables, one for each column value in the result.



Options

ind_var

Stores an INTEGER value (-1 or 0) that indicates whether the preceding *into_var* received a null value or a non-null value; this is an optional indicator variable. If you omit indicator variables, R:BASE displays a message and assigns a negative integer to *SQLCODE* if it encounters a null value. The command continues to process rows.

INDICATOR

Indicates the following variable is an indicator variable, which is used to indicate if a null value is retrieved.

into_var

Assigns the result associated with a column, expression, or function named in the command clause to the corresponding variable named in the INTO clause. The number of items or variables named in the command and INTO clauses, as well as their data types, must be the same.

About the SELECT INTO command

This optional clause loads the results of a SELECT command into variables, but does not display the results on screen.

An INTO clause loads the resulting value of each column, expression, or function included in the command clause into a variable. If previous clauses have returned more than one row, the values assigned to the variables are unpredictable. You should make sure you are returning only one row. Either test the results before using an INTO clause or check the value of the variable *sqlcode* after executing the command. If the clause is successful, *sqlcode* is 0.

Comments

The INTO clause must have a corresponding variable for every item in the command clause; values are assigned to variables in the order of items in the command clause. The data type of each command clause item and its corresponding *into_var* must be compatible. For example:

```

SELECT MAX(listprice), MIN(listprice) +
      INTO vmaxprice INDICATOR vind_max, +
          vminprice INDICATOR vind_min +
      FROM product

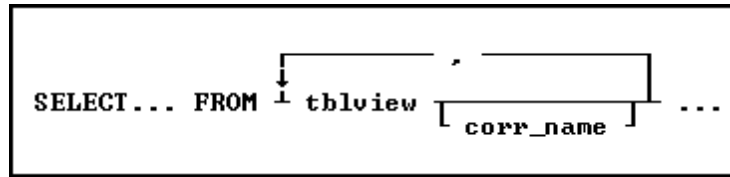
```

The MAX and MIN functions assign the value \$3,100.00 to the variable *vmaxprice* and \$1,900.00 to *vminprice*. These values are the maximum and minimum values for the *listprice* column in the *product* table. Since both functions returned values, the value of both indicator variables is 0. Also, since only SELECT functions are specified, a [GROUP BY](#) clause is not required.

If you select and load a value into an undefined numeric variable, that variable acquires the precision and scale of the column from which the value is selected.

1.19.7.5 FROM

Starting with all the tables, views, rows, and columns in the database, this clause specifies one or more tables or views from which you want data.



Options

' Indicates that this part of the command is repeatable.

corr_name

A correlation name is an alias or nickname for a table. It lets you refer to the same table twice in one command, use a shorter name, and explicitly refer to a column when referring to the same column if that column appears in more than one table.

tblview

A table or view containing one or more columns named in the command clause.

About the FROM Clause

The FROM clause names one or more tables and/or views from which the information is used in a SELECT command or other command. It is one of the two **REQUIRED** portions of a SELECT statement. The other required portion being the column listing. Some other commands that may use a FROM clause include [TALLY](#), [COMPUTE](#) and [CHOOSE](#).

Examples

The following command selects all columns from the *transmaster* table in the R:BASE sample database, *concomp*.

```
SELECT * FROM transmaster
```

The result of this command appears in the following table. The *transid* column is the primary key for this table; that is, *transid* contains a unique value for each row in the table. Columns that are not primary keys can have the same value in more than one row. The result shown here is used in the discussions of other SELECT clauses later in this section.

transid	custid	empid	transdate	netamount	freight
4760	100	133	01/02/94	\$32,400.00	\$324.00
4780	105	160	01/08/94	\$9,500.00	\$95.00
4790	104	129	01/09/94	\$6,400.00	\$64.00
4795	101	102	01/11/94	\$176,000.00	\$1,760.00
4800	105	160	02/22/94	\$194,750.00	\$1,947.50
4865	102	129	02/22/94	\$34,125.00	\$341.25
4970	103	131	02/23/94	\$152,250.00	\$1,522.50
4975	101	102	02/26/94	\$87,500.00	\$875.00
4980	101	102	02/27/94	\$22,500.00	\$225.00
5000	101	102	02/28/94	\$40,500.00	\$405.00
5010	107	131	03/02/94	\$108,750.00	\$1,087.50
5015	103	131	03/05/94	\$80,500.00	\$805.00
5050	104	129	03/06/94	\$56,250.00	\$562.50
5060	101	102	03/07/94	\$57,500.00	\$575.00
5065	106	160	03/13/94	\$140,300.00	\$1,403.00
5070	104	129	03/14/94	\$95,500.00	\$955.00
5075	102	129	03/15/94	\$155,500.00	\$1,555.00

5080	100	133	03/19/94	\$88,000.00	\$880.00
5085	107	131	03/18/94	\$130,500.00	\$1,305.00
5045	100	102	09/26/94	\$3,060.00	\$30.60
5046	101	165	09/27/94	\$3,060.00	\$30.60
5047	102	167	09/27/94	\$3,830.00	\$38.30
5048	103	133	-0-	\$12,740.00	\$127.40
5049	102	165	04/21/94	\$26,310.00	\$263.10

When a column appears in more than one table, enter the table name and a period preceding each column name to specify the column you want. For example:

```
SELECT transmaster.transid, transmaster.netamount, +
transdetail.model FROM transmaster, transdetail +
WHERE transmaster.transid = transdetail.transid
```

Or, you can assign a correlation name to a table. The following command is equivalent to the previous example:

```
SELECT t1.transid, t1.netamount, t2.model +
FROM transmaster t1, transdetail t2 +
WHERE t1.transid = t2.transid
```

In this SELECT command, the FROM clause assigns correlation names to the *transmaster* and *transdetail* tables. Because the *transid* column appears in both tables, the correlation names, *t1* and *t2*, clarify which table each column is from.

Because R:BASE processes the FROM clause first, you must use correlation names, if you have assigned them, throughout the SELECT command.

1.19.7.6 LIMIT

This parameter includes the support to LIMIT the [SELECT](#) results.

```
SELECT * FROM tablename LIMIT offset, row_count
```

The LIMIT clause can be used to constrain the number of rows returned by the SELECT statement. LIMIT takes one or two numeric arguments, which must be integer constants.

With two arguments, the first argument specifies the offset of the first row to return, and the second specifies the maximum number of rows to return. The offset of the initial row is 0 (not 1):

```
SELECT * FROM table LIMIT 5,10
Retrieves rows 6-15
```

To retrieve all rows from a certain offset up to the end of the result set, you can use some large number for the second parameter. This statement retrieves all rows from the 96th row to the last:

```
SELECT * FROM table LIMIT 95,99999999
```

With one argument, the value specifies the number of rows to return from the beginning of the result set:

```
SELECT * FROM table LIMIT 5
Retrieve first 5 rows
```

In other words, LIMIT n is equivalent to LIMIT 0,n.

Examples:

SELECT * FROM tablename LIMIT 0,30
Gets rows 1-30 from a table

SELECT * FROM tablename LIMIT 5,10
Gets rows 6-15 from a table

SELECT * FROM tablename LIMIT 10
Gets the first 10 rows from a table

1.19.7.7 OUTER JOIN

This clause is used to retrieve data from two tables.

```
SELECT ... FROM lefttblview [corr_name] [ LEFT
                                RIGHT
                                FULL ] ...

    ... OUTER JOIN righttblview [corr_name] ...

    ... ON [ lefttblview ] .column1 = [ righttblview ] .column2 ...

    ... [ WHERE clause ] ...
```

Options

.column1

Defines the column on which to link.

.column2

Defines the column on which to link.

corr_name

A correlation name is an alias or nickname for a table. It lets you refer to the same table twice in one command, use a shorter name, and explicitly refer to a column when referring to the same column if that column appears in more than one table.

FROM lefttblview

Specifies the left table or view.

LEFT RIGHT FULL

Specifies the type of outer join.

lefttblview

Explicitly defines the column on which to link the left table name or view.

OUTER JOIN righttblview

Specifies the right table or view.

righttblview

Explicitly defines the column on which to link the right table name or view.

WHERE clause

Limits rows of data. See [WHERE](#).

About JOIN

When you perform a SQL JOIN, you specify one column from each table to join on. These two columns contain data that is shared across both tables. You can use multiple joins in the same SQL statement to query data from as many tables as you like.

JOIN Types

Depending on your requirements, you can do an "[INNER](#)" join or an "**OUTER**" join. The differences are:

- **INNER JOIN:** This will only return rows when there is at least one row in both tables that match the join condition.
- **LEFT OUTER JOIN:** This will return rows that have data in the left table (left of the JOIN keyword), even if there's no matching rows in the right table.
- **RIGHT OUTER JOIN:** This will return rows that have data in the right table (right of the JOIN keyword), even if there's no matching rows in the left table.
- **FULL OUTER JOIN:** This will return all rows, as long as there's matching data in one of the tables.

About OUTER JOIN

When you use an outer join, rows are not required to have matching values. The table order in the FROM clause specifies the left and right table. You can include a WHERE clause and other SELECT clause options such as [GROUP BY](#). The result set is built from the following criteria:

- In all types of outer joins, if the same values for the linking columns are found in each table, R:BASE joins the two rows.
- For a left outer join, R:BASE uses each value unique to the left (first) table and completes it with nulls for the columns of the right (second) table when the linking columns do not match.
- A right outer join uses unique values found in the right (second) table and completes the rows with nulls for columns of the left (first) table when the linking columns do not match.
- A full outer join first joins the linking values, followed by a left and right outer join.

Example

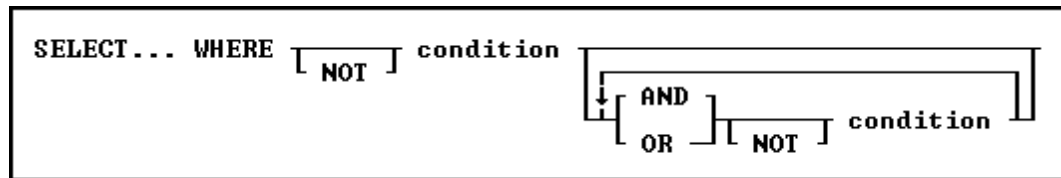
The following example lists all of the employees and their total sales, including those employees who have not yet completed a sale.

```
SELECT t1.empid, SUM(t2.netamount) FROM employee t1 +
FULL OUTER JOIN transmaster t2 ON t1.empid = t2.empid +
GROUP BY t1.empid
```

t1.empid	SUM(t2.netamount)
102	\$387,060.00
129	\$347,775.00
131	\$472,000.00
133	\$133,140.00
160	\$344,550.00
165	\$29,370.00
166	\$0.00
167	\$3,830.00

1.19.7.8 WHERE

This clause determines which rows of data to include.



Options

AND

Indicates the following condition must be met along with the preceding condition.

condition

Identifies requirements to be in the WHERE syntax.

NOT

Reverses the meaning of a connecting operator. AND NOT, for example, indicates that the first condition must be met and the following condition must not be met.

OR

Indicates the following condition can be met instead of the preceding condition.

About the WHERE Clause

In most commands, a WHERE clause follows the syntax diagram above. The [JOIN](#) command does not use that syntax diagram because it uses only comparison operators in a WHERE clause to compare two columns.

The two main elements in any WHERE clause are conditions and connecting operators.

We now support "COUNT = LAST" in two different ways. If the entire WHERE clause is "WHERE COUNT = LAST" then R:BASE works like it always has to quickly fetch the last row of the table. The NEW functionality is to have other conditions in the WHERE clause and you want the last row of whatever qualifies.

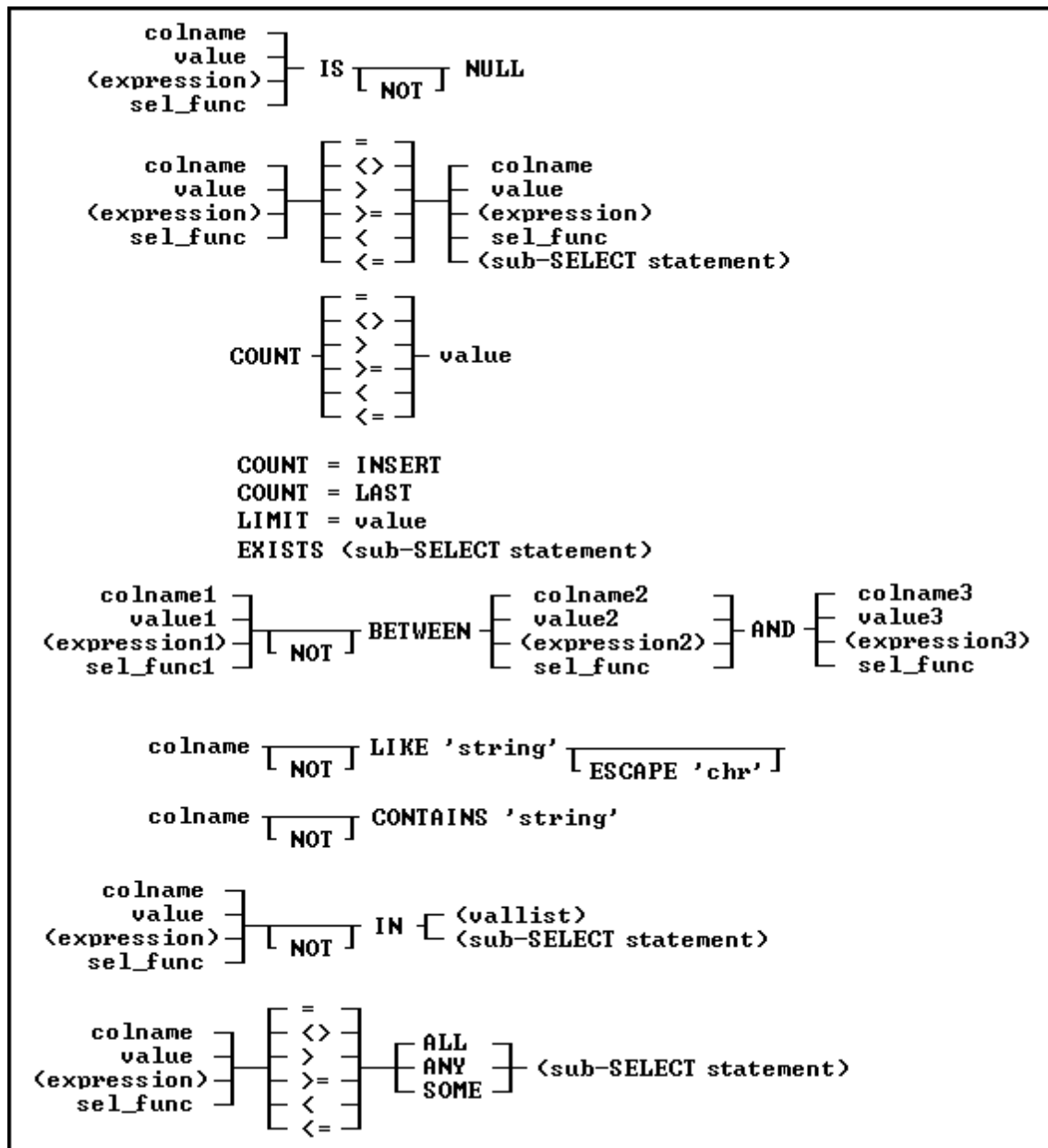
To make it work this way specify the other conditions and then add "AND COUNT = LAST".

Here is an example:

```
SELECT * FROM Customer WHERE CustID > 100 AND COUNT = LAST
```

WHERE Clause Conditions

The following syntax diagram and table show the basic formats for WHERE clause conditions, which can be used alone or together.



Basic WHERE Clause Conditions

Condition Syntax	Description
<code>colname op DEFAULT</code>	True if a column value compares correctly with the DEFAULT value for the column. <i>Op</i> can be <code>=</code> , <code><></code> , <code>>=</code> , <code>></code> , <code><=</code> , or <code><</code> .
<code>colname = USER</code>	True if a column value equals the current user identifier.
<code>item1 IS NULL</code>	True if <i>item1</i> has a null value. <i>Item1</i> can be a column name, value, or expression. A null value cannot be used in a comparison with an operator.
<code>item1 op item2</code>	True if the relationship between two items is true as defined by an operator. <i>Item1</i> can be a column name, value, or expression; <i>item2</i> can be a column name, value, expression, or sub-SELECT statement.
<code>COUNT=INSERT</code>	Refers to the last row inserted by the current user, even if it has been modified by another user.
<code>COUNT=LAST</code>	Refers to the last row in a table. COUNT cannot be used with a

	view.
COUNT <i>op value</i>	Refers to a number of rows defined by <i>op</i> and <i>value</i> .
LIMIT= <i>value</i>	Specifies a number of rows affected by a command. A LIMIT condition should be the last condition in a WHERE clause.
EXISTS (sub-SELECT statement)	True if sub-SELECT statement returns one or more rows.
<i>item1</i> BETWEEN <i>item2</i> AND <i>item3</i>	True if the value of <i>item1</i> is greater than or equal to the value of <i>item2</i> , and if the value of <i>item1</i> is less than or equal to the value of <i>item3</i> .
<i>colname</i> LIKE ' <i>string</i> '	True if a column value equals the text string. With LIKE, a string can also be a DATE, TIME, or DATETIME value. The text string can contain R:BASE wildcard characters.
<i>colname</i> LIKE ' <i>string</i> ' ESCAPE ' <i>chr</i> '	True if a column value equals a text string. If you want to use a wildcard character as a text character in the string, specify the ESCAPE character <i>1chr</i> . In the string, use <i>chr</i> in front of the wildcard character.
<i>colname</i> CONTAINS ' <i>string</i> '	True if a column value contains the text string.
<i>colname</i> SOUNDS ' <i>string</i> '	True if the soundex value of a column matches the soundex value of the text string.
<i>item1</i> IN (<i>vallist</i>)	True if <i>item1</i> is in the value list.
<i>item1</i> IN (sub-SELECT statement)	True if <i>item1</i> is in the rows selected by a sub-SELECT.
<i>item1 op</i> ALL (sub-SELECT statement)	True if the relationship between <i>item1</i> and every row returned by a sub-SELECT statement matches an operator.
<i>item1 op</i> ANY(sub-SELECT statement)	True if the relationship between <i>item1</i> and at least one value returned by a sub-SELECT statement matches an operator.
<i>item1 op</i> SOME (sub-SELECT statement)	ANY and SOME are equivalent.

Notes:

- Placing NOT before most text operators (such as NULL or BETWEEN) reverses their meaning.
- When a SELECT statement is part of a WHERE clause, it is called a [sub-SELECT](#) clause. A sub-SELECT clause can contain only one column name (not a column list or *), expression, or function. The [INTO](#) and [ORDER BY](#) clauses in a sub-SELECT are ignored.

You can only use the current wildcard characters to compare a column to a text value when using the LIKE comparison. The default wildcard characters are the percent sign (%), which is used for one or more characters, and the underscore (_), which is used for a single character.

If you compare a column with a value, you can either enter the value or specify a global variable. If you specify a variable, R:BASE compares the column with the current value of the variable.

To significantly reduce processing time for a WHERE clause, use INDEX processing. To use indexes, the following conditions must be met:

- A condition in the WHERE clause compares an indexed column.
- If the WHERE clause contains more than one condition, R:BASE selects the condition that places the greatest restriction on the WHERE clause.
- Conditions are not joined by the OR operator.
- The comparison value is not an expression.

Connecting Operators

When you use more than one condition in a WHERE clause, the conditions are connected using the connecting operators AND, OR, AND NOT, and OR NOT.

The connecting operator AND requires that both conditions it separates must be satisfied. The connecting operator OR requires that either condition it separates must be satisfied.

The connecting operator AND NOT requires that the preceding condition must be satisfied, and the following condition must not be satisfied. The connecting operator OR NOT requires that either the preceding condition must be satisfied, or any condition except the following condition must be satisfied.

In WHERE clauses with multiple conditions, conditions that are connected by AND or AND NOT are evaluated before those connected by OR or OR NOT. However, you can control the order in which

conditions are evaluated by either placing parentheses around conditions or using the [SET AND](#) command. If you set AND off, conditions are always evaluated from left to right.

Examples

The following WHERE clause chooses sales amounts that are less than the value of a variable containing the daily average.

```
... WHERE amount < .dailyave
```

The following WHERE clause specifies the seventh row.

```
... WHERE COUNT = 7
```

The following WHERE clause specifies each row from the *employee* table that contains both the first name *June* and the last name *Wilson*.

```
SELECT * FROM employee WHERE empfname = 'june' AND emplname = 'wilson'
```

The following WHERE clause selects dates in the *actdate* column that are greater than dates in the *begdate* column or are less than dates in the *enddate* column.

```
... WHERE actdate BETWEEN begdate AND enddate
```

The next three WHERE clauses use the following data:

```
empfname emplname
-----
    Mary Jones
    John Smith
    Agnes Smith
    John Brown
```

In both of the following clauses, R:BASE first evaluates the conditions connected by AND, selecting John Smith. Then R:BASE adds any Marys to the list because the connecting operator is OR. The final result includes John Smith and Mary Jones.

```
...WHERE empfname = 'Mary' OR empfname = 'John' +
AND emplname = 'Smith'
```

```
...WHERE empfname = 'Mary' OR (empfname = 'John' +
AND emplname = 'Smith')
```

By moving the parentheses around the conditions connected by OR, you can select only John Smith. In the following WHERE clause, the first name can be either Mary or John, but the last name must be Smith.

```
...WHERE (empfname = 'Mary' OR empfname = 'John') AND +
emplname = 'Smith'
```

The following example illustrates a sub-SELECT in a WHERE clause. Assume you wanted a list of all sales representatives that had transactions greater than \$100,000, and the information for such a list was contained in two tables, *employee* and *transmaster*. The relevant columns in these tables are:

```
employee transmaster
empid  emplname empid  netamount
-----
    102 Wilson      133   $32,400.00
    129 Hernandez   160    $9,500.00
    133 Coffin      129    $6,400.00
    165 Williams    102  $176,000.00
```

166 Chou	160	\$194,750.00
167 Watson	129	\$34,125.00
160 Smith	131	\$152,250.00
131 Simpson	102	\$87,500.00
102		\$22,500.00
102		\$40,500.00
131		\$108,750.00
131		\$80,500.00
129		\$56,250.00
102		\$57,500.00
160		\$140,300.00
129		\$95,500.00
129		\$155,500.00
133		\$88,000.00
131		\$130,500.00
102		\$3,060.00
165		\$3,060.00
167		\$3,830.00
133		\$12,740.00
165		\$26,310.00

To display a list of employees in the *transmaster* table with a transaction larger than \$100,000, enter the following command:

```
SELECT empid, emplname FROM employee WHERE empid IN +
  (SELECT empid FROM transmaster WHERE netamount > 100000)
```

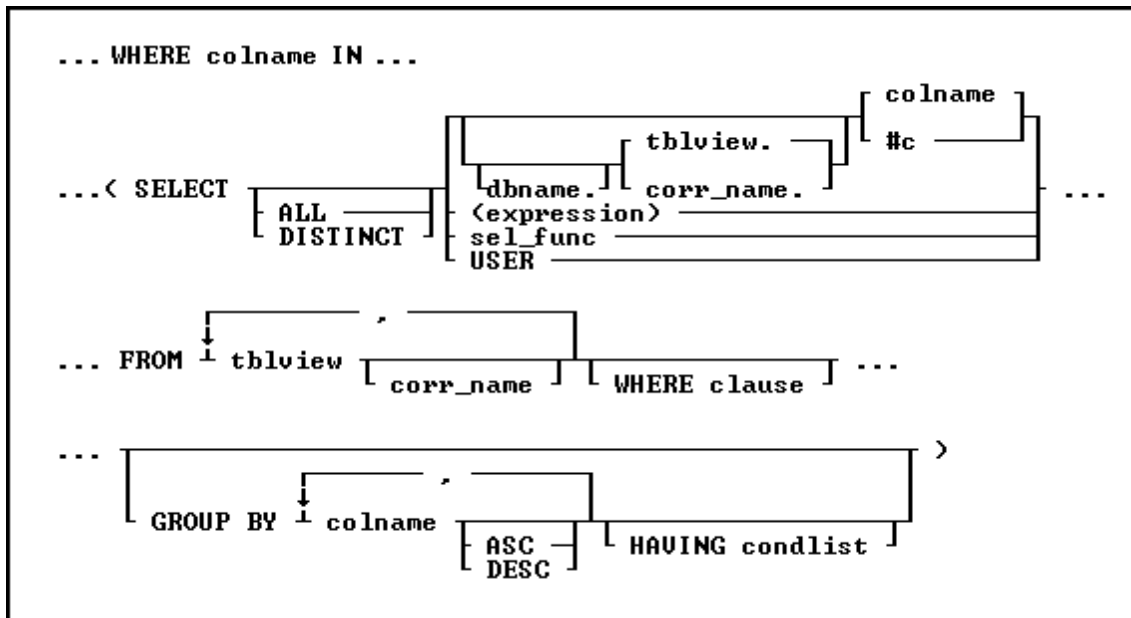
R:BASE displays the following list:

```
    empid emplname
-----
    102 Wilson
    129 Hernandez
    131 Simpson
    160 Smith
```

Note: You can use a [sub-SELECT](#) in any command that allows a full WHERE clause.

1.19.7.9 Sub-SELECT

This clause works in conjunction with the [WHERE](#) clause to determine which rows of data to include.



Options

For a description of the options, see [SELECT](#).

About the Sub-SELECT Clause

A sub-SELECT command, which is a SELECT command nested within another command, always appears in a [WHERE](#) clause, whether the sub-SELECT command is nested in the WHERE clause of a SELECT command or in another command such as [EDIT USING](#) form. R:BASE processes the clauses in a sub-SELECT in the same order as in a SELECT command.

Example

The following example selects customers that have purchased items in the month of January.

```

SELECT company FROM customer WHERE custid +
  IN (SELECT custid FROM transmaster +
    WHERE (IMON(transdate)=1))

```

company

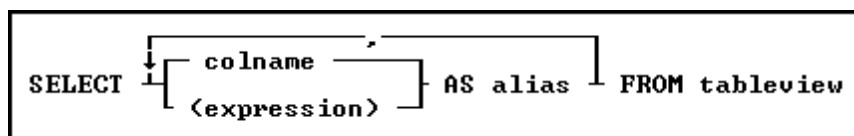
```

PC Distribution Inc.
Computer Distributors Inc.
Industrial Concepts Inc.
PC Consultation and Design

```

1.19.7.10 AS

This clause dynamically renames columns in a [SELECT](#) clause.



Options

'
Indicates that this part of the command is repeatable.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

AS alias

Determines the alias of the column. This may be used to refer to the column in other locations.

FROM tableview

Specifies the table or view to draw information from.

About the SELECT AS command

The SELECT AS command functions exactly like any other SELECT command and will accept all other SELECT options. The one noticeable exception is that this allows you to give a column an alias. This can be most useful when used in conjunction with the [CREATE VIEW](#) command.

Examples

The following 3 examples are based on the *ConComp* database.

The following command selects the EmpID and the EmpName columns from the Employee table and renames them to "EmployeeID" and "Name".

```
SELECT EmpID AS EmployeeID, EmpName AS Name FROM Employee
```

The following command creates a VIEW using the SELECT AS notation. This view contains a column for Employee ID, Employee Name (which is a single column based on the EmpFName and EmplName columns) from the Employee table and the Transdate and NetAmount columns from the Transmaster table. The immediately following command browses the Employee Name, Transdate and Netamount column. For more on Views please see [CREATE VIEW](#).

```
CREATE VIEW EmpAmount AS SELECT T1.EmpID, +
    (T1.EmpFName + ' ' + T1.EmplName) AS EmpName, +
    T2.TransDate, T2.NetAmount FROM Employee T1, +
    TransMaster T2 WHERE T1.EmpID = T2.EmpID
```

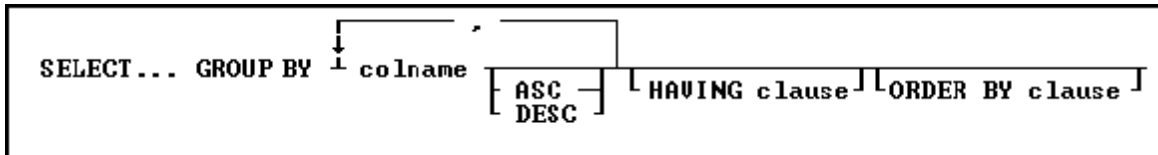
```
BROWSE EmpName, TransDate, NetAmount FROM EmpAmount
```

This final example uses IDQuotes to create a column name with spaces in it and then uses the [SELECT HTML](#) option to turn that into an HTML table with the column name. The [OUTPUT](#) commands redirect output to a file called "Emp.HTM" and then back to the screen. We do NOT recommend using this method to create VIEWS or TABLES with names that contain spaces as this could lead to Database Corruption.

```
OUTPUT EMP.HTM
SELECT EmpID as `Employee ID` FROM Employee HTML
OUTPUT SCREEN
```

1.19.7.11 GROUP BY

This clause determines which rows of data to include.



Options

' Indicates that this part of the command is repeatable.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter #*c*, where #*c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*).

GROUP BY

Returns a groups of rows as a summary resulting in only unique rows. This option is generally used with [SELECT Functions](#).

HAVING clause

Limits the rows affected by the GROUP BY clause.

ORDER BY clause

Sorts rows of data.

About the GROUP BY command

This optional clause groups rows according to the values in one or more columns and sorts the results. GROUP BY consolidates the information from several rows into one row. This results in a table with one row for each value in the named column or columns and one or more values per column.

The columns listed in the GROUP BY clause are related to those listed in the command clause. Any column named in the GROUP BY clause can also be named in the command clause, but any column not named in the GROUP BY clause can be used only in the command clause if the column is used in a SELECT function.

Examples

The [SELECT](#) command clause can contain the columns named in the GROUP BY clause, and SELECT functions that refer only to columns not named in the GROUP BY clause. Because the GROUP BY clause processes information resulting from a [WHERE](#) clause, you can add a GROUP BY clause to see the sales each employee has made:

```
SELECT empid FROM transmaster WHERE netamount < $100,000 + GROUP BY empid
```

The following intermediate result table contains columns not named in the command clause because the command clause has not been processed yet (not all the columns fit in the display, however). The first part of the processing is to group the rows by *empid*. Because seven different employees are included, the intermediate result table includes seven rows.

Intermediate Result Table-GROUP BY empid

<u>transid</u>	<u>custid</u>	<u>empid</u>	<u>netamount</u>
4975, 4980, 5000, 5060, 5045	101, 101, 101, 101, 100	102	\$87,500, \$22,500, \$40,500, \$57,500, \$3,060
4790, 4865, 5050, 5070	104, 102, 104, 104	129	\$6,400, \$34,125, \$56,250, \$95,500
5015	103	131	\$80,500
4760, 5080, 5048	100, 100, 103	133	\$32,400, \$88,000, \$12,740
4780	105	160	\$9,500
5046, 5049	101, 102	165	\$3,060, \$26,310
5047	102	167	\$3,830

You can include more than one column in a GROUP BY clause. If you group the rows in the above example by *custid* as well as *empid*, the command looks like this:

```
SELECT empid, custid FROM transmaster +
WHERE netamount < $100,000 GROUP BY empid, custid
```

In the following table, rows are now grouped by both *empid* and *custid*, resulting in eleven groups.

Intermediate Result Table-GROUP BY empid and custid

<u>transid</u>	<u>custid</u>	<u>empid</u>	<u>netamount</u>
5045	100	102	\$3,060
4975, 4980, 5000, 5060	101	102	\$87,500, \$22,500, \$40,500, \$57,500
4865	102	129	\$34,125
4790, 5050, 5070	104	129	\$64,000, \$56,250, \$95,500
5015	103	131	\$80,500
4760, 5080	100	133	\$32,400, \$88,000
5048	103	133	\$12,740
4780	105	160	\$9,500
5046	101	165	\$3,060
5049	102	165	\$26,310
5047	102	167	\$3,830

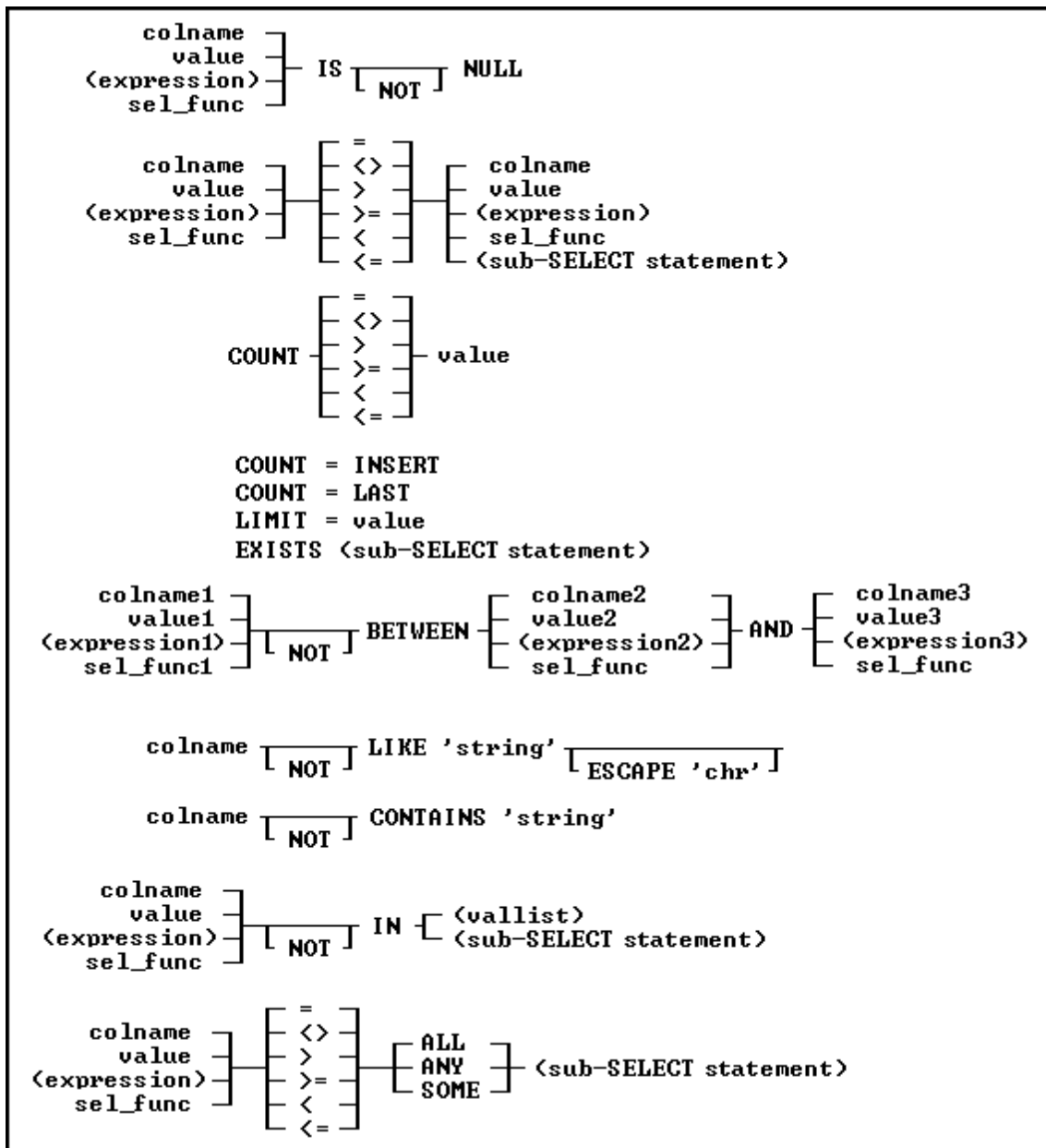
If one or more of the columns named in the GROUP BY clause contain null values, R:BASE forms a separate group for null values. Review the result of this [SELECT](#) command for the *employee* table:

```
SELECT empid, emplname, hiredate, emptitle FROM employee
```

<u>empid</u>	<u>emplname</u>	<u>hiredate</u>	<u>emptitle</u>
102	Wilson	03/18/90	Manager
129	Hernandez	08/28/91	Manager
131	Smith	04/14/92	-0-
133	Coffin	11/26/93	Representative
160	Simpson	01/09/94	-0-
165	Williams	07/05/92	Representative
167	Watson	07/10/92	Representative
166	Chou	07/10/93	Sales Clerk

If you group these rows by the *emptitle* column, which contains null values, you get the following intermediate result table:

Intermediate Result Table-GROUP BY emptitle



Examples

To display sales information for only those employees who have made more than one sale to the same customer, add a HAVING clause such as the following to one of the examples shown in [GROUP BY](#). When used in a HAVING clause, [SELECT functions](#) compute results based on the values grouped in the specified column. In this HAVING clause, COUNT returns the number of values grouped in the *transid* column.

```
SELECT empid, custid FROM transmaster +
WHERE netamount < $100,000 +
GROUP BY empid, custid HAVING COUNT(transid) > 1
```

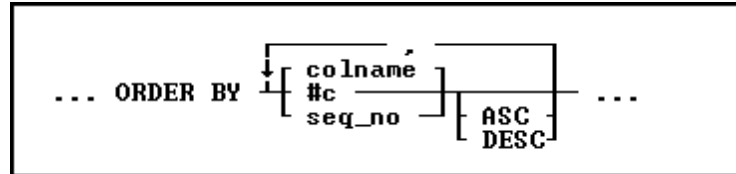
Intermediate Result Table-HAVING COUNT(transid) > 1

<u>transid</u>	<u>custid</u>	<u>empid</u>	<u>netamount</u>

4975, 4980, 5000, 5060	101	102	\$87,500, \$22,500, \$40,500, \$57,500
4790, 5050, 5070	104	129	\$6,400, \$56,250, \$95,500
5080	100	133	\$32,400, \$88,000

1.19.7.13 ORDERBY

This clause specifies how to sort the result of the [SELECT](#) command.



Options

' Indicates that this part of the command is repeatable.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

#c

Takes the place of a column name and refers to the column numbers displayed with the [LIST TABLE](#) command.

colname

Sorts by any column name or combination of column names.

seq_no

Refers to the items listed in the [SELECT](#) command that is using the ORDER BY command, ordered from left to right. An item can be a column name, expression, or [SELECT function](#).

About the ORDER BY Command

The syntax for the ORDER BY clause is the same for all commands. ORDER BY must refer to only one table or view.

You can significantly reduce the time R:BASE takes to process an ORDER BY clause when the column or columns listed in the ORDER BY clause are included in an index with the same column sort order as that specified in the ORDER BY clause.

Using the SET SORT Command

The ORDER BY command uses the R:BASE automatic sort optimizer. If you are sorting extremely large tables, and if your disk space is limited, the automatic sort optimizer might be unable to sort the data. Instead, use the [SET SORT ON](#) condition because it uses the least disk space necessary to sort data; however, the SET SORT ON condition is slower than the automatic sort.

Examples

The following command displays data from the *custid*, *company*, and *custcity* columns from the *customer* table.

```
SELECT custid, company, custcity FROM customer
```

The ORDER BY clause in the command below arranges the *custid* values in descending order.

```
SELECT custid, company, custcity FROM customer +
ORDER BY custid DESC
```

You can substitute a column's sequence number for a column named in the ORDER BY clause. You must use a sequence number when referring to an expression, function, constant, or when a UNION operator is used. The following command is equivalent to the command example above.

```
SELECT custid, company, custcity FROM customer ORDER +
BY 1 DESC
```

1.19.7.14 UNION

You can use this operator to combine the results of two or more [SELECT](#) statements.

```
SELECT clause FROM clause... UNION [ ALL ] SELECT clause FROM clause
```

About the UNION SELECT command

This optional operator combines the results of two SELECT commands or clauses, displaying the results of the second SELECT command below those of the first. By default, UNION deletes duplicate rows. Include the optional keyword ALL to include duplicate rows in the final result. You cannot combine [sub-SELECT](#) commands using UNION.

The UNION operator requires the following three conditions:

- The SELECT statements must specify an equal number of columns.
- Columns that are being combined must have the same data type.
- Only the last SELECT statement can contain an ORDER BY clause.

Examples

The following example lists all employees and the sales transactions for each, including those employees who have not yet completed a sale.

```
SELECT employee.empid, transid +
FROM employee, transmaster +
WHERE employee.empid = transmaster.empid +
UNION SELECT empid, 0 +
FROM employee +
WHERE employee.empid NOT IN +
(SELECT empid FROM transmaster)
```

The first SELECT displays the *empid* column from the *employee* table and *transid* from the *transmaster* table, linking the tables by the common column, *empid*. In short, the first SELECT displays all employees who have made a sale.

The second SELECT command selects the *empid* column from *employee*, including rows only for those employees who are not listed in the *transmaster* table. Because the results of the second SELECT are appended to those of the first (by the UNION operator), those employees who have not yet made a sale are shown at the bottom of the results with a zero in the *transid* column. The final results look like this:

empid	transid
102	4795
102	4975
102	4980
102	5000
102	5045
102	5060
129	4790
129	4865

```

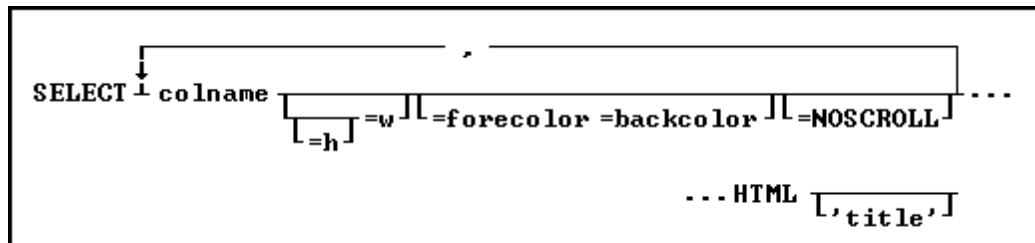
129      5050
129      5070
129      5075
131      4970
131      5010
131      5015
131      5085
133      4760
133      5048
133      5080
160      4780
160      4800
160      5065
165      5046
165      5049
166      0

```

1.19.7.15 HTML

This clause sends the [SELECT](#) command output into HTML format.

The rows and columns that each clause selects produce an intermediate result table that exists only in temporary memory. One after another, the clauses restrict the rows or columns included in the intermediate result table. After R:BASE has processed all the clauses and any [UNION](#) operators, the intermediate result table becomes the final result table.



Options

=backcolor

Is currently not supported; it must be set to: =default.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

=forecolor

Specifies the text color for the data in the column.

=h =w

Specifies height and width parameters. For both, the keywords LEFT, RIGHT, and CENTER can be used. For width, the range is from 0 - 255 pixels. 0 is no width specification. The height is the addition of the following values, one value from each item:

data justification:	0=default, 1=left, 2=center, 3=right
heading justification:	0=default, 4=left, 8=center, 12=right
vertical alignment:	0=default, 16=top, 32=middle, 48=bottom
HTML format flag:	0=no, 64=yes

HTML

Converts the data to HTML code.

'title'

This is the text that appears in the caption at the top of the Web Browser window. Adding *'title'* creates the beginning and ending table tag as well as putting the text you enter in the caption. Without *'title'*, a partial HTML file with the selected data in table row format is generated.

About the SELECT...HTML Command

The SELECT...HTML command is a modification of the [SELECT](#) command to output data in HTML format. If a title is specified, a full HTML file will be generated. Otherwise, a partial HTML file with the selected data in table row format will be generated.

1.19.8 SET

Use SET to change the current status of R:BASE special characters and operating conditions for any SET category. The SET operating conditions are used to set the database environment while you build and run commands from the R> Prompt or command files and applications.

About the SET Command

To change the setting from the R> Prompt for a special character, enter:

```
SET character_name = value
```

Do not use spaces on either side of the equal sign (=). For example:

```
SET QUOTES='
```

To change the setting from the R> Prompt for an operating condition, enter:

```
SET keyword setting
```

For example:

```
SET BELL ON
```

To remap keys on the keyboard from the R> Prompt, enter:

```
SET KEYMAP keyname = remapped keys
```

For example, if you want to remap [SHIFT] [F6] to [F2], enter:

```
SET KEYMAP [SHIFT] [F6] = [F2]
```

To reset a key to its original default value from the R> Prompt, enter:

```
SET KEYMAP keyname OFF
```

Saving Settings

Settings can be changed at the R> Prompt for the current session. They will revert to the default upon exiting R:BASE. Users can save settings in the R:BASE configuration file (RBENGINE9.CFG). Some settings are stored within the database itself and only the database owner can save changes to the settings that are stored in the database. If you are not the database owner, you can only change the settings that are stored in the database for the current database session.

The R:BASE/Oterro database provides the following categories of SET Keywords:

- Data Integrity
- Display Control
- Environment
- Format
- Programming

- Special Characters
- Transaction Processing and Multi-User
- Database Specific

1.19.8.1 AND

Operating Condition

Syntax: SET AND ON/OFF

Default: ON

SET AND gives the connecting operator AND precedence over OR in [WHERE](#), [IF](#), and [WHILE](#) conditions. When on, R:BASE processes conditions in the following order: all AND NOT followed by all AND OR, and all AND before all OR conditions.

SET AND OFF directs R:BASE to process conditions from left to right. The WHERE clause A AND B OR C AND D is evaluated as (((A AND B) OR C) AND D). SET AND ON directs R:BASE to give the operator AND precedence over OR. The WHERE clause A AND B OR C AND D is evaluated as ((A AND B) OR (C AND D)), which is a different result set.

1.19.8.2 ANSI

Operating Condition

Syntax: SET ANSI ON/OFF

Default: ON

Set ANSI off to restrict R:BASE to the list of reserved words shown in the table below. To change the default, save the setting to the RBENGINE9.CFG file.

R:BASE reserved words with ANSI set off:

ABS	ACOS	AINT	ALL
AND	ANINT	ASIN	ATAN
ATAN2	AVERAGE	CHAR	CHKKEY
COS	COSH	COUNT	CTR
CTXT	CURRENT	CVAL	DATE
DATETIME	DEXTRACT	DIM	ENVVAL
EXP	FLOAT	FOR	FORMAT
FROM	FULL	FV1	FV2
GETKEY	HELP	ICAP1	ICAP2
ICHAR	IDAY	IDWK	IFEQ
IFGT	IFLT	IFRC	IHASH
IHR	IMIN	IMON	IN
INT	ISEC	ISTAT	IYR
JDATE	LAST	LASTKEY	LAVG
LIMIT	LIST	LJS	LMAX
LMIN	LOG	LOG10	LUC
MAXIMUM	MINIMUM	MOD	NEW
NEWPAGE	NEXT	NINT	NOT
OR	OUTER	PASSTAB	PMT1
PMT2	PROMPT	PROMPTS	PV1
PV2	RATE1	RATE2	RATE3
RDATE	RJS	RTIME	SFIL
SGET	SIGN	SIN	SINH
SLEN	SLOC	SMOVE	SPUT
SQRT	SRPL	SSUB	STDEV
STRIM	SUM	TAN	TANH
TDWK	TERM1	TERM2	TERM3
TEXT	TEXTRACT	TMON	UDF

ULC	UNNAMED	VARIABLE	VARIANCE
WHERE			

1.19.8.3 AUTOCOMMIT

Operating Condition

Syntax: SET AUTOCOMMIT ON/OFF

Default: OFF

Mode: Transaction Processing

SET AUTOCOMMIT toggles AUTOCOMMIT processing on and off. When transaction processing and AUTOCOMMIT are on, each command that is executed successfully is immediately made permanent and visible to network users. If transaction processing is on and AUTOCOMMIT is off, you must enter a [COMMIT](#) command to make changes permanent. Also, leaving the database causes R:BASE to issue the COMMIT command.

When transaction processing is on and AUTOCOMMIT is off, you can enter a series of commands (a transaction) that change data or the database structure, then enter either a COMMIT or [ROLLBACK](#) command. COMMIT makes permanent all changes executed by commands in the transaction. ROLLBACK deletes all the changes, restoring the database to its state before the transaction began.

If you have started a transaction when you set AUTOCOMMIT on, R:BASE commits the transaction and turns AUTOCOMMIT on. You cannot open a cursor while AUTOCOMMIT is set on, and you cannot set AUTOCOMMIT on while a cursor is open.

AUTOCOMMIT can affect system performance. You can increase performance by setting AUTOCOMMIT to on when you do not need to enter commands in groups.

1.19.8.4 AUTOCONVERT

Operating Condition

Syntax: SET AUTOCONVERT ON/OFF

Default: OFF

Set AUTOCONVERT on to automatically convert R:BASE databases created in versions prior to 6.0 to R:BASE eXtreme. The user is not given the option to halt the conversion process.

1.19.8.5 AUTODROP

Operating Condition

Syntax: SET AUTODROP ON/OFF

Default: OFF

Controls the feature for a combo-box in a form to automatically drop-down the list when it gets focus.

When AUTODROP is SET to ON, this new setting controls the feature for a combo-box in a form to automatically drop-down the list when it gets focus. The editable or non-editable automatic drop-down list of combo box will allow the user to place the value into a column or variable.

The default for AUTODROP is to be OFF which makes previous versions of R:BASE 2000 (ver 6.5++) for Windows behave in the standard way.

This new setting can be saved in your R:BASE eXtreme for Windows RBENGINE9.CFG file or in your individual application startup file.

Supported environments:

- RBENGINE9.CFG

- Command File
- EEPs

Notes:

- [SHOW](#) AUTODROP will display the current setting of AUTODROP.
- (CVAL('AUTODROP')) will return the current setting of AUTODROP

1.19.8.6 AUTORECOVER

Operating Condition

Syntax: SET AUTORECOVER ON/OFF

Default: OFF

Mode: Transaction Processing

If AUTORECOVER is set on, errors that can occur during transaction processing when the program is interrupted, for example from a network or power failure, are automatically corrected.

1.19.8.7 AUTOROWVER

Operating Condition

Syntax: SET AUTOROWVER ON/OFF

Default: OFF

AUTOROWVER is used for [Oterro](#) compatibility only. If AUTOROWVER is set on, every [CREATE TABLE](#) or [ALTER TABLE](#) command will add the SYS_ROWVER column if it does not already exist. The SYS_ROWVER column is not compatible with R:BASE 6.0 and lower databases.

1.19.8.8 AUTOSKIP

Operating Condition

Syntax: SET AUTOSKIP ON/OFF

Default: OFF

Set AUTOSKIP on to move the cursor automatically to the next data-entry field in a form after filling the entire field. Specify off to press [Tab] after each entry. R:BASE stores the setting with the database.

1.19.8.9 AUTOSYNC

Operating Condition

Syntax: SET AUTOSYNC ON/OFF

Default: OFF

If AUTOSYNC is set on, connecting to a database will automatically synchronize the database files if necessary. If AUTOSYNC is set off and an error occurs during the connect because the files are out of sync, the database is not connected.

See also:

[RBSYNC](#)

1.19.8.10 AUTOUPGRADE

Operating Condition

Syntax: SET AUTOUPGRADE ON/OFF

Default: OFF

AUTOUPGRADE converts R:BASE 6.0 databases to R:BASE eXtreme and adds the new system tables for handling Stored Procedures and Triggers.

1.19.8.11 BELL

Operating Condition

Syntax: SET BELL ON/OFF

Default: ON

Set BELL on to sound the bell when an error occurs. Specify off to suppress the bell. R:BASE stores the setting with the database.

1.19.8.12 BLANK

Special Character

Sets the character for spaces. The BLANK character is used to separate words in a command string.

Syntax: SET BLANK=NULL

Syntax: SET BLANK=char
(Use NULL to disable the special character.)

Default: (space)

1.19.8.13 CAPTION

Operating Condition

Syntax: SET CAPTION 'window title'

SET CAPTION specifies a title for an application or a command file. This title appears in the title bar when you run the application or command file.

1.19.8.14 CASE

Operating Condition

Syntax: SET CASE ON/OFF

Default: OFF

SET CASE sets the uppercase or lowercase distinction when a comparison is used with [WHERE](#) clauses, [IF](#) structures, [WHILE](#) loops, the [TALLY](#) command, and in the [RULES](#) command where comparisons are equal or not equal. R:BASE stores the setting with the database. If CASE is set off, both uppercase and lowercase text are displayed for a comparison regardless of how you enter the text. For example, if you enter "case", you could find "Case" and "CASE".

1.19.8.15 CLEAR

Operating Condition

Syntax: SET CLEAR ON/OFF

Default: ON

Mode: Single-user

SET CLEAR determines when R:BASE clears the internal buffers and transfers the data to disk.

When CLEAR is set on, the internal buffers are cleared and data is transferred to disk after each

modification. Setting CLEAR on does not always make R:BASE automatically write each new or changed row to disk. For example, when you use a form, R:BASE writes the edits to disk when you finish using the form.

Set CLEAR off to write modified data to disk only when the buffer is full, a database is closed, or you exit R:BASE. If CLEAR is set off, repetitive modifications to a database can run faster, but you could lose all of your changes stored in the buffer if an accident, such as a fluctuation in power supply, occurs.

In multi-user mode, the CLEAR setting has no effect and always acts as though it is set on.

1.19.8.16 CLIPBOARD

Environment

Syntax: SET ClipBoard <TextString or Variable>

About the SET CLIPBOARD Command

Places the specified string onto the Windows clipboard, where it can be accessed from other programs.

EXAMPLE 01:

```
SET CLIPBOARD 'Here is text for the clipboard'
```

EXAMPLE 02:

Assuming you have already created a MS Word Document (TestDoc.DOC) or WordPerfect Document (TestDoc.WPD).

```
-- ClipBrd.RMD
CONNECT ConComp
SET VAR vAddressBlock TEXT = NULL
SET VAR vCustID INTEGER = 100
SELECT (Company+(CHAR(10))+CustAddress+(CHAR(10)) +
+CustCity+', '&CustState&CustZip+(CHAR(10))) +
      INTO vAddressBlock INDIC IvAddressBlock +
      FROM Customer WHERE CustID = .vCustID
SET CLIPBOARD .vAddressBlock
LAUNCH TestDoc.DOC
or
LAUNCH TestDoc.WPD
```

Once the MS Word or WordPerfect is launched and the document is opened, you could either use Edit > Paste or Ctrl+V to paste the windows clipboard text!

The resulting pasted block of text would look like:

```
Pc Distribution Inc.
3200 Westminster Way
Boston, MA 02178
```

Above example could be used in your application command file or in an EEP.

1.19.8.17 COLOR

Operating Condition (**R:BASE for DOS ONLY**)

Syntax: SET COLOR
SET COLOR FOREGRND color
SET COLOR BACKGRND color
SET COLOR BACKGRND (redvalue, greenvalue, bluevalue)

You can specify foreground and background colors for the DOS R> Prompt window. You can alter the R:BASE for Windows R> Prompt background and font color by adjusting the settings available from the main Menu Bar under "Settings" > "R> Prompt".

You can also specify colors using the SET command. For example, to change the background color to cyan, enter:

```
SET COLOR BACKGRND cyan
```

You can even select a custom color for the background using a combination of red, blue, and green values. For example, to change the background to orange, enter:

```
SET COLOR BACKGRND (255, 128, 64)
```

Available Colors:

0		Black	8		Light Black
1		Blue	9		Light Blue
2		Green	10		Light Green
3		Cyan	11		Light Cyan
4		Red	12		Light Red
5		Magenta	13		Light Magenta
6		Brown	14		Yellow
7		Gray	15		White

1.19.8.18 COMPATIB

Operating Condition

Syntax: SET COMPATIB ON/OFF

Default: ON

Compatibility with R:BASE Transactions

SET COMPATIB toggles COMPATIBILITY with R:BASE transactions on and off.

The first "compatibility" setting in R:BASE goes back to the 3.1 versions where it was used to allow concurrent access to a database from both 3.1 and 2.11 at the same time. Remember that database files back then were still "rbf" files. Once we went to the "rb1,rb2,..." files the setting was obsolete.

In 1997 when [Oterro](#) was first released, it supported a different scheme for managing transactions (when [TRANSACTIONS](#) are set on) than R:BASE itself used at that time. This new scheme used a different file to track transactions and had a larger allocated buffer size on the file to manage ongoing transactions. To make R:BASE compatible with Oterro when transactions were on it needed to support the new method, but it also needed to be able to support the older style that previous versions of R:BASE used (version 6.0 and older).

The first R:BASE version that could support the two methods was 6.1. When the "compatibility" setting is

on, R:BASE will use the older "non-Oterro" style of transaction. When "compatibility" is off, the Oterro method will be used. If you never run with TRANSACTIONS ON then the compatibility setting does not have any effect.

1.19.8.19 CURRENCY

Operating Condition

Syntax: SET CURRENCY \$ PREF 2 B

SET CURRENCY sets the symbol, location, subunits, and format for currency values. R:BASE stores the setting with the database.

Changing the CURRENCY parameters affects all columns in the database that have a CURRENCY data type. You must enter the parameters in the following order: SYMBOL, PREF or SUFF, digits, and format; that is, even if you want to change the digits only, you must also enter the symbol and its position.

- **Symbol** (default \$)--A symbol is any ASCII character or string of one to four characters. You can include a space in place of one character at the beginning or end of the string, but if you do, enclose the string in quotation marks.
- **PREF and SUFF** (default PREF)--Specify the position of the symbol as before (PREF) or after (SUFF) the currency value. In the SET CURRENCY command, enter a space between the symbol and its position, PREF or SUFF.
- **Subunit digits** (default 2)--Indicates the number of digits from 0 to 16 to be displayed in a currency subunit. In the case of dollars, the subunit is cents, so the digits setting for dollars is 2. For example, setting digits to 3 will display currency values similar to these:

```
20.000,000DM
20,000.000
2,000.000
```

If you change digits when the database contains data, the new digit setting affects how R:BASE displays and uses the data already entered. For example, if you change the setting from 2 to 4, an existing value such as 1,234.00 becomes 12.3400.

- **Format** (default B)--Format specifies how R:BASE displays the thousands and decimal delimiters. A, B, and C specify how the thousands and decimal delimiter displays for values with CURRENCY, REAL, and DOUBLE data types. Before you change the format, you must change the DELIMIT character.

Delimiter Conventions for CURRENCY, REAL, and DOUBLE Values

Convention	Thousands Delimiter	Decimal Delimiter	Example
A	.	,	123.456.793,01
B	,	.	123,456,793.01
C	(blank)	,	123 456 793,01
D	N/A	.	123456793.01

For example, to display currency in two digits with a prefix of DM (deutsche marks) with the display format A, at the R> Prompt, enter:

```
SET DELIMIT=!
SET CURRENCY DM PREF 2 A
```

This command displays 1,500 deutsche marks and 25 pfennigs in the format like this:

```
DM 1.500,25.
```

If you set CURRENCY to the delimiter formats A or C without changing DELIMIT to a character other

than the comma, the default setting for DELIMIT, R:BASE displays the following message: "Decimal character cannot be the same as DELIMIT."

In this case, use [SET DELIMIT](#) to change the delimiter to a less commonly used character such as an exclamation point; then set CURRENCY to the delimiter formats *A* or *C*.

1.19.8.20 DATE

Operating Condition

Syntax: SET DATE CENTURY *value*
SET DATE YEAR *value*
SET DATE MM/DD/YY (date sequence and format)
SET DATE SEQ MMDDYY (date sequence)
SET DATE FOR MM/DD/YY (date format)

SET DATE sets the date sequence for entry, and format for display. A valid date can have up to 30 characters. R:BASE stores the setting with the database.

Use the SET DATE CENTURY *value* command to set the default century (the first two digits of a four-digit year). For example, if you enter a two-digit year and you want it to default to the twenty-first century, enter the following command:

```
SET DATE CENTURY 20
```

A year such as "25" would be stored as "2025."

Use the SET DATE YEAR *value* command to have two default centuries for dates entered, depending on the year. All years from 00 to (YEAR *value*-1) are stored with the next century (CENTURY *value*+1), and all years from *value* to 99 are stored with the default century (CENTURY *value*). For example, you can have all dates from the year 50 to 99 default to the twentieth century, and all dates from 00 to 49 default to the twenty-first century by entering the following commands:

```
SET DATE CENTURY 19  
SET DATE YEAR 50
```

The SET DATE CENTURY 19 command sets the default century to 19. The SET DATE YEAR 50 command stores all years from 50 to 99 with the default century, 19. All dates from 0 to 49 are stored with the next century, 20. Therefore, years entered from 50 to 99 are stored as 1950 to 1999, and years entered from 00 to 49 are stored as 2000 to 2049.

Note: The DATE CENTURY and DATE YEAR options can only be used when the date format and sequence options are set to the defaults (SET DATE FOR MM/DD/YY and SET SEQ MMDDYY).

R:BASE accepts a date between January 1, 3999 BC and December 31, 9999 AD. You can set the date sequence and format separately. R:BASE displays the date based on the format. When setting the format to display numerals for the month, day, and year, use a separator such as the slash (/), hyphen (-), comma (,) or space (blank). For example, if you set the date format to MM/DD/YY and enter 061193, R:BASE displays 06/11/93.

You can also include text for the weekday and month in the date format to a maximum of 30 characters. Include WWW for a three-letter day abbreviation, WWW+ for the full day name, MMM for a three-letter month abbreviation, and MMM+ for the full month name. If the date format contains spaces or commas, enclose the format in quotes. For example, if the special character for QUOTES is set to the R:BASE default ('), the format 'MMM DD, YYYY CC' displays Jun 11, 1993 AD.

If you use YY in the date format, R:BASE displays only the last two digits of the year. To view dates in other centuries, use a date format with a four-digit year such as 'MM DD, YYYY'. If you use BC dates, add CC to the format. Dates entered with BC are shown with BC; otherwise, the date is shown with AD. For example, you could use the sequence MMDDYY and the format 'MMM DD, YYYY CC' to accept and display BC dates. If you enter '06 11 93BC,' R:BASE displays Jun 11, 0093 BC.

Example: Valid date formats, using June 11, 1993

Date Format	Display
'MMM+ DD' MM/YY	June 11 06/93
'WWW the DD' 'WWW+, MMM+ DD, YYYY CC'	Sun the 11 Sunday, June 11, 1993 AD

You can omit the SEQ and FOR keywords to set both sequence and format in a single SET DATE command. For example, enter SET DATE MM/DD/YYYY to set both date sequence and format to a four-digit year.

Enter the date in any form as long as the sequence of M's, D's, Y's, and C's are in the same order defined for the date sequence. The display, however, is always exactly as defined by the DATE format.

If, for example, you set the date sequence to a four-digit year with SET DATE SEQ MMDDYYYY, set the date format to a two-digit year with SET DATE FOR MM/DD/YY, and later enter a two-digit year, R:BASE will store and might display a date you do not expect. As the following table shows, if you enter 06/11/93, R:BASE stores the date as 06/11/0093 and displays 06/11/93.

How R:BASE stores and displays dates

DATE Sequence	Date Entered	Date Stored
MMDDYYYY	6/11/94	06/11/0094
MMDDYY	6/11/0094	06/11/1994
MMYY	6/11	06/01/1994
MMDD	6/11	06/11/1994
DDYY	6/11	01/06/1911
DDYY	11/94	01/11/1994

DATE Format (2-digit Year)	DATE Format (4-digit Year)
06/11/94	06/11/0094
06/11/94	06/11/1994
06/01/94	06/11/1994
06/11/94	06/11/1994
01/06/11	01/06/1911
01/11/94	01/11/1994

If the sequence is set to a four-digit year and the format is set to a two-digit year, R:BASE stores the date you enter, such as 06/11/95, as a four-digit year. As a result, if you use a WHERE clause to display rows that have dates greater than 06/11/95, R:BASE returns all rows greater than 06/11/0095.

The DATE format can affect date functions. For best results, set the format to the default MM/DD/YY and then use a date function.

1.19.8.21 DEBUG

Operating Condition

Syntax: SET DEBUG ON/OFF

Default: OFF

You can use SET DEBUG as follows:

1. Precede any R:BASE command you want to control with the DEBUG modifier in a command file.
2. Set DEBUG on to have R:BASE run the command; set DEBUG off to have R:BASE ignore the command.

1.19.8.22 DELIMIT

Special Character

Separates a character, string, or items in a list used in commands. Also used to separate repeatable parts of a command.

Syntax: SET DELIMIT=NULL
SET DELIMIT=char
(Use NULL to disable the special character.)

Default: ,

1.19.8.23 ECHO

Operating Condition

Syntax: SET ECHO ON/OFF

Default: OFF

SET ECHO displays or suppresses commands as they are processed from the current ASCII input device. Specify on or ECHO to display commands; specify ECHO off or NOECHO to turn off the command display.

Use SET ECHO as a debugging technique as you develop a command file. With ECHO set on, you can see the commands as they are processed when you run a command file. SET ECHO works only when the command file is an ASCII file; it will not display commands that were run from a binary procedure file.

Enter a SET ECHO ON command at the beginning of the program. Then, when the program runs, R:BASE displays each command as it is interpreted and, if needed, runs it. Sometimes the commands scroll on the screen faster than you can read, especially if they are read by R:BASE but not run. You can temporarily stop the display by pressing [Ctrl]+[Break], which stops the file from running, and restart it by pressing [Enter], or stop completely by pressing [Esc]. Set ECHO off to suppress command display.

Even more useful, you can direct output to a printer or a file before you set ECHO to on. Then, when the command file runs, the commands and any errors are saved either in printed form or in a file you can look at.

When R:BASE runs the commands below in an ASCII file, you see the commands on lines two, three, and four displayed at the top of the screen and the message displayed on line 10 beginning at column 20.

```
CLS
SET ECHO ON
WRITE 'This is a message' at 10 20
SET ECHO OFF
```

1.19.8.24 EDITOR

Environment Setting

Syntax: SET EDITOR RBEDIT/filespec

Default: RBEDIT

The SET EDITOR setting allows you to specify the internal R:BASE Editor or some other text editor as your default text editor for R:BASE command files.

For example, if you wish to alter the default text editor to the external R:BASE Editor program, you would use the following syntax:

```
SET EDITOR C:\RBTI\RBEdit\RBEdit.exe
```

1.19.8.25 EOFCHAR

Operating Condition

Syntax: SET EOFCHAR ON/OFF

Default: ON

If EOFCHAR is set OFF, a control-Z character will not be appended to the end of output files.

1.19.8.26 EQNULL

Operating Condition

Syntax: SET EQNULL ON/OFF

Default: OFF

This Command determines whether or not NULL = NULL.

Compare these code samples:

```
SET VAR v1 TEXT = NULL
SET VAR v2 TEXT = NULL
```

```
SET EQNULL OFF
IF v1 = .v2 THEN
  -- will not be a hit
ENDIF
IF v1 <> .v2 THEN
  -- will not be a hit
ENDIF
IF v1 <> 'This' THEN
  -- will not be a hit (it used to be before this fix)
ENDIF
```

```
SET EQNULL ON
IF v1 = .v2 THEN
  -- will be a hit
ENDIF
IF v1 <> .v2 THEN
  -- will not be a hit
ENDIF
IF v1 <> 'This' THEN
  -- will be a hit
ENDIF
```

Before this fix, the comparison "IF v1 <> 'This' THEN" would be a hit with EQNULL set ON or FALSE when it should only be a hit when EQNULL is ON. This means that now "IF (.v1) <> 'This' THEN" and "IF v1 <> 'This' THEN" will both process the same way. In the past they would be different because of this problem.

In your code if you want the comparison of a NULL variable and a non-NULL constant to be a hit then you should run with EQNULL set ON.

1.19.8.27 ERRORMESSAGE

Special Condition

Syntax: SET ERROR MESSAGE Error# ON/OFF

Default: ON

SET ERROR MESSAGE Error# ON/OFF displays or suppresses a particular error message when a system error occurs.

You can selectively turn OFF any -ERROR- message(s) in your command file (very handy for debugging) by doing the following:

```
SET ERROR MESSAGE Error# OFF
```

To turn it back ON:

```
SET ERROR MESSAGE Error# ON
```

For example, to not see the:

```
-WARNING- No rows exist or the specified clause.
```

You can do the following:

```
SET ERROR MESSAGE 2059 OFF
```

This new feature has a limit of 50 -ERROR- numbers to set OFF and each one requires a separate command.

NOTE: Each turned OFF message must be turned back ON before turning it OFF again. If not, you'll get the error message.

1.19.8.28 ERRORMESSAGES

Operating Condition

Syntax: SET ERROR MESSAGES ON/OFF

Default: ON

SET ERROR MESSAGES displays or suppresses an error message when a system error occurs. Specify off to suppress error and rule violation messages.

1.19.8.29 ERRORVARIABLE

Operating Condition

Syntax: SET ERROR VARIABLE varname
SET ERROR VARIABLE OFF

SET ERROR VARIABLE defines an error variable to hold error message numbers. The variable name (*varname*) defines the variable R:BASE uses to hold R:BASE error codes. If set to off (the default), error variable processing is removed.

When an error occurs in a command file, R:BASE normally displays a system error message. SET ERROR enables a programmer to anticipate errors in command and procedure files and program the file to keep running even when an error occurs.

You must always set ERROR VARIABLE off, rather than clearing it with the [CLEAR VARIABLES](#) command.

R:BASE resets the error variable to zero as each command is successfully run. If an error occurs, the error variable is set to the error number value. To determine the error condition for any line, you must immediately check the value of the error variable or capture the error value in a global variable for later

examination.

By checking the error variable for a non-zero value, you can detect (or trap) many errors and run a sequence of error-handling commands such as an error-recovery procedure. Once the error number is captured in an error variable, you can write error-handling command files to control a program's flow based on these errors (error values).

The error variable value is set for each command that is run, not each line in a command file. If you have placed multiple commands on a line, the last command's error value is placed in the error variable. A similar situation occurs for multi-line commands such as the subcommands you can use when loading a data block with the [LOAD](#) command. For example, a data block loaded with the LOAD command leaves the error variable with a value of zero because the END command runs successfully, whether or not the data is actually loaded.

Rule violations do not set the error variable to a non-zero value; they are not the same as errors recognized by R:BASE.

The command below defines *errvar* as the current error variable:

```
SET ERROR VARIABLE errvar
```

When a command is run, R:BASE sets the error variable *errvar* to the error code before anything else happens. The following command lines illustrate how to use *errvar* in a command file.

```
LABEL tryagain
FILLIN vdbname USING 'Enter the database name: '
CONNECT .vdbname
IF errvar <> 0 THEN
    WRITE 'Database not found.'
    GOTO tryagain
ENDIF
```

The first command establishes a label to return to, the second requests that the user enter the name of a database, and the third opens the specified database using the global variable defined by the [FILLIN](#) command.

The [IF...ENDIF](#) structure checks the error variable value. If the value is not zero (that is, if the database was not opened successfully), then it sends a message to the screen and passes control to the label *tryagain* so that the user is asked to enter the database name again.

You can also write a separate command file specifically designed to handle a variety of errors. In this case, the above code might look like this:

```
FILLIN vdbname USING 'Enter the database name: '
CONNECT .vdbname
SET VARIABLE verr1 = .errvar
IF verr1 <> 0 THEN
    RUN errhandl.cmd USING .verr1
ENDIF
```

This series of commands captures the error value in the global variable *verr1* so that it can be passed through the USING clause of the [RUN](#) command to an error-handling routine. The routine itself determines the nature of the error and how to take care of the problem.

You can use the [WHENEVER](#) command to run status-checking routines for SQL commands. WHENEVER uses the special R:BASE variable SQLCODE.

1.19.8.30 ESCAPE

Operating Condition

Syntax: SET ESCAPE ON/OFF

Default: ON

SET ESCAPE allows you to use [Ctrl]+[Break] to escape or abort command file processing or database file access. Specify on to enable users to abort processing in the middle of command files, [WHILE](#) loops, and database access. Specify off to prevent users from prematurely aborting a command file or an application such as when R:BASE runs processes that create new tables (such as the relational commands) from within a command.

1.19.8.31 EXPLODE

Operating Condition (**R:BASE for DOS only**)

Syntax: SET EXPLODE ON/OFF

Default: OFF

Controls how DOS dialogs are displayed.

When EXPLODE is set on, dialog boxes are displayed in full size instantly. When EXPLODE is set off, dialog boxes are displayed in an expanding fashion from the center.

1.19.8.32 FASTFK

Operating Condition

Syntax: SET FASTFK ON/OFF

Default: OFF

This setting, when on, permits R:BASE to operate a foreign key index using a condensed index for maintaining that foreign key. If the foreign key is not used for retrieving data or linking columns, a complete index is unnecessary and actually inhibits speed. When set to on, R:BASE creates a condensed index for any existing foreign keys.

To switch to condensed indexes on existing foreign keys, you need to run a [PACK](#), [PACK KEYS](#), or [RELOAD](#) command with FASTFK on; these actions cause R:BASE to rebuild the database with condensed foreign key indexes.

Keep in mind, however, that you might need complete indexes on foreign keys where such indexes are needed for retrieving data. Retaining a separate index on columns used in foreign keys that link tables is preferred. Indexes are also needed on foreign keys that you use for selecting column values; therefore, use the [CREATE INDEX](#) command to explicitly create indexes for columns used in foreign keys in a database where FASTFK is set on.

The command [SHOW](#) FASTFK displays the FASTFK state and whether FASTFK is operational in the current database. For example:

```
SHOW FASTFK
(FASTFK ) ON Use fast Foreign Key (FK) structures on rebuild.
OFF FASTFK setting for current database
```

Once you rebuild the keys in a database with the FASTFK setting on, SHOW FASTFK displays the following:

```
SHOW FASTFK
(FASTFK ) ON Use fast Foreign Key (FK) structures on rebuild
ON FASTFK setting for current database
```

1.19.8.33 FASTLOCK

Operating Condition

Syntax: SET FASTLOCK ON/OFF

Mode: Multi-user and [STATICDB](#)

Set FASTLOCK on for faster multi-user performance while modifying data. With FASTLOCK on, R:BASE does not place a table lock on the table, allowing for greater throughput. A table lock is only needed to prevent structure changes.

FASTLOCK can only be set on when STATICDB is set on, and both FASTLOCK and STATICDB must be set on before the database is connected. Like other R:BASE database modes (SET MULTI and SET STATICDB), FASTLOCK requires all users to be connected with the same setting.

The following command lines set STATICDB and FASTLOCK correctly.

```
SET STATICDB ON
SET FASTLOCK ON
CONNECT concomp
```

See Also:

Environment Optimization

1.19.8.34 FEEDBACK

Operating Condition

Syntax: SET FEEDBACK ON/OFF

Default: OFF

This setting displays processing results when either calculating or editing rows. With this setting on, R:BASE displays the number of rows processed and the elapsed time to completion in a dialog window. Displays occur while in the Data Browser, when printing reports and labels, and when using certain R:BASE commands including:

1. [ALTER TABLE](#)
2. [COMPUTE](#)
3. [CREATE INDEX](#)
4. [CROSSTAB](#)
5. [DELETE](#)
6. [INSERT](#)
7. The data transfer for the [JOIN](#) of two tables
8. [LOAD](#)
9. The data transfer for a [PROJECT](#) command
10. [SELECT](#)
11. SORTing a huge record set
12. the data transfer for a [SUBTRACT](#) of two tables
13. [TALLY](#)
14. The [UNION](#) command

When using any of the various commands at the R> Prompt with FEEDBACK set ON, the **#FEEDBACK** system variable is generated to hold the elapsed time for the command. To display to the elapsed time, use the following:

```
PAUSE 2 USING .#FEEDBACK
```

1.19.8.35 FILES

Operating Condition

Syntax: SET FILES value

Range: 1 to 255 files

Default: 5

SET FILES sets the maximum number of files that can be open at a time. The maximum, depending on available memory, is 255.

1.19.8.36 FIXED

Operating Condition

Syntax: SET FIXED ON/OFF

Default: ON

Controls column width in [SELECT](#)

This controls whether R:BASE will automatically shrink column widths in SELECT commands.

1.19.8.37 FONT

Operating Condition (**R:BASE for DOS ONLY**)

Syntax: SET FONT keyword

Default: OEM

The FONT setting changes the font used in the "R> Prompt" window.

You can choose from three settings, which are mono-spaced stock fonts used in Windows:

- System--fonts compatible with the system font in Windows.
- OEM--an IBM PC character set for IBM computers.
- Ansi--a fixed-pitch font based on the Windows character set. A Courier font is typically used.

1.19.8.38 HEADINGS

Operating Condition

Syntax: SET HEADINGS ON/OFF

Default: ON

SET HEADINGS displays columns with or without headings when you enter the [SELECT](#) and [TALLY](#) commands.

1.19.8.39 IDQUOTES

Operating Condition

Syntax: SET IDQUOTES

Default: Reverse Quote (`) Prior Versions: NULL

Controls the character that is used to set off object names.

IDQUOTES sets the character for enclosing object names, such as column or table names, in R:BASE and ODBC commands. This is especially critical when using ODBC to connect to a non-R:BASE Database that allows characters such as spaces in Column or Table names. This is also critical to SOME internal R:

BASE processing.

Note: Older databases may default to NULL which is not ODBC or SQL compliant.

In general, setting IDQUOTES will have no effect on legacy applications as they will not, in most cases, know that this exists. The one exception to this is unloading data from a database that has IDQUOTES set, such as an current upgraded R:BASE database, and importing into a Legacy database. If you are attempting to do this you should set your IDQUOTES to NULL.

The Reverse Quote is located, on most standard US Keyboards, under the Tilde (~) character and to the left of the numeral 1.

1.19.8.40 INDEXONLY

Operating Condition

Syntax: SET INDEXONLY ON/OFF

Default: ON

Sets a flag to disable "index only" select retrievals.

1.19.8.41 INSERT

Operating Condition

Syntax: SET INSERT ON/OFF

Default: ON

SET INSERT turns insert/overwrite on or off. Set INSERT on to use either the insert or overwrite mode. Pressing the [Insert] key when you have specified INSERT to be on toggles you between insert mode and overwrite mode. In insert mode, the cursor indicator is larger. Press the space bar to insert a space. Set INSERT off to use only the overwrite mode; the cursor indicator is smaller.

1.19.8.42 INTERVAL

Operating Condition

Syntax: SET INTERVAL value

Default: 5

Range: 0 to 9 tenths of a second

Mode: Multi-user

The SET INTERVAL command specifies the time to elapse before R:BASE retries the command that caused a conflict within the waiting period. Also, see [WAIT](#).

1.19.8.43 KEYMAP

Operating Condition

Syntax: SET KEYMAP keyname OFF
SET KEYMAP ALL OFF
SET KEYMAP keyname=remapped keys

You can define key maps with a single statement, which enables you to define key maps in command files. For example:

```
SET KEYMAP [SHIFT][F3] TO [F2]  
SET KEYMAP [ALT]M= [F2]
```

To return to the original key mapping, enter the following:

```
SET KEYMAP [ALT]M OFF
```

1.19.8.44 LAYOUT

Operating Condition

Syntax: SET LAYOUT ON/OFF

Default: ON

SET LAYOUT switches saving layouts on or off. When LAYOUT is set on, R:BASE saves the layout of data displayed in the Data Browser when you exit. R:BASE saves layouts for single tables only. The next time you display the table with the Data Browser, the layout of data will be as you previously arranged it.

When LAYOUT is set off, R:BASE does not save the layout of data and ignores any saved layouts. If you want to see a table displayed in its default format without changing the saved layout for it, set LAYOUT off before displaying the table.

1.19.8.45 LINEEND

Special Character

Syntax: SET LINEEND value

Default: [Alt] + [0254]

You can set an end of line ASCII character for NOTE and TEXT fields in forms, reports and the Data Browser. When you insert the character in those fields and then zoom in by pressing [SHIFT]+[F2] or print a report, you see lines break as established by the line end character. The default character is ASCII value 0254.

Note: If the line end character has been set to the currency character, R:BASE changes it to ASCII value 0254 when you connect to a database.

1.19.8.46 LINES

Operating Condition

Syntax: SET LINES value

Range: 0 to 32,767 lines

Default: 20

SET LINES sets the number of lines per page or screen when you use the [CROSSTAB](#), [DISPLAY](#), [DIR](#), [LIST](#), [OUTPUT](#), [SELECT](#), [LIST RULES](#), [SHOW VARIABLES](#), [TALLY](#), or [TYPE](#) commands. LINES does not affect report generation; you can define the number of lines on a page for each report. Setting LINES to zero displays lines as continuous output.

1.19.8.47 LOCK

Operating Condition

Syntax: SET LOCK tbllist ON/OFF

Default: OFF

Mode: [Multi-user](#)

SET LOCK manually sets locks on or removes locks from tables specified in the list of tables (*tbllist*). Use the command whenever you want a procedure or transaction to have exclusive use of tables. Setting LOCK to off disables locks for each of the tables in *tbllist*.

In command or procedure files it is sometimes necessary to prevent access to a table or group of tables

while certain operations are performed. Although R:BASE handles most locks automatically according to the command that is running, at times you might want more control over table locking. SET LOCK provides you explicit control over access to tables during processing by the commands that retrieve and update data.

If R:BASE cannot lock all the tables listed after SET LOCK, it issues a message saying that not all tables are available to be locked. R:BASE does not lock any tables unless it can lock all tables listed, and it sets an error code when SET LOCK fails.

Group the tables used into one SET LOCK command to avoid tying up needed resources. Be sure to issue the SET LOCK OFF command to remove the locks after processing is complete. Locks set with this command are cumulative. You need to issue one SET LOCK OFF command for each SET LOCK ON command that you have entered for a given table. The user who set the table locks must issue SET LOCK OFF; otherwise no other user can access the locked tables until the first user exits the database.

[LIST](#) displays locked tables in reverse video. With LIST TABLE, on the other hand, the type of multi-user locks is displayed. Only the highest priority lock is displayed for each table.

LIST TABLE tells you whether the lock is an edit, row, cursor, local, or remote lock. Edit, row, and cursor locks are set by R:BASE as part of its internal concurrency control. A local lock is set by a SET LOCK command issued at the workstation that issued the LIST TABLE command. And a remote lock is set by a command that obtains a table lock and is issued from a workstation other than the workstation that issued the [LIST TABLE](#) command.

The first command line below sets an exclusive lock on the *customer* table. The second command line sets additional exclusive locks on the *transmaster* and *transdetail* tables. These exclusive locks prevent access to the three tables by any user other than the one who issued the SET LOCK ON commands. The SET LOCK OFF command removes the locks on all three tables.

```
SET LOCK customer ON
SET LOCK transmaster, transdetail ON
SET LOCK customer, transmaster, transdetail OFF
```

1.19.8.48 LOOKUP

Operating Condition

Syntax: SET LOOKUP value

Default: 5

SET LOOKUP tells R:BASE how many form look-up expressions to store in memory. Storing a look-up expression in memory enables R:BASE to display data more quickly in a form. The number of look-up expressions you can specify depends on the memory available in your computer. R:BASE needs approximately 500 bytes of RAM for each look-up expression. SET LOOKUP does not affect master look-up expressions or pop-up menus in a form.

For example, when your form contains 10 look-up expressions and LOOKUP is set to 5, only the first five look-up expressions are stored in memory. R:BASE must retrieve/reevaluate the remaining look-up expressions. So that the form works faster, you can set LOOKUP to a higher value to store more look-up expressions in memory.

1.19.8.49 MANOPT

Operating Condition

Syntax: SET MANOPT ON/OFF

Default: OFF

MANOPT set to OFF disables the automatic table-order optimization that R:BASE performs when running queries. This gives maximum control over the order in which columns and tables are assembled in response to a query.

With MANOPT set to ON, R:BASE uses the order of the tables in the [FROM](#) clause and the order of the

columns in the column list of the [SELECT](#) clause to construct the query.

When MANOPT is set on, the **#TABLEORDER** system variable stores the table join order, and the applicable indexed columns. To display the table order, use the following:

PAUSE 2 USING .#TABLEORDER

See Environment Optimization

1.19.8.50 MANY

Special Character

Sets the character for the many wildcard for R:BASE commands and clauses.

Syntax: SET MANY=NULL
SET MANY=char
(Use NULL to disable the special character.)

Default: %

1.19.8.51 MAXTRANS

Operating Condition

Syntax: SET MAXTRANS value

Range: 1 to 255

Default: 10

Mode: Transaction Processing

SET MAXTRANS specifies the maximum number of users who can have the same database open concurrently with transaction processing on.

Only the first user to connect to a closed database can enter the MAXTRANS setting for that database. Enter the command before connecting to the database. If anyone else already has the database open, R:BASE displays a message telling you that your SET MAXTRANS command will have no effect on the database.

MAXTRANS can affect system performance. The higher the MAXTRANS setting, the more overhead the system must carry to process transactions across the network. Also, the more users who are actually entering transactions, the slower the system operates.

1.19.8.52 MDI

Operating Condition

Syntax: SET MDI ON/OFF

Default: ON

Use the SET MDI command to launch forms from the Database Explorer, toolbar, or the R> Prompt as MDI windows.

1.19.8.53 MESSAGES

Operating Condition

Syntax: SET MESSAGES ON/OFF

Default: ON

SET MESSAGES either displays or suppresses system messages. Set MESSAGES to off when ERROR is set on to display only error messages.

1.19.8.54 MIRROR

Operating Condition

Syntax: SET MIRROR <path>

Syntax: SET MIRROR OFF/DELETE

Default: OFF

SET MIRROR <path> maintains a duplicate copy of the database. This duplicate copy is created and maintained in the directory designated in *path*. The duplicate database will have the same name, therefore *path* must designate a backup directory. With this setting, all modifications to the original database are duplicated in the mirrored database. Be sure all users are mapped to the save drive letters. SET MIRROR OFF turns off mirroring of the database; SET MIRROR DELETE turns off mirroring and then deletes the duplicate database.

1.19.8.55 MOUSE

Operating Condition (**R:BASE for DOS ONLY**)

Syntax: SET MOUSE -1 to 100

Default: 30

Controls how DOS dialogs are displayed.

When [EXPLODE](#) is set on, dialog boxes are displayed in full size instantly. When EXPLODE is set off, dialog boxes are displayed in an expanding fashion from the center.

Controls mouse sensitivity

Used with R:BASE for DOS only. SET MOUSE controls the period of time in hundreths of a second in which the mouse registers a double click. Setting the time too low makes it impossible to double-click the mouse. A setting of -1 disables the mouse. To set the mouse speed each time you use R:BASE, include the SET MOUSE command in a startup file.

1.19.8.56 MULTI

Operating Condition

Syntax: SET MULTI ON/OFF

Default: OFF

SET MULTI sets Multi-User capability on or off when you next connect a database. This setting must be used while you are disconnected from a database.

1.19.8.57 NAME

Operating Condition

Syntax: SET NAME network identification

Default: USER***** (USER and 14 numerals for date and time)

Mode: Multi-user

SET NAME specifies a network identification for your system when you start R:BASE. NAME is text and can contain spaces.

NAME must be saved to the RBENGINE9.CFG file.

1.19.8.58 NAMEWIDTH

Operating Condition

Syntax: SET NAMEWIDTH 0 to 100

Default: 18

SET NAMEWIDTH controls the name width of a table, column, form, report, label, etc. that R:BASE directs to the printer, screen, or file when using the [BACKUP](#), [COMPUTE](#), [CROSSTAB](#), [DISPLAY](#), [SELECT](#), [TYPE](#), [UNLOAD](#), or [WRITE](#) commands. Do not set the width to a number greater than the number of characters your printer can fit on a line; a typical page and computer screen display 80 characters. [WIDTH](#) does not affect report generation; each report defines the width of a data line.

1.19.8.59 NOTE_PAD

Operating Condition

Syntax: SET NOTE_PAD

Range: 0 to 100 (percent)

Default: 10

This setting allocates an additional percentage of storage space in NOTE columns to accommodate value increases (additional text), so that rows don't need to move to different disk locations. This increases performance by reducing disk reading.

The default setting is 10% of the row size, and can be set from 0 to 100%. Setting NOTE_PAD to 0% disables padding.

1.19.8.60 NULL

Special Character

Syntax: SET NULL -0- (1 to 4 characters)

Default: -0-

SET NULL sets the display symbol for null. You can use up to four characters. If you set null to a blank space, R:BASE does not display rows composed entirely of nulls with the [SELECT](#) command. R:BASE stores the setting with the database.

If you enter the following command, R:BASE displays a slash in the absence of data:

```
SET NULL /
```

1.19.8.61 ONELINE

Operating Condition

Syntax: SET ONELINE ON/OFF

Default: OFF

Controls text and note field wrapping.

When set to ON NOTE and TEXT fields will never wrap to the next line in Reports and [SELECT](#) commands. Instead they will be truncated at the end of the column.

1.19.8.62 PAGEMODE

Operating Condition

Syntax: SET PAGEMODE ON/OFF

Default: OFF

Mode: Single- or Multi-user

Use PAGEMODE to design reports through a custom R:BASE program. With PAGEMODE on, you create a page of a report in memory, then send the report to a printer or file. PAGEMODE is an alternative method to creating reports through the Report Designer; the two methods work in different ways.

The [WRITE](#), [SHOW VARIABLE](#), and [SELECT](#) commands are used to "display" data on a virtual page. To determine the row location of the cursor on the virtual page, use the ISTAT function with the keyword PAGEROW after a SHOW VARIABLE command. (ISTAT('pagerow') does not work with the WRITE command.) The [DECLARE CURSOR](#) command is usually used for retrieving data for printing. You control form feeds (new pages) in your program by using the [NEWPAGE](#) or [OUTPUT SCREEN](#) command.

At the beginning of your program, set both [SET LINES](#) and [SET WIDTH](#) to accommodate the report's size. Then set PAGEMODE to on. You cannot write to a line longer than the current LINES setting or wider than the current WIDTH setting. You also cannot change the LINES and WIDTH setting without setting PAGEMODE to off first.

To send printer control codes to a printer, use the CHAR function to define a variable containing the printer control codes. Then, that variable is sent to the printer using the SHOW VARIABLE or WRITE command. For example,

```
SET VARIABLE vLandscape = (CHAR(27) + CHAR+
                           (38) + CHAR(108) + CHAR+
                           (49) + CHAR(79))
WRITE .vLandscape
```

The printer control codes can be found in the user's manual for your printer. PAGEMODE allows you to:

Create reports wider than 255 columns and longer than 84 lines.

- Produce multi-column reports.
- Design different styles for the pages of the report, such as a report with a personalized letter as the first page.
- Create reports from tables with many-to-many relationships that have two or more detail sections.
- Format odd and even pages differently throughout a report.
- Print headers and footers only on the last page or first page.
- Customize reports so break headers and footers are located on the same page.
- Place a different break header on every page or the same break header on every page.

The following command line sets PAGEMODE on:

```
SET PAGEMODE ON
```

For more information on using PAGEMODE, please see Using PAGEMODE.

1.19.8.63 PASSTHROUGH

Operating Condition

Syntax: SET PASSTHROUGH ON/OFF

Default: OFF

When PASSTHROUGH is set on, [SELECT](#) statements are sent directly to the foreign data source and are not processed by R:BASE. Set PASSTHROUGH on when you use special syntax, such as non-SQL syntax, or syntax that is not supported by R:BASE.

1.19.8.64 PLUS

Special Character

Sets the character for the command line continuation character.

Syntax: SET PLUS=NULL

Syntax: SET PLUS=char
(Use NULL to disable the special character.)

Default: +

1.19.8.65 POSFIXED

Operating Condition (**R:BASE for DOS ONLY**)

Syntax: SET POSFIXED ON/OFF

Default: OFF

Controls how the AT parameter works.

When you use *AT row,col* to position dialog, pause and other windows, the actual position depends on the current font size of the R> Prompt window. If you always want the calculation to use the 8x12 size of the OEM font then set POSFIXED to ON. If you want the dimensions of the current font to be used then set POSFIXED to OFF.

1.19.8.66 PRINTER

Operating Condition (**R:BASE for DOS ONLY**)

Syntax: SET PRINTER printername

Controls the DOS printer

SET PRINTER specifies the printer for your system. The configuration file reads the printer file and sets the printer values as variables for the printer.

```
SET PRINTER Epson.prd
```

This command specifies an Epson printer as the printer file named in the configuration file. The extension .PRD is not required.

1.19.8.67 PROCEDURE

The SET PROCEDURE command locks a procedure so it can be replaced.

**SET PROCEDURE procname LOCK [ON
OFF]**

Area: Stored Procedures & Triggers

Options**ON**

Enables a lock

OFF

Disables a lock set by SET PROCEDURE or GET LOCK.

About the SET PROCEDURE Command

The SET PROCEDURE works like the [GET LOCK](#) command without retrieving the Stored Procedure into an ASCII file. ON sets the lock; OFF releases the lock placed by the SET PROCEDURE or the GET command.

When a procedure is locked, only the user placing the lock can replace the procedure or remove the lock. The NAME setting is used for identification of the user.

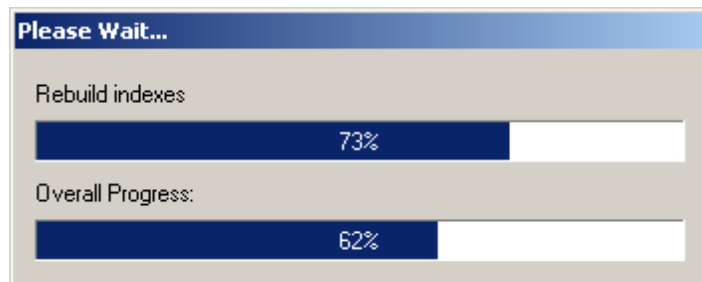
1.19.8.68 PROGRESS

Operating Condition

Syntax: SET PROGRESS ON/OFF

Default: OFF

This setting displays processing results when building indexes, packing or reloading a database. With this setting on, R:BASE displays the process being performed, the overall progress, and the completion percentage of each.



1.19.8.69 QUALCOLS

Operating Condition

Syntax: SET QUALCOLS (2 or 10)

Mode: Multi-user

Default: 10

Use QUALCOLS to choose how R:BASE locks data

The settings for QUALCOLS are:

- **10** - R:BASE uses page locking or row locking as appropriate. When QUALCOLS is set to 10 and two or more users are updating rows within the same page of data, R:BASE only lets the first user update rows--the other users are locked out until the first user's update has been completed.
- **2** - R:BASE uses a fast row-locking method where only row locking is applied with no page locking. When QUALCOLS is set to 2, you can lock rows of data instead of locking a page of data. You increase multi-user performance when QUALCOLS is set to 2. And even more so when [STATICDB](#) and [FASTLOCK](#) are on.

If you know that your application mainly updates or deletes data a row at a time, rather than many rows, set QUALCOLS to "2" for row locking. In this case, R:BASE locks a row, reads the row, makes the change, and then releases the row. Otherwise, set QUALCOLS to 10 for page locking when you are doing an [UPDATE](#) and/or [DELETE](#) affecting many rows in a table.

Keep in mind that the QUALCOLS setting can be changed dynamically and can be different for different users using the same database.

Technically, the efficient and fastest method for updating data in multi-user environment is to SET

STATICDB ON, SET FASTLOCK ON, and SET QUALCOLS 2. This particular combination will result in the fewest contentions between users.

The following command line sets QUALCOLS to 2:

```
SET QUALCOLS 2
```

Notes:

- FASTLOCK and QUALCOLS can be set on at the same time.
- Setting STATICDB and FASTLOCK to ON (in that order), with QUALCOLS set to 2 will significantly increase multi-user performance with individual row changes.
- By setting QUALCOLS to "2", [ROWLOCKS](#) is no longer used to turn row locking ON or OFF.
- QUALCOLS is not the same as SET ROWLOCKS.

Example for Testing:

```
-- The UPDATE must alter the values for many rows
SET FEEDBACK ON
SET QUALCOLS 10 -- change to page locking
UPDATE <tablename> SET <columnname> = value -- no WHERE Clause
SET QUALCOLS 2 -- change to row locking
SET FEEDBACK OFF
CLS
```

See Also:

Environment Optimization

1.19.8.70 QUOTES

Special Character

Sets the character for quotation marks. This character is used around all text strings.

Syntax: SET QUOTES=NULL
SET QUOTES=char
(Use NULL to disable the special character.)

Default: '

1.19.8.71 RBADMIN

Operating Condition

Syntax: SET RBADMIN ON/OFF

Default: OFF

The RBADMIN setting is used for [RBAAdmin](#), the R:BASE Network Database Administrator utility.

In order for RBAAdmin to disconnect users from the database, this setting must be set to ON. All users, whether their setting for RBADMIN is ON or OFF will be seen within RBAAdmin. It is recommended that this setting be added to a database application startup file for ease of implementation with RBAAdmin. When RBADMIN is set ON and connections are made to the database, a hidden binary file will be created in the database directory. The name of the file is unique to each database; consisting of "RBAAdmin_" + dbname + ".bin". The binary file for the ConComp sample database, with RBADMIN set ON, would be "RBAAdmin_ConComp.bin".

1.19.8.72 RECYCLE

Operating Condition

Syntax: SET RECYCLE ON/OFF

Default: OFF

If RECYCLE is ON, when adding new rows require a new block from file 2, a new routine is called which searches for a suitable unused block rather than always adding a new block to the end of the file

The criteria for such a block are:

- No other table uses it
- The block is further down the file than the current last block of the table

PROS

If a suitable block is found, the block will be allocated to the table requiring the additional space and File 2 will not grow as a result of this allocation. The main benefit of using RECYCLE is that the growth of File 2 will be minimized. Over time this can add up to significant savings on disk space and backup media.

CONS

Since a new routine is being called to search for a suitable block, there will be a slight performance penalty. The penalty will only be incurred when an [INSERT](#) requires a new block.

Considerations

For RECYCLE to be effective, all users should have the setting ON. Do this in the RBENGINE9.CFG configuration file. RECYCLE will only have an impact when used in conjunction with [PACK TABLE](#). Dead space in File 2 must first be freed up before it can be reused. RECYCLE will not be of benefit if your database does not end up with lots of deleted rows over time, providing the opportunity to recover dead space.

Conclusions

Periodic use of [PACK TABLE](#) tablename in conjunction with RECYCLE ON will retard File 2 growth and reduce fragmentation. Use of [PACK INDEX](#) FOR tablename will keep the index statistics fresh and query optimization results maximized. The need for planned downtime will be reduced.

1.19.8.73 REFRESH

Operating Condition

Syntax: SET REFRESH value

Range: 10 to 65535 seconds

Default: 0

Mode: Multi-user

SET REFRESH specifies how often R:BASE refreshes the form or the Data Browser, and displays current data. It also automatically recalculates lower tables in forms. Specify zero to turn the setting off. When REFRESH is set off, R:BASE tells you of edits when you save or delete a row.

1.19.8.74 REVERSE

Operating Condition

Syntax: SET REVERSE ON/OFF

Default: ON

SET REVERSE ON displays data-entry fields in reverse video in forms. R:BASE stores the setting with the database.

1.19.8.75 ROWLOCKS

Operating Condition

Syntax: SET ROWLOCKS ON/OFF

Mode: Multi-user

Default: ON

R:BASE uses row-level locking in a multi-user environment. This command causes R:BASE to lock only the required row for the current command instead of locking the entire table. For example, if multiple users are modifying the same table using the [UPDATE](#) command, R:BASE locks only the rows affected by each UPDATE. When ROWLOCKS is set off, R:BASE sets table locks during each UPDATE, regardless of how many rows are affected.

See Also:

Environment Optimization

1.19.8.76 RULES

Operating Condition

Syntax: SET RULES ON/OFF

Default: ON

SET RULES determines whether R:BASE checks data against all existing rules during data entry and modification when you use the [EDIT](#), [EDIT USING](#), [ENTER](#), [INSERT](#), [LOAD](#), or [UPDATE](#) commands, or the [import/export](#) utility.

Set RULES off to direct R:BASE to ignore all rules when rules are not defined for a table, you are archiving data, or you are transferring data into another database. This speeds up processing because R:BASE normally checks the SYS_RULES table even if no rules are defined for a table. R:BASE does, however, check each entry against the data type of the column regardless of the RULES setting. If the database is protected by a database owner's user identifier, R:BASE does not accept the SET RULES command until you enter the owner's user identifier.

1.19.8.77 SCRATCH

Operating Condition

Syntax: SET SCRATCH ON / OFF / TMP / <path>

Default: ON

SET SCRATCH sets the drive and directory location for temporary files created when sorting data.

- SET SCRATCH ON to store temporary sort files on the database drive and directory.
- SET SCRATCH OFF to store temporary files on the current drive and directory.
- SET SCRATCH TMP to store temporary files in the Windows TEMP directory.
- SET SCRATCH <path> provides the path to the location where temporary files are stored.

You can use the SCRATCH command in the RBENGINE9.CFG file so that the setting is made prior to launching R:BASE.

About SCRATCH TMP

By default, R:BASE configuration files (RBEEngine.CFG, RBEEngine75.CFG, RBEEngine76.CFG, RBEEngine8.CFG, RBEEngine9.CFG, OTERRO.CFG, OTERRO35.CFG, OTERRO40.CFG, and OTERRO8.CFG) include the TMP for SCRATCH setting.

This default will allow R:BASE or Oterro sessions to use the user's TMP environment settings for SCRATCH files on startup and eliminate all issues related to setting the SCRATCH directory and related files.

To take advantage of this setting, use the option "SCRATCH TMP" in the appropriate R:BASE and OTERRO configuration files or use the "SET SCRATCH TMP" command in your application startup files.

1.19.8.78 SELMARGIN

Operating Condition

Syntax: SET SELMARGIN value

Range: 0 to the width of your screen

Default: 0

Use SELMARGIN to set the left margin for displaying the results of a [SELECT](#) command. The default for SELMARGIN is 0 (zero), which sets the margin to column 2. Use SELMARGIN when an ASCII file requires a predefined position or when data has a required starting point.

Note: Setting SELMARGIN to 0 (zero) or 2 begins the left margin at column two.

1.19.8.79 SEMI

Operating Condition

Syntax: SET SEMI ON/OFF

Default: OFF

Use SEMI to set the semicolon (;) key as the command terminator instead of the [Enter] key.

When SEMI is set on, you can enter multiple command lines without a continuation symbol. Also, the semicolon runs commands created for other SQL databases, such as SQL Server.

Note: When SEMI is set on, all commands, including [EXIT](#), must be followed with a semicolon.

The following command line sets SEMI off:

```
SET SEMI OFF;
```

1.19.8.80 SEMI (Special Character)

Special Character

Sets the character for the command separator.

Syntax: SET SEMI=NULL
SET SEMI=char
(Use NULL to disable the special character.)

Default: ;

1.19.8.81 SERVER

Operating Condition

Syntax: SET SERVER ON/OFF

Default: ON

SET SERVER sets messages from a foreign data source on or off. When SERVER is set on, messages from the foreign data source are displayed.

1.19.8.82 SHORTNAME

Operating Condition

Syntax: SET SHORTNAME ON/OFF

Default: OFF

Alters the display format of the directory contents, where the file names are listed.

With SHORTNAME set to ON, the [DIR](#) command lists the file name, extension, size in bytes, and the date and time files were last modified, only listing the contents in the traditional format.

In the following example, the DIR command could be used to review the Help files you have placed in your R:BASE program directory (your results may vary). The display uses SHORTNAME set to ON.

```
R>dir *.CHM
```

```
Volume in drive C is ACER
Directory of C:\RBT\RBG9\
```

```
APPLIC~1 CHM   2544918  5-14-08  12:02a Applications.chm
Codelock chm   149699  5-14-08  12:01a Codelock.chm
DATABR~1 CHM   655880  6-13-08  4:34p DataBrowseEdit.chm
DBEXPL~1 CHM   960100  5-13-08  11:57p DBExplorer.chm
EForms  chm    3602275  6-19-08  6:37a EForms.chm
Forms   chm    7860989  6-19-08  6:35a Forms.chm
Gateway chm    5099517  5-14-08  12:03a Gateway.chm
Labels  chm    698008  6-19-08  6:35a Labels.chm
PLUGIN~1 CHM   829028  12-07-07  6:02p PluginPowerPack.chm
QBE     chm    1159139  5-14-08  12:05a QBE.chm
RArchive chm    289839  2-19-08  4:27p RArchive.chm
RBDefine chm    470249  5-14-08  12:00a RBDefine.chm
RBEEdit chm    560594  6-13-08  4:34p RBEEdit.chm
RBG9    chm    5875651  6-19-08  4:44p RBG9.chm
RBZip   chm    618075  5-26-08  2:42p RBZip.chm
RDWFVI~1 CHM   54558  5-16-08  8:52a RDWFViewer.chm
Reports chm    956389  6-19-08  6:38a Reports.chm
RFTPCL~1 CHM   86373  11-06-07  9:53a RFTPClient.chm
RMail30 chm    207788  4-11-08  10:13a RMail30.chm
RPrompt chm    120572  5-14-08  12:05a RPrompt.chm
RSPELL~1 CHM   583616  11-20-07  3:51p RSpellChecker.chm
RZip    chm    178341  5-16-08  9:46a RZip.chm
Trace   chm    190983  5-30-08  4:28p Trace.chm
Tutorial chm   3498551  6-13-08  4:36p Tutorial.chm
      24 File(s)      37251132 bytes
      24 Dir(s)      54854406144 bytes free
```

1.19.8.83 SINGLE

Special Character

Sets the character for the single wildcard for R:BASE commands and clauses.

Syntax: SET SINGLE=NULL
SET SINGLE=char
(Use NULL to disable the special character.)

Default: _

1.19.8.84 SORT

Operating Condition

Syntax: SET SORT ON/OFF

Default: OFF

SET SORT sets the sort optimizer on or off. When set on, R:BASE sorts the minimal amount of data for large tables and recombines the sorted data with the unsorted rows using the minimum amount of disk space. Set SORT on when a sort fails.

SORT is only to be used when displaying a column, or columns, which are not indexed. Otherwise, no results will be displayed.

1.19.8.85 SORTMENU

Operating Condition

Syntax: SET SORTMENU ON/OFF

Default: ON

SORTMENU causes all data dictionary menus to be in alphabetical ascending order, including all pop-up menus that display tables, forms, views, labels, and reports. Menus with column names and values remain unsorted in their original order.

1.19.8.86 STATICDB

Operating Condition

Syntax: SET STATICDB ON/OFF

Mode: Multi-user

Default: OFF

Activates a read-only schema mode. A user who first connects to a database with STATICDB set to on engages that database to operate in a read-only schema mode whereby any user must have their STATICDB setting on in order to connect to that database.

The effect of having STATICDB set on is that no schema changes can occur during that connection session.

See Also:

Environment Optimization

1.19.8.87 TIME

Operating Condition

Syntax: SET TIME SEQUENCE HHMMSS (time sequence)
SET TIME FORMAT HH:MM:SS (time format)

Default: SET TIME SEQUENCE HHMMSS (time sequence)
SET TIME FORMAT HH:MM:SS (time format)

SET TIME sets the time entry sequence and output format, using up to 20 characters. Set TIME entry

sequence and display format in separate commands. Use H to specify hours, M for minutes, S for seconds, and .SSS for thousandths. TIME can be specified of up to thousandths of a second. R:BASE stores the setting with the database.

The keyword SEQ (sequence) sets the entry sequence such as HHMMSS while the keyword FOR (format) sets the display format. For example, the format HH:MM:SS can display 14:30:20. R:BASE displays midnight (24:00) as 0:00. You can also include AP to display time on a 12-hour clock. In the previous example, if you change the format to HH:MM:SS AP, R:BASE displays 2:30:20 PM. If the format contains spaces or commas, enclose the format in quotes. Enter the hours, minutes, and seconds in the order SEQ specifies.

Example: Valid Time Formats using two thirty and twenty seconds, p.m.

<u>Time Format</u>	<u>Displays</u>
HH:MM:SS	14:30:20
HH:MM:SS AP	2:30:20 PM
HH/MM/SS	14/30/20
HH-MM-SS AP	2-30-20 PM

TIME can affect time functions. For best results, first set TIME to the default HH:MM:SS and then use the time functions.

1.19.8.88 TIMEOUT

Operating Condition

Syntax: SET TIMEOUT value

Range: 0 to 1440

Default: 0

Use TIMEOUT to shut down an inactive R:BASE workstation and exit to Windows after a set amount of time passes. A countdown only begins when R:BASE is waiting for a keystroke, not while R:BASE is processing commands or while you are entering data. This is a useful feature for automatically disconnecting idle R:BASE sessions for scheduled database maintenance.

The default for TIMEOUT is 0 (zero), which does not activate a countdown. TIMEOUT is set in minutes (not seconds), and all workstations must set TIMEOUT separately.

When a TIMEOUT occurs, a command file can be run; however, the command file cannot include a [FILLIN](#), [DIALOG](#), or [PAUSE](#) command, or expect a keystroke. If you want to run a command file when a TIMEOUT occurs, you need to store the name of the file in a variable called **RBTI_TIMEOUT**.

For example:

```
SET VARIABLE RBTI_TIMEOUT TEXT = 'c:\CustDB\cleanup.rmd'
```

The following command line will exit a user to Windows after the user's workstation is inactive for one hour:

```
SET TIMEOUT 60
```

Notes:

- The RBTI_TIMEOUT command file must end with a [RETURN](#) command.
- The TIMEOUT command will close ANY and ALL open forms, designers, and editors WITHOUT saving the changes which have been made since the last save. It is the responsibility of the developer and end-user to implement proper coding and/or behavior to eliminate unexpected shutdowns without saving the changes. The TIMEOUT command will disconnect from the currently opened database (if applicable) before terminating the R:BASE session.

1.19.8.89 TOLERANCE

Operating Condition

Syntax: SET TOLERANCE value

Default: 0

SET TOLERANCE sets the tolerance for comparisons between numbers with REAL and DOUBLE data types. The default tolerance of 0 means that the match must be exact to six digits of accuracy for REAL numbers and to 15 digits of accuracy for DOUBLE numbers. R:BASE stores the setting with the database.

The following command specifies a tolerance of .1 when testing column values. If the tolerance is .1 and the [WHERE](#) clause specifies *colname* = 100, then values between 99.9 and 100.1 are valid. If you set the tolerance to one, the values between 99 and 101 are valid.

```
SET TOLERANCE .1
```

1.19.8.90 TRACE

Operating Condition

Syntax: SET TRACE ON/OFF

Default: OFF

SET TRACE ON will execute TRACE (Interactive Command File Debugger) inside a command file to trace a block of code as defined.

Example 01:

TRACE filename.ext (typical command line option)

Example 02: (in a command file)

your code here

SET TRACE ON (this will start the trace within a command file)

Your code here ...

SET TRACE OFF (this will stop the trace within a command file)

Your remaining code here ...

Notes:

- Once you turn OFF the TRACE in an R:BASE session, you will need to turn it back ON.
- (CVAL('TRACE')) will return the current status of TRACE (Values: ON or OFF)
- [SHOW TRACE](#) will display the current status of TRACE (Values: ON or OFF)
- Newly created RBENGINE9.CFG will also include the option for **TRACE ON**
- If you want no one to TRACE your code, setting the TRACE option to OFF at the beginning of your code or startup file will disable the TRACE command.

1.19.8.91 TRANSACT

Operating Condition

Syntax: SET TRANSACT ON/OFF
SET TRANS ON/OFF

Default: OFF

Mode: Transaction Processing

SET TRANSACT toggles transaction processing on and off. When transaction processing is set on and [AUTOCOMMIT](#) is set off, all commands entered after one [COMMIT](#) or [ROLLBACK](#) command until the next comprise a transaction. The commands in a transaction are executed as they are entered, but changes to the data and database structure are not made permanent until you enter COMMIT (or exit the

database). You can undo all changes in the transaction by entering ROLLBACK.

When transaction processing is on and AUTOCOMMIT is also on, each command that is executed successfully is treated as a transaction and made permanent. ROLLBACK has no effect when AUTOCOMMIT is on.

Only the first user to connect to a closed database can enter the TRANSACT setting for that database. Enter the command before connecting to the database. If anyone else already has the database open, R:BASE displays a message telling you that your TRANSACT setting must match that of the open database before you can connect. Transaction processing is either on for all users or off for all users in a given database.

1.19.8.92 USER

Use the SET USER command to run R:BASE with a user identifier and password, if one has been set up.

```
SET USER [userid [password]]
SET USER PASSWORD [password]
SET USER PASSWORD FOR userid TO password
```

Options

FOR userid

Specifies a user identifier.

password

Creates a new password. Enter NONE to remove an existing password.

PASSWORD

Specifies or changes the password for the current user identifier.

TO password

Creates a new password. Enter NONE to remove an existing password.

userid

Specifies a user identifier.

About the SET USER Command

Passwords are specific to user identifiers and databases and are not required by R:BASE; however, once a password is set up, R:BASE prompts for the user's password every time the user connects to the database or issues his/her user identifier. User identifiers have a maximum length of 36 characters. Passwords have a minimum length of three characters and maximum length of 36 characters.

To run R:BASE with your user identifier then connect to the database, enter the following command line:

```
SET USER <UserId>
```

You can also enter the following command line to run R:BASE with your user identifier:

```
SET USER
```

R:BASE displays a dialog box and prompts you for your user identifier.

When a password has been set up for a user identifier, R:BASE prompts for the password after the correct user identifier has been entered.

Note: When a user enters a user identifier or password in a dialog box, the user identifier is not

displayed on screen.

To add or change a password, connect to the database with the user identifier and enter the following command line:

```
SET USER PASSWORD
```

R:BASE prompts for the user's identifier, then prompts for the password. A user can cancel a password by entering NONE.

If the database owner is the current user, the database owner can assign him/herself a password using the SET USER PASSWORD command; however, if the database owner forgets the assigned password, the password cannot be found or changed.

As the database owner, to change a user's password, connect to the database and enter the following command line:

```
SET USER PASSWORD FOR <Userid> TO <Password>
```

Enter your current password when R:BASE prompts you for it, then when R:BASE prompts you for a new password, enter NONE.

Note: A user's password is revoked when the database owner revokes all the user's privileges.

1.19.8.93 VERIFY

Operating Condition

Syntax: SET VERIFY COLUMN/ROW

Default: COLUMN

Mode: Multi-user

SET VERIFY, used in the multi-user environment, specifies the level of concurrency control as a row or a column within a row.

SET VERIFY allows you to specify whether R:BASE concurrency control will apply to individual columns within a row or to all columns in the row. When the level of concurrency control is set to COLUMN, R:BASE checks only the columns you change while you are editing. When the level of concurrency control is set to ROW, if you change data in any column, R:BASE checks every column in the row.

R:BASE concurrency control operates automatically when you are using a form in multi-user environments. Concurrency control includes autorefresh and verification. When you refresh or try to save a row, R:BASE checks whether data has been changed by another user and alerts you if it has changed. This prevents simultaneous changes that could corrupt the integrity of your data. The SET VERIFY command affects the operation of both autorefresh and verification when you are using a form.

When concurrency control is set to COLUMN, R:BASE looks for conflicts, those instances when two users have both modified the same column. When R:BASE detects a conflict in at least one field:

- R:BASE displays all of the other user's changes in the appropriate fields.
- Where there is no conflict, changes you made continue to be displayed.
- R:BASE displays a message informing you that data has changed.

When concurrency control is set to ROW, R:BASE looks for a change to any column in the whole row, whether it is a conflict or not. When R:BASE detects a change:

- R:BASE displays all of the other user's changes in the appropriate fields.
- Where there is no conflict, changes you made to the data are discarded.
- R:BASE displays a message informing you that data has changed.

Whether concurrency control is set to COLUMN or ROW, you can review the changes and then continue editing the data in the form. After autorefresh, R:BASE prompts you to press any key to continue editing. After verification, you can either move on to your next task or edit the data again. If you choose to

move on when the level of concurrency control is set to COLUMN, you will be discarding any changes you made that are still displayed. R:BASE prompts you to press [Esc] if you want to move on, or to press [Enter] if you want to edit the displayed data.

When you edit data in a form, concurrency control is always enforced.

When you enter data in a form, concurrency control is enforced only when you are entering values in fields defined with a same-table look-up, or when you return to a row in a region that you had entered previously.

The first command line in the following example sets the level of concurrency control to check for changes in the entire row. The second command line starts an editing session using the form named *custform*.

```
SET VERIFY ROW
EDIT USING custform
```

1.19.8.94 WAIT

Operating Condition

Syntax: SET WAIT value

Range: 0 to 16383 seconds

Default: 4

Mode: Multi-user

SET WAIT sets the minimum number of seconds to retry the last requested resource (a table or database) before halting execution. Rather than aborting execution, SET WAIT allows you to set a length of time for R:BASE to keep trying to access a resource. A message is displayed showing the approximate percentage of wait time remaining.

If you do not run a SET WAIT command, R:BASE automatically retries the locked resource for approximately four seconds before issuing a retry message.

For commands that wait for resources, the precise period of the wait is at least as long as the time specified. On some computers, processing requirements may extend the length of the wait to longer than one second for each second designated.

When you enter a command from the R> Prompt and the waiting period expires, R:BASE displays a message prompting you to retry or ignore the command. When the command runs as part of a command file, however, and the waiting period expires, R:BASE ignores the command and goes on to the next command.

The following command tells R:BASE to retry the last requested resource for approximately 20 seconds.

```
SET WAIT 20
```

You can also adjust the interval in which R:BASE tries the command during the SET WAIT period.

For more information, see [INTERVAL](#).

1.19.8.95 WALKMENU

Operating Condition

Syntax: SET WALKMENU ON/OFF

Default: OFF

Allows shortcut access to menus

SET WALKMENU is a menu shortcut function allowing the user to access menu selections by typing the

beginning characters (up to when a match is made) of their names. Pressing any navigational keys (such as [Home] or [Page Up]) clears the buffer containing the keystrokes entered by the user while traversing the menu list. Any keystrokes not resulting in a match are not stored in the buffer, causing a beep.

Also see Walkmenu Time Interval

1.19.8.96 WHILEOPT

Operating Condition

Syntax: SET WHILEOPT ON/OFF

Default: ON

SET WHILEOPT improves the optimization and processing of [WHILE ...ENDWHILE](#) loops within applications by pre-compiling variables used within the WHILE loop. Follow these guidelines:

- Don't clear your WHILE variable(s).
- Don't define variables within your WHILE loop, only outside the loop; values can change within the loop.
- Adhere to the syntax rules for the [SWITCH](#) statement by making sure that the argument for the SWITCH statement is an expression.
- If you issue multiple [SET VARIABLE](#) commands on a single command line, those variables will not be optimized. If you want to increase the speed for that loop, put those SET VARIABLE commands on separate lines.

The WHILEOPT setting must be changed in a command file. The setting cannot be saved to the RBENGINE9.CFG file.

1.19.8.97 WIDTH

Operating Condition

Syntax: SET WIDTH value

Range: 40 to 5000 characters

Default: 79

SET WIDTH controls the width of a data line that R:BASE directs to the printer, screen, or file when using the [BACKUP](#), [COMPUTE](#), [CROSSTAB](#), [DISPLAY](#), [SELECT](#), [TYPE](#), [UNLOAD](#), or [WRITE](#) commands. Do not set the width to a number greater than the number of characters your printer can fit on a line; a typical page and computer screen display 80 characters. WIDTH does not affect report generation; each report defines the width of a data line.

Note: The 5000-character maximum does not apply to the TYPE or DISPLAY commands, which continue to have 256-character width limits.

1.19.8.98 WINBEEP

Operating Condition

Syntax: SET WINBEEP ON/OFF

Default: OFF

Allows R:BASE to access a subset of the Sound Events in Windows. WINBEEP command will use the current system sound schema.

When set to OFF R:BASE will use a standard Windows sound for all errors and the [BEEP](#) command. When set to ON certain sounds, as set in the Windows Control Panel, will be used instead depending on which sound type is used. Below is a table of Types and their corresponding sound event. You must set the actual sounds used in the Windows Sounds Control Panel Applet. You will also need to ensure that your speakers are Un-Muted and working properly. In either case, WINBEEP ON or WINBEEP OFF, if you have the "No Sounds" scheme selected in your Sounds Control Panel you will not hear any beeps from R:

BASE for Windows.

Sound Type	Sound Event
0	SYSTEM
1	SYSTEMSTART
2	SYSTEMEXIT
3	SYSTEMHAND
4	SYSTEMASTERISK
5	SYSTEMQUESTION
6	SYSTEMEXCLAMATION
7	SYSTEMWELCOME
8	SYSTEMDEFAULT

Example

```

SET CAPTION ' '
SET VAR VRows INTEGER = 0
SET VAR VMsg TEXT = NULL
SELECT COUNT(*) INTO VRows INDIC IVRows FROM TableName
IF VRows = 0 THEN
  CLS
  SET WINBEEP ON 2
  BEEP
  PAUSE 2 USING 'No Record(s) on File!' AT CENTER CENTER
  SET WINBEEP OFF
  GOTO Done
ELSE
  CLS
  SET WINBEEP ON 1
  BEEP
  SET VAR VMsg = ((CTXT(.VRows)) & 'Record(s) on File!')
  PAUSE 2 USING .VMsg AT CENTER CENTER
  SET WINBEEP OFF
  GOTO Done
ENDIF

LABEL Done
CLS
CLEAR ALL VAR

```

1.19.8.99 WRAP

Operating Condition

Syntax: SET WRAP ON/OFF

Default: ON

Text in fields that have NOTE and TEXT data types will wrap in forms and [FILLIN](#) windows, variables, and reports.

1.19.8.10 WRITECHK

Operating Condition

Syntax: SET WRITECHK ON/OFF

Default: OFF

SET WRITECHK ON tells R:BASE to verify every write to disk.

1.19.8.10 ZERO

Operating Condition

Syntax: SET ZERO ON/OFF

Default: OFF

SET ZERO allows a null to be treated as a zero in a mathematical expression involving INTEGER, NUMERIC, REAL, DOUBLE, CURRENCY, DATE, DATETIME, or TIME data types. With ZERO set on, R:BASE returns a negative number when you subtract an integer from a null. With ZERO set off, the same computation results in a null. R:BASE stores the setting with the database.

1.19.8.10 ZOOMEDIT

Operating Condition (**R:BASE for DOS only**)

Syntax: SET ZOOMEDIT ON/OFF

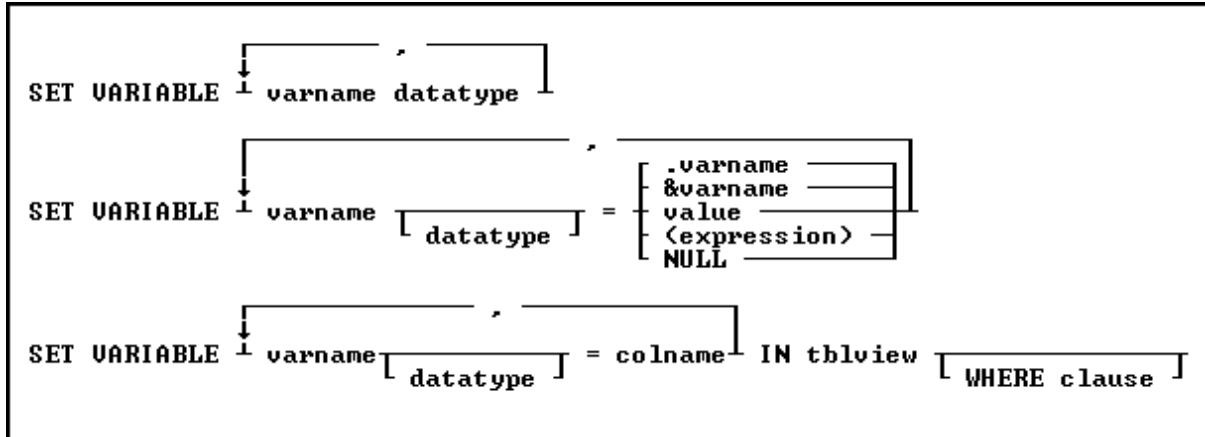
Default: OFF

Toggles the field expansion method.

Set ZOOMEDIT to on if you want to open REdit, the R:BASE text editor, when you zoom in on a NOTE filed in a form. When ZOOMEDIT is off, a dialog box that wraps text is opened instead.

1.19.9 SET VARIABLE

Use the SET VARIABLE command to define or redefine a variable value and/or data type.



Options

,
Indicates that this part of the command is repeatable.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter *#c*, where *#c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*).

datatype

Specifies an R:BASE data type for the variable. See Data Types.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

IN tblview

Specifies a table or view.

NULL

Sets the variable equal to NULL.

value

Sets the variable equal to a specified value. A value is a constant amount, text string, date, or time, or the value assigned to *varname*.

varname

Specifies a variable name, which must be unique among the variable names within the database. In R:BASE eXtreme 9.0 (32), the variable name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the variable name is limited to 128 characters.

&varname

Sets the first variable equal to the exact contents of a second variable; the ampersand tells R:BASE to evaluate the contents of the variable first.

For example, if *varname* is the text string (2+3), then *&varname* is the value 5.

.varname

Sets the first variable equal to the exact contents of a second variable.

For example, if *varname* is the text string (2+3), then *.varname* is (2+3).

WHERE clause

Limits rows of data. For more information, see the [WHERE](#) Clause.

About the SET VARIABLE Command

Variables identify a changeable value. R:BASE provides three kinds of variables: global, error, and system. The SET VARIABLE command defines global variables, which are temporary variables that exist within R:BASE, but are not part of any database. Global variables remain in memory until you clear them or exit from R:BASE. R:BASE sets error and system variables internally.

Global variables have several uses: they can provide a temporary value in a command, hold the result of a calculation, act as a counter, or capture keyboard entries for use with menus or screens. The most common method of defining variables is to assign the variable value with the SET VARIABLE command. For information about defining global variables, see Variables.

Variable names have the following restrictions:

- The variable name is not an R:BASE reserved word.
- The variable name begins with a letter, contains only letters, numbers, and the following special characters: #, \$, _ , and %.

It is good programming practice to always define the data type for the variable before assigning it a value, unless you are setting a variable to a column value or using the variable in the [CHOOSE](#) command.

When defining an variable as a text string, enclose the text string in single quote marks (or the current QUOTES character); otherwise, it might be interpreted as an arithmetic expression.

Assigning a Data Type to a Variable

The *datatype* option refers to one of the valid R:BASE data types: BIT, BITNOTE, CURRENCY, DATE, DATETIME, DOUBLE, INTEGER, LONG VARBIT, LONG VARCHAR, NOTE, NUMERIC, REAL, TIME, VARBIT, OR VARCHAR. You can define a variable to have a NOTE data type, but R:BASE treats it as TEXT for

most uses. You can also specify the precision and scale for NUMERIC data types.

The *datatype* option creates a variable, determines its data type, and sets its value to null. Use this option to define a variable's data type before assigning a value to the variable. To set multiple variables in the same command, separate the variables by a comma or the current delimiter.

For an existing variable, you can use the *datatype* option to change the data type, but it is recommended to use one of the conversion functions. If you change the data type, the new data type must be compatible with the current variable value; if the variable is not compatible, R:BASE displays an error message and leaves the value and data type unchanged. If you change a variable with a TEXT data type to a non-compatible data type, R:BASE changes the value to null.

Assigning a Value to a Variable

The *value* option is a data value or constant, such as 10, TOM, 3.1416, or \$17.23. If the variable already exists, any new value must have a data type that is compatible with that variable. If the variable does not exist, R:BASE defines the variable's data type.

You can also define the variable's data type in this command before assigning it a value.

Setting the Value of a Variable to Another Variable

When you set the variable to the value of another variable, the second variable must be a dot variable (.) or an ampersand (&) variable.

When you precede a variable with a dot (.), R:BASE uses the value stored in the variable as if it were a constant.

When you precede a variable with an ampersand (&), R:BASE first evaluates the value contained in the ampersand variable. For example, consider the following uses of the command:

```
SET VARIABLE v1 TEXT = '(A + B)'
SET VARIABLE v2 = .v1
SET VARIABLE v3 = &v1
```

When the first command line runs, variable *v1* will contain (A + B). When the second command line runs, variable *v2* will also contain (A + B) because the dot tells R:BASE to set the value as an exact match to the contents of variable *v1*. When the third command line runs, variable *v3* will contain AB (the concatenation of A and B) because the ampersand tells R:BASE to compute the contents of variable *v1*.

As shown in the example above, an ampersand variable can contain one command or part of one command, such as an expression. The first variable is set to the computed value of the ampersand variable. Below is an example:

1. SET VARIABLE v1 TEXT
2. SET VARIABLE v2 INTEGER
3. SET VARIABLE v1 = '((50 + 100)/ 2)'
4. SET VARIABLE v2 = &v1
5. SHOW VARIABLE

- Sets the data types for variables *v1* and *v2* to TEXT and INTEGER, respectively.
- Sets variable *v1* to a text value that is a valid arithmetic expression.
- Sets variable *v2* to &*v1*.
- Displays the value of all variables, including the system variables.

R:BASE computes the expression contained in *v1* and assigns the calculated value to *v2*. When R:BASE sees a variable name preceded by ampersand, it treats the contents of the variable as if it was entered from the keyboard. The [SHOW VARIABLE](#) display would like the following:

Variable	= Value	Type
#date	= 04/12/94	DATE
#time	= 22:52:52	TIME

<i>#pi</i>	= 3.14159265358979	DOUBLE
<i>sqlcode</i>	= 0	INTEGER
<i>v1</i>	= ((50 + 100) / 2)	TEXT
<i>v2</i>	= 75	INTEGER

Setting a Variable to an Expression

An (*expression*) can be either an arithmetic operation that combines two or more items in an arithmetic computation, or a string expression that concatenates two or more text items, or uses a TEXT function. The items can be values or the values contained in variables.

If you do not predefine the data type of a variable, the original data type of each item determines the data type of the result. For example, if you add a variable that has an INTEGER data type to a variable that has a REAL data type, the resulting variable has a REAL data type unless you define the result to be an INTEGER data type.

If any item in an arithmetic expression is null, the result will be null unless you specify [SET ZERO ON](#).

Assigning Column Values in a Table or View

If you specify a table or view in a SET VARIABLE command, you can include an optional WHERE clause to indicate which row to use. If you do not include the WHERE clause, R:BASE uses the value for the column in the first row.

You must have [SELECT privileges](#) on the table to use this form of SET VARIABLE.

In instances where your building a dynamic SET VARIABLE command based on previous options made, you must use an ampersand variable in place of a column or table name, for example:

```
CHOOSE vtab FROM #TABLES
CHOOSE vcol FROM #COLUMNS IN &vtab
SET VARIABLE vnewpr = &vcol IN &vtab
```

Enter the table and column names into the *vtab* and *vcol* variables first. You can do this by using the CHOOSE #TABLES and CHOOSE #COLUMNS commands, as shown in the above example. The [CHOOSE](#) command displays a menu of tables or columns from which to choose. By using ampersand variables to hold the table and column names, you can use the same SET VARIABLE command to get values from different columns in a table or from different tables. Each time SET VARIABLE requests a column, it retrieves information from the first row in the table stored in &vtab.

You can define multiple variables with a single SET VARIABLE command when you set the value of the variables to the value of columns in a table. However, when capturing column data into variables, it is better to use the SELECT command; specifically, [SELECT INTO](#). SELECT INTO is the SQL compliant command when capturing table data into variables.

Examples

The following table provides examples of the SET VARIABLE command.

SET VARIABLE Examples

Examples	Description
SET VARIABLE vtext TEXT	Defines the <i>vtext</i> variable to have a TEXT data type.
SET VARIABLE vreal REAL = 100.9	Defines <i>vreal</i> variable to have a REAL data type, and assigns it the value 100.9.
SET VARIABLE vnumer NUMERIC (9,3)	Defines the <i>vnumer</i> variable to have a NUMERIC data type having a precision of 9 and scale of 3.
SET VARIABLE vnum = 14322	Assigns the integer value 14322 to the <i>vnum</i> variable.
SET VARIABLE VTWO = .VNUM	Assigns the value of the <i>vnum</i> variable to the <i>vtwo</i> variable.
SET VARIABLE V3 = &V4	Assigns the computed value of <i>v4</i> to the <i>v3</i> variable.
SET VARIABLE vltdate = ('12/25/93' + 90)	Assigns the value 03/25/94 to the <i>vltdate</i> variable.

SET VARIABLE <i>vfulln</i> = (. <i>VFIRSTN</i> & . <i>VLASTN</i>)	Assigns to the <i>vfulln</i> variable the value of the full name formed by concatenating the values in the <i>vfirstn</i> and <i>vlastn</i> variables. The ampersand inserts a space between the two values.
SET VARIABLE <i>v1</i> = <i>col1</i> , <i>v2</i> = <i>col2</i> , <i>v3</i> = <i>col3</i> IN <i>tbl1</i> WHERE <i>col1</i> = 'Smith' OR SQL compliant variation: SELECT <i>col1</i> , <i>col2</i> , <i>col3</i> INTO <i>v1</i> INDI <i>iv1</i> , <i>v2</i> INDI <i>iv2</i> , <i>v3</i> INDI <i>iv3</i> FROM <i>tbl1</i> WHERE <i>col1</i> = 'Smith' See SELECT INTO	Assigns Smith to the variable <i>v1</i> ; the value of column <i>col2</i> to <i>v2</i> ; and the value of column <i>col3</i> in <i>tbl1</i> , from the row where <i>col1</i> contains Smith, to variable <i>v3</i> .

1.19.10 SETFOCUS

Use the SETFOCUS command to return focus to an MDI form from within an application.

SETFOCUS *alias*

Options

alias
Specifies a window name, which is the name given to the instance of a form started with the MDI option when using the AS *alias* option with the [EDIT USING](#) or [ENTER](#) command.

1.19.11 SHOW

Use the SHOW command to display the status of system parameters, format conditions, and default conditions.

SHOW {

CHARACTERS
keyword

 }

Options

CHARACTERS
Shows the current settings for the BLANK, DELIMIT, LINEEND, MANY, PLUS, QUOTES, SEMI, and SINGLE special characters.

keyword
Shows the current setting for the environment condition specified in the *keyword* option.

About the SHOW Command

Use any of the SET keywords in a SHOW command to display an environment setting. You can also display special characters and environment settings using **Utilities: Settings**.

The following commands are not displayed in a standard SHOW and must be requested manually.

- [AUTOCONVERT](#)
- [AUTODROP](#)
- [AUTORECOVER](#)
- [AUTOROWVER](#)
- [AUTOSYNC](#)
- [AUTOUPGRADE](#)
- [COMPATIBLE](#)
- [FASTFK](#)
- [FASTLOCK](#)
- [PAGEMODE](#)

- [PASSTHRO](#)
- [QUALCOLS](#)
- [SERVER](#)
- [WHILEOPT](#)

Examples

The following command displays all characters and settings.

SHOW

The following command displays settings for the special characters BLANK, DELIMIT, LINEEND, MANY, PLUS, QUOTES, SEMI, and SINGLE.

SHOW CHARACTERS

The following command displays the status of rules checking.

SHOW RULES

The following command displays the current user identifier.

SHOW USER

The following command displays the front end build of the product.

SHOW BUILD

R:BASE for Windows

What follows is the output from a SHOW command in R:BASE for Windows

```

BLANK
DELIMIT      /
LINEEND      p
SEMI         ;
QUOTES       '
PLUS         +
SINGLE        _
MANY         %
IDQUOTES     `

DATE format   MM/DD/YYYY
DATE sequence MMDDYY
Century threshold YEAR is 30
Default CENTURY is 19
TIME format   HH:MM:SS
TIME sequence HHMMSS
LINES/page    20
WIDTH/line    79
TOLERANCE           0.
NULL symbol    -0-
Lock WAIT time   2
Lock INTERVAL time (in .1 sec)  5
REFRESH TIME     0
MAXTRANS         10
EDIT verification level COLUMN
FILES            5
PRINTER
```


NOTE column PADding percentage 10

CURRENCY SYMBOL '\$'
 CURRENCY LOCATION PREF
 CURRENCY DIGITS 2
 CURRENCY CONVENTION B

(AND) ON Use SQL precedence for ANDs and ORs
 (ANSI) ON ANSI SQL name restrictions
 (AUTOCOMM) OFF AUTOMatically COMMIT after each command
 (AUTOSKIP) OFF AUTOMatically SKIP to the next field when editing
 (BELL) ON Sound the BELL on an error
 (CAPTION) ON Display caption on DIALOG, PAUSE
 (CASE) OFF Distinguish between UPPER and lower CASE
 (CLEAR) ON CLEAR data buffers after modifications
 (DEBUG) OFF Debug command is effective
 (ECHO) OFF ECHO input from command files
 (EQNULL) OFF Treat null EQUAL NULL as a match
 (EOFCHAR) ON Add EOF CHARACTER to the end of output files
 (ERROR) ON Display ERROR messages during processing
 (ESCAPE) ON ESCAPE allowed to abort processing
 (EXPLODE) OFF Exploding windows on dialog boxes
 (FEEDBACK) OFF Display FEEDBACK status during command processing
 (FIXED) OFF Delete header space for SELECT
 (HEADINGS) ON Display column HEADINGS on SELECT and TALLY commands
 (INSERT) ON Use INSERT key as a toggle
 (INTENSITY) ON Intensity field in form
 (LAYOUT) OFF Save the LAYOUT of data displayed on Browse/Edit
 (MANOPT) OFF Disable optimizer table reordering
 (OLDLINE) ON Use OLD style vertical LINE processing in reports
 (ONLINE) OFF Display only one line per row in select output
 (MDI) ON Default startup mode for forms will be MDI
 (MESSAGES) ON Display informational MESSAGES during processing
 (MIRROR) OFF Maintains MIRROR images of database files
 (MULTI) ON MULTI-user database access capability
 (POSFIXED) ON POSition DIALOG, PAUSE based on FIXED OEM font size
 (REVERSE) ON REVERSE video highlighting on data entry/edit
 (ROWLOCKS) ON ROW LOCK database access capability
 (RULES) ON Check data validation RULES
 (SCRATCH) ON SCRATCH files on same path as the database
 (SORT) OFF Optimize SORTing to minimize disk space used
 (SORTMENU) ON Sort menu items for selected menu types
 (STATICDB) OFF STATIC DataBase schema mode
 (TRANSACTION) OFF Connect in TRANSACTION mode
 (WALKMENU) OFF Search menus based on multiple keystrokes
 (WINBEEP) OFF Use selected WINDOWS BEEPs for beep sound
 (WRAP) ON Word WRAP text for display and edit
 (WRITECHK) OFF Verify each WRITE to disk
 (ZERO) OFF Treat nulls as ZERO in numeric calculations
 (ZOOMEDIT) ON Convert LINEEND to CR and back on ZOOMed EDIT fields

R:BASE for DOS

What follows is the output from a SHOW command in R:BASE for DOS

BLANK
 DELIMIT ,

LINEEND ^
 SEMI ;
 QUOTES '
 PLUS +
 SINGLE _
 MANY %
 IDQUOTES `

DATE format MM/DD/YYYY
 DATE sequence MMDDYY
 Century threshold YEAR is 30
 Default CENTURY is 19
 TIME format HH:MM:SS
 TIME sequence HHMMSS
 LINES/page 20
 WIDTH/line 79
 TOLERANCE 0.
 NULL symbol -0-
 Lock WAIT time 4
 Lock INTERVAL time (in .1 sec) 5
 REFRESH TIME 0
 MAXTRANS 201
 EDIT verification level COLUMN
 FILES 5
 PRINTER HPLASER1
 NOTE column PADding percentage 10
 CURRENCY SYMBOL '\$'
 CURRENCY LOCATION PREF
 CURRENCY DIGITS 2
 CURRENCY CONVENTION B

(AND) ON Use SQL precedence for ANDs and ORs
 (ANSI) ON ANSI SQL name restrictions
 (AUTOCOMM) OFF AUTOMatically COMMit after each command
 (AUTOSKIP) OFF AUTOMatically SKIP to the next field when editing
 (BELL) ON Sound the BELL on an error
 (CASE) OFF Distinguish between UPPER and lower CASE
 (CLEAR) ON CLEAR data buffers after modifications
 (DEBUG) OFF Debug command is effective
 (ECHO) OFF ECHO input from command files
 (EQNULL) OFF Treat null EQUAL NULL as a match
 (EOFCHAR) ON Add EOF CHARACTER to the end of output files
 (ERROR) ON Display ERROR messages during processing
 (ESCAPE) ON ESCAPE allowed to abort processing
 (EXPLODE) OFF Exploding windows on dialog boxes
 (FEEDBACK) OFF Display FEEDBACK status during command processing
 (HEADINGS) ON Display column HEADINGS on SELECT and TALLY commands
 (INSERT) ON Use INSERT key as a toggle
 (LAYOUT) OFF Save the LAYOUT of data displayed on Browse/Edit
 (MANOPT) OFF Disable optimizer table reordering
 (ONLINE) OFF Display only one line per row in select output
 (PROGRESS) OFF Display PROGRESS status during command processing
 (RADMIN) OFF Network database administration
 (REVERSE) ON REVERSE video highlighting on data entry/edit
 (ROWLOCKS) ON ROW LOCK database access capability
 (RULES) ON Check data validation RULES

```
(SCRATCH ) C:\DOCUME~1\Admini~1\LOCALS~1\Temp --SCRATCH files location
(SORT     ) OFF Optimize SORTing to minimize disk space used
(SORTMENU ) ON  Sort menu items for selected menu types
(STATICDB ) OFF STATIC DataBase schema mode
(TRACE    ) ON  Interactive command file debugger
(TRANSACTION ) OFF Connect in TRANSACTION mode
(WALKMENU ) OFF Search menus based on multiple keystrokes
(WINBEEP   ) OFF Use selected WINDOWS BEEPs for beep sound
(WRAP      ) ON  Word WRAP text for display and edit
(WRITECHK  ) OFF Verify each WRITE to disk
(ZERO      ) ON  Treat nulls as ZERO in numeric calculations
(ZOOMEDIT  ) ON  Use rBEDIT for ZOOM on notes in forms
```

1.19.12 SHOW BUILD

SHOW BUILD displays the build number of the R:BASE Front-End graphic user interface.

SHOW BUILD

The build can also be captured, using the (CVAL('BUILD')) Function, to store the build number information in a variable.

See also:

```
(CVAL('VERSION'))
(CVAL('VERSION BUILD'))
```

1.19.13 SHOW ERROR

Use the SHOW ERROR command without a variable name to display error-message status, which can be on or off.

SHOW ERROR [*varname*] **AT** *scrnrow,scrncol*

Options

AT scrnrow, scrncol

Defines the screen row and column at which to display the first character of the error message. Screen rows are between 1 and 25 and screen columns are between 1 and 80. (Supported in DOS platforms only)

varname

Specifies the name of the error variable, defined with the [SET ERROR VARIABLE](#) command. Alternatively, *varname* can be a global variable that captures the value of an error code from the defined error variable. If you enter the SHOW ERROR command without the *varname* option, R:BASE tells you whether error messages are on or off.

About the SHOW ERROR Command

Use SHOW ERROR with the name of an error variable to display the error message of the last R:BASE error that occurred.

The SHOW ERROR command with a variable name displays the error message of the error variable

captured by the SET ERROR VARIABLE command. The variable can be either a currently defined error variable (defined with the SET ERROR VARIABLE command) or a global variable used to capture the value of the error variable.

You can capture an error-variable value in a global variable so that any additional commands that are processed do not change the value of the variable. Once you have captured the error number in a global variable, you can use the SHOW ERROR command to display the error message by referencing the global variable name.

Examples

The following series of commands:

- Define an error variable named *verrvar*.
- Connect to the *dbname* database.
- Capture the value of the *verrvar* error variable in the *vglobal* global variable.
- Check the value of *vglobal* and, if it is not 0 (zero), display the error message on screen line 20, beginning in screen column 10.

```
SET ERROR VARIABLE verrvar
CONNECT dbname
SET VARIABLE vglobal = .verrvar
IF vglobal <> 0 THEN
  SHOW ERROR vglobal AT 20 10
ENDIF
```

Used without a variable name, the following command displays the error-message status.

```
SHOW ERROR
(ERROR) ON Display ERROR messages during processing
```

1.19.14 SHOW VARIABLE

Use the SHOW VARIABLE command to display the current values and data types of global variables at the R> Prompt.

```
SHOW VARIABLE [ varname [ =w,h ] [ AT scrnrow,scrncol ] ]
```

Options

AT scrnrow, scrncol

Defines the screen row and column where the first character of the value of the variable is displayed. Text values are left justified and numeric values are right justified within the display width specified by the =w option. Screen rows range from 1 to 25 and screen columns from 1 to 80. (Supported in DOS platforms only)

varname

Specifies a variable to be displayed. You can use wildcards to display variables that match a specific pattern.

=w,h

Specifies a maximum width and number of lines for the display of the variable. Long text variables are displayed on multiple lines. The =w,h option is limited to 30 characters and one line for these data types: CURRENCY, DATE, DATETIME, DOUBLE, INTEGER, NUMERIC, REAL, and TIME. When using the =w,h option for BIT, BITNOTE, LONG VARCHAR, LONG VARBIT, NOTE, TEXT, VARCHAR, and VARBIT variables, you can widen the display and set the total number of lines to display. If h is omitted, the value defaults to the number of lines needed to display the text for LONG VARCHAR, NOTE, TEXT, or VARCHAR data types, and only displays the file type for BIT, BITNOTE, LONG VARBIT, and VARBIT data types.

About the SHOW VARIABLE Command

Use SHOW VARIABLE to include variable values in screen displays in a procedure or command file. After displaying the variable values, R:BASE moves the cursor to the beginning of the next row.

You can also use SHOW VARIABLE with the [PAUSE](#) command to help debug command files by tracing the changing values of one or more variables.

Examples

The following command displays all global variables and the system variables *#date*, *#pi*, *#time*, and *sqlcode*.

```
SHOW VARIABLE
```

The following command displays the contents of the *vstatus* variable at screen row 10 and screen column 15.

```
SHOW VARIABLE vstatus AT 10 15
```

The following command displays the contents of the *vaddr2* variable starting at row 10 and in column 15. If *vaddr2* has a TEXT data type with a length of 30, R:BASE displays only the first 22 characters on line 10. R:BASE displays the remaining characters starting at row 11 in column 15. The setting for [SET WRAP](#) affects the display of variables.

```
SHOW VARIABLE vaddr2 =22 AT 10 15
```

The following command displays all global variables that begin with the characters "emp." You must use the wildcard characters that are defined by the settings for [SET SINGLE](#) and [SET MANY](#).

```
SHOW VARIABLE emp%
```

1.19.15 SHOW VERSION

SHOW VERSION displays version as well as the build number of the R:BASE Engine.

SHOW VERSION

This can also be used with CVAL to store the version as well as build number information in a variable.

The version can also be captured, using the (CVAL('VERSION')) Function, to store the build number information in a variable.

See also:

```
(CVAL('BUILD'))  
(CVAL('VERSION BUILD'))
```

1.19.16 SKIP

Use the SKIP command in an entry/exit procedure (EEP) to automatically skip fields in a form.

```
SKIP { n
      TO fldname
```

Options

n

Specifies the number of fields you want to skip, where *n* is a positive or negative integer. Use positive numbers to move forward through the field order in the form, and negative numbers to move backward. When you establish a value for *n*, count only those fields that allow you to enter or edit data.

SKIP 0 returns you to the same field you were in.

TO fldname

Specifies the field to which you want to skip, either forward or backward through the field order, where *fldname* is the column or variable name associated with a field.

About the SKIP Command

When an EEP is run, SKIP moves backward or forward through the fields in a form. SKIP operates within a form like the [Tab] key-it does not move to fields other than those in the current row or current table. You can use SKIP only in an EEP within a form.

This command is useful when you want to skip over certain fields in the form, depending on the data entered. A variable must always be associated with a field. You can set up an EEP to check the value entered in the field's variable, and then, depending on the value, skip to another field.

NOTES:

- SKIP TO Fldname or SKIP *n* when used in a Tab Form will ONLY be applicable to that Tab (i.e., page) of the Tab Control. Therefore, you CANNOT use SKIP command to switch from any field on one page of the Tab Control to another page of the Tab Control. A Tab (page) of the tab control is similar to a "container". Thus, the SKIP command will ONLY work with objects on the same page of the Tab Control.
- You may also use the [PROPERTY](#) command as such; **PROPERTY <ComponentID> SET_FOCUS 'TRUE'** to achieve the same functionality.

Examples

The following commands cause a form to skip three fields if the *vcashtype* variable contains any value.

```
IF vcashtype IS NOT NULL THEN
  SKIP 3
ENDIF
```

The following commands cause a form to skip to the field containing the *amount* column if the *vcashtype* variable contains any value.

```
IF vcashtype IS NOT NULL THEN
  SKIP TO amount
ENDIF
```

1.19.17 SSQL

Use the SSQL command to pass SQL commands through R:BASE to the foreign data source, where the commands will be interpreted and executed.

SSQL SQLcommand

Options

SQLcommand

Specifies the SQL command to send.

About the SSQL Command

SSQL requires R:BASE to already be connected to the data source. With a few exceptions, you can use every SQL command supported by the connected data source. You cannot use commands that usually return data, such as the [SELECT](#) command.

To retrieve data from a foreign table, attach the table and use standard R:BASE commands. SSQL can be used to send structure modification commands that alter the foreign database, which are subject to the access rights and restrictions placed on the user by the foreign database.

1.19.18 SUBTRACT

Use the SUBTRACT command to form a new table by subtracting rows in one table from a second table based on common columns.

```
SUBTRACT tblname1 FROM tblname2 FORMING tblname3 [ USING collist ]
```

Options

FORMING tblname3

Specifies the name of the table that is produced when you subtract one table from another.

FROM tblname2

Specifies the name of the table from which you want to subtract another table.

tblname1

Specifies the name of the table you want to subtract from a second table.

USING collist

Specifies the columns and order of columns that you want to include in the new table. At least one column in the USING clause must exist in both tables. If you do not include the clause, R:BASE makes a new table that includes all columns from both tables.

About the SUBTRACT Command

When one table (*tblname1*) is subtracted from another table (*tblname2*), R:BASE looks for columns in both tables that have the same name. The values in these common columns are subtracted and placed in the new table (*tblname3*). *Tblname3* also contains all columns from *tblname2* that are not in common with *tblname1*, unless columns are listed in the USING clause. It is worth noting that when there is more than one column with the same name R:BASE will not allow you to use the SUBTRACT Command without using the USING clause to specify which columns to compare.

Use SUBTRACT when you want to create a table that identifies differences between two tables, such as unique rows. When computed columns are common, R:BASE subtracts the current values in the

computed columns. If you include a computed column in the USING clause, R:BASE transfers the current value for the computed column to the new table. If you want a computed column in a new table, include each column used in the computed column's expression with a USING clause. If you do not, R:BASE changes the computed column to a regular column containing data.

In a new table, R:BASE does not update the value in a row for an autonumbered column. An autonumbered column becomes a regular column.

You can increase the speed of SUBTRACT by creating an index for a common column in *tblname1*. SUBTRACT does not use an index in *tblname2*. If both tables have multiple common columns, index the common column in *tblname1* that is most distinctive.

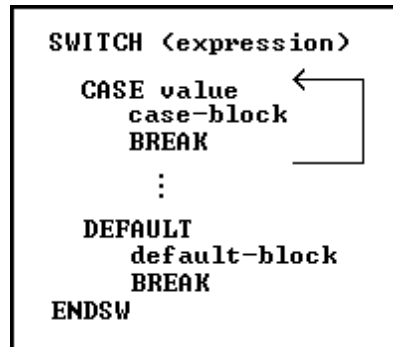
Example

The following command creates a new table named *nobonus* by subtracting the common columns in the *salesbonus* table from the *employee* table.

```
SUBTRACT salesbonus FROM employee FORMING nobonus
```

1.19.19 SWITCH/ENDSW

Use the SWITCH...ENDSW command in a program to define a block of possible actions to take depending on the value of an expression.



Options

BREAK

Ends SWITCH processing; use this option within each CASE comparison and in the DEFAULT block.

case-block

Contains one or more commands to execute if the CASE value matches the SWITCH expression.

CASE value

Compares the SWITCH value to another value. If the values match, the commands following CASE are executed; otherwise, the next CASE comparison is checked.

DEFAULT

Provides commands to execute if no CASE comparisons are true.

default-block

Contains one or more commands to execute if no CASE comparisons are true.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

About the SWITCH...ENDSW Command Structure

SWITCH...ENDSW structures are composed of a series of options. The syntax diagram above shows the

entire SWITCH...ENDSW structure, including the SWITCH value, CASE blocks, and the DEFAULT block.

The SWITCH Expression

SWITCH defines the expression to be compared. You can have multiple comparisons, so ENDSW defines the end of the comparisons. The SWITCH expression result must be either an INTEGER or a TEXT data type. The SWITCH expression can be a calculation, constant value, or variable. Any length of text can be compared, but only the first 30 characters are checked in each CASE block.

CASE Blocks

A CASE block consists of three parts: the CASE comparison, the commands following each comparison, and the BREAK statement.

CASE comparisons must be the same data type as the SWITCH expression result - either INTEGER or TEXT. A CASE value cannot be an expression, but must be a constant value or a variable. You can have multiple CASE comparisons to run a single set of commands. For an example of how to use multiple comparisons, see "Examples" below.

The commands following a CASE comparison can include any R:BASE command, including a nested SWITCH...ENDSW structure. You can nest as many SWITCH...ENDSW structures as memory allows.

Use a BREAK statement as the last command in a CASE block to exit from the SWITCH...ENDSW structure. The BREAK command stops R:BASE from checking any additional CASE comparisons.

The DEFAULT Block

You can have only one DEFAULT block for each SWITCH...ENDSW structure. A DEFAULT block provides a set of commands to run if none of the CASE comparisons is valid; make the DEFAULT block the last statement block in a SWITCH...ENDSW structure. If a CASE block follows a DEFAULT block, R:BASE generates a warning.

Example

The following SWITCH...ENDSW structure uses a date entered in a [FILLIN](#) command in the expression. The TDWK function calculates day of the week as text from the date stored in *vday*.

```
FILLIN vday USING 'Enter a date: '
SWITCH (TDWK(.vday))
  CASE 'Saturday'
  CASE 'Sunday'
    WRITE 'This is a weekend day.'
    SHOW VARIABLE vday
    BREAK
  DEFAULT
    WRITE 'This is a weekday.'
    SHOW VARIABLE vday
    BREAK
ENDSW
```

If you entered 12/17/94 when prompted for the date, the first CASE comparison would check whether the day of the week is the word *Saturday*. Because the word is *Saturday*, R:BASE would display the message below. The BREAK command prevents R:BASE from processing the rest of the commands in the SWITCH...ENDSW structure.

```
This is a weekend day.
12/17/94
```

If the date entered is not *Saturday* or *Sunday* - for example, 12/22/94 - the information in the DEFAULT block would display the following.

```
This is a weekday.
```

12/22/94

1.20 T

1.20.1 TALLY

Use the TALLY command to count the occurrence of identical values in a column.

TALLY colname FROM tblview [WHERE clause]

Options

colname

Specifies the name of the column from which you want to tally information.

FROM tblview

Specifies the table or view.

WHERE clause

Limits rows of data. See [WHERE](#).

About the TALLY Command

The TALLY command displays each value and its number of occurrences as a list.

You can also count identical values by choosing **Calculate: Tally** from the Data Browser.

TALLY does not work with a NOTE data type.

To display data without column headings, set [HEADINGS](#) off before using the TALLY command. Setting HEADINGS off is useful if you direct output of the TALLY command to a device (printer or file) other than the screen, for example, to a file that you intend to load into another database. The default setting for HEADINGS is on.

Example

The following command lists occurrences of each customer identification number in the *transmaster* table. R:BASE sorts the information in ascending order by *custid*.

```
TALLY custid FROM transmaster
```

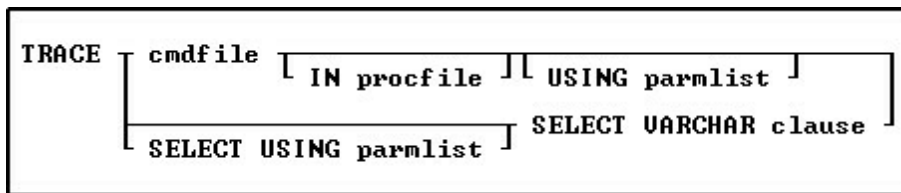
1.20.2 TILE

Use the TILE command to tile the open windows.

TILE

1.20.3 TRACE

Use the TRACE command to debug command blocks, command files, or command files requiring passed parameters. [Ctrl] + [Shift] + [F12] will trace the next entry/exit procedure.



Options

cmdfile

Specifies the name of the command file to debug.

On a workstation with multiple drives (local or mapped), especially when the files are on the different drive, it is always the best practice to define a drive letter when copying, deleting, renaming or running files, unless the specified files are located in the working directory. You will not need to specify the drive letter if all of the files are located in the default directory when using the copy, delete, rename or run commands.

IN procfile

Specifies the name of a procedure file. A procedure file is a compiled binary file that contains stored menu, screen, and command blocks. Include a drive and path name if the procedure file is not on the current drive and directory.

SELECT VARCHAR clause

Specifies a column defined with the VARCHAR data type from a table, then you can run the contents. The [SELECT](#) clause must limit the data to only one row; otherwise, an error is returned.

USING parmlist

Lists the values the command file will use when it executes. The parameter list can contain up to nine values. The first value in the list is referenced in the executed file as %1, the second as %2, and so on through %9. They are treated just like other variables. To reference the contents of these variables, preface the variable name with a dot (.); for example, set v1 = .%1.

About the TRACE Command

The TRACE command starts the R:BASE interactive debugger. It allows you to go through the command file line-by-line to facilitate debugging the code. The TRACE command must be on a line by itself and not combined with other commands. For TRACE to work correctly with a file encoded by CodeLock, the ASCII file must be present in the current drive and directory, and have the file extension of .APP. If TRACE cannot find the ASCII file, it does not go through the procedure file line-by-line when it runs.

New Features (Windows Only):

Immediately after you find error in the R:BASE debugger, you can correct bad syntax using the R:BASE Editor.

To correct an error :

1. You start R:BASE debugger using the TRACE command.
2. A error point is found, R:BASE debugger display a error message and Do you modify this file? message.
3. If you wish modify it at once, you click Yes button. The R:BASE Editor is started and you move the cursor to the error point.
4. You finish to correct error, save the file and exit the R:BASE Editor.
5. If you want to repeat debugging, you restart the R:BASE debugger using trace command.

For more information about the Interactive Debugger, refer to the TRACE compiled Help in the R:BASE program directory.

Examples

The following command displays the MYCMDFIL file for debugging in the current directory.

TRACE mycmdfil

The following command displays a command block for debugging named *mycmdfil* in a procedure file named MYPROCFL.APX.

TRACE mycmdfil IN myprocfl.apx

The following TRACE command displays the Trace Debugger for the command syntax with the *CmdData* VARCHAR data field inside the table *IntrnlCmd*.

TRACE SELECT CmdData FROM IntrnlCmd WHERE CmdName = 'NewKey'

TRACE can also be initiated inside a command file to trace a block of code using the [SET TRACE ON](#) operating condition.

1.20.4 TURBO

Use the TURBO command to convert your database files to R:BASE eXtreme database files.

TURBO dbname [**V9**] [**IDENTIFIED BY** [**OWNER**]]

Options

dbname

Specifies a database name

V9

Specifies that you are converting a R:BASE Turbo V-8 database to R:BASE eXtreme 9.0.

IDENTIFIED BY

Specifies the user identifier. If left blank, R:BASE prompts you for the user identifier. R:BASE does not display it as you enter the text.

OWNER

Optional; specifies the database owner name. If omitted and an OWNER name exists, you will be prompted.

Notes:

- If you are converting a RB1-RB4 database file structure to RX1-RX4 for the first time, you do not need to specify the "V9" parameter.
- The TURBO command will convert database files as far back as version 4.5. But, if you need to salvage any DOS forms and reports, you must connect to R:BASE 6.5++ for Windows to convert the forms and reports from DOS to Windows.

For more on database conversions, please refer to the R:BASE eXtreme Database Conversion Guide (PDF document) located within the R:BASE program directory).

1.20.5 TYPE

Use the TYPE command to send an ASCII text file to another file, screen, or printer.

TYPE filespec

Option

filespec

Specifies a file name with a drive and path specification in the form D:\PATHNAME/FILENAME.EXT.

About the TYPE Command

This command is similar to the operating system TYPE command. Unlike the operating system command, the R:BASE command displays a screen of data and prompts you to press any key to continue.

In R:BASE, you can use TYPE to display a file and review its contents. If TYPE is used in an R:BASE command file, it can display any ASCII file.

The [SET WIDTH](#) and [SET LINES](#) conditions effect how TYPE displays data. For example, if you set width to 40 characters and lines to 15, TYPE displays 15 lines of data before the "Press any key to continue" message displays, and wraps each line of data after 40 characters.

Use TYPE to check an ASCII file for "nondisplayable" characters before transferring data from an external source to R:BASE. If your computer displays irrelevant graphic characters, the file contains nondisplayable characters that R:BASE might not be able to read.

To send a file to a printer, enter [OUTPUT PRINTER](#) before the TYPE command.

Examples

The following command displays the ASCII file named THISFILE.DAT stored on the current drive and directory.

```
TYPE thisfile.dat
```

The following command displays the ASCII file named MYFILE.DAT stored in the MYDIR directory on drive B:.

```
TYPE b:\mydir\myfile.dat
```

The following command prints the ASCII file named THISFILE stored on drive A:, then returns output to the screen.

```
OUTPUT PRINTER
TYPE a:thisfile
OUTPUT SCREEN
```

1.21 U

1.21.1 UNION

Use the UNION command to combine the columns and rows of two tables into a new table.

```
UNION tblname1 WITH tblname2 FORMING tblname3 [ USING collist ]
```

Options

FORMING tblname3

Specifies the name of the new table that is produced.

tblname1

Specifies the name of one of two tables you will combine.

USING collist

Specifies the columns and order of columns that you want to include in the new table. At least one column in the USING clause must exist in both tables. If you do not include the clause, R:BASE makes a new table that includes all columns from both tables.

WITH tblname2

Specifies the name of the second of two tables you will combine.

About the UNION Command

When two tables are added together, R:BASE looks for columns in both tables that have the same name. The values in these common columns are added together and placed in the new table.

If the common columns have more than one match, R:BASE adds a new row for each possible combination. For example, if a table with a column named *empid* has two rows with an identification number of 100, and a second table has three rows with the same identification number, the new table will have six rows—one for each combination.

You can increase the speed of UNION by creating an index for the common column in both tables. If the tables have multiple common columns, index the common column that has the greatest number of unique values.

Combining Various Column Types

When computed columns are common columns, R:BASE compares the current values and changes the computed columns to regular columns in the new table. If you want a computed column in the new table, you should include each column used in the computed column's expression before the computed column.

If you use the UNION command to combine one or more columns that have NOTE data types, R:BASE truncates the columns that have the NOTE data types until the total row length is within the 32,768-character limit.

In a new table, R:BASE does not update the value in a row for an autonumbered column. An autonumber column becomes a regular column.

Creating New Columns

You can also use UNION to add new columns to a table. Define the new columns in a temporary table with the [CREATE TABLE](#) command. Then, combine the existing table with the temporary table to form a new table containing all the columns. When the union is complete, use the [DROP](#) command to delete the temporary table. The temporary table must have at least one column in common with the existing table.

Examples

The following command creates the *saleshist* table by combining all rows for the *empid*, *empfname*, *emplname*, and *netamount* columns. The *saleshist* table contains the columns listed in the USING clause in the order given.

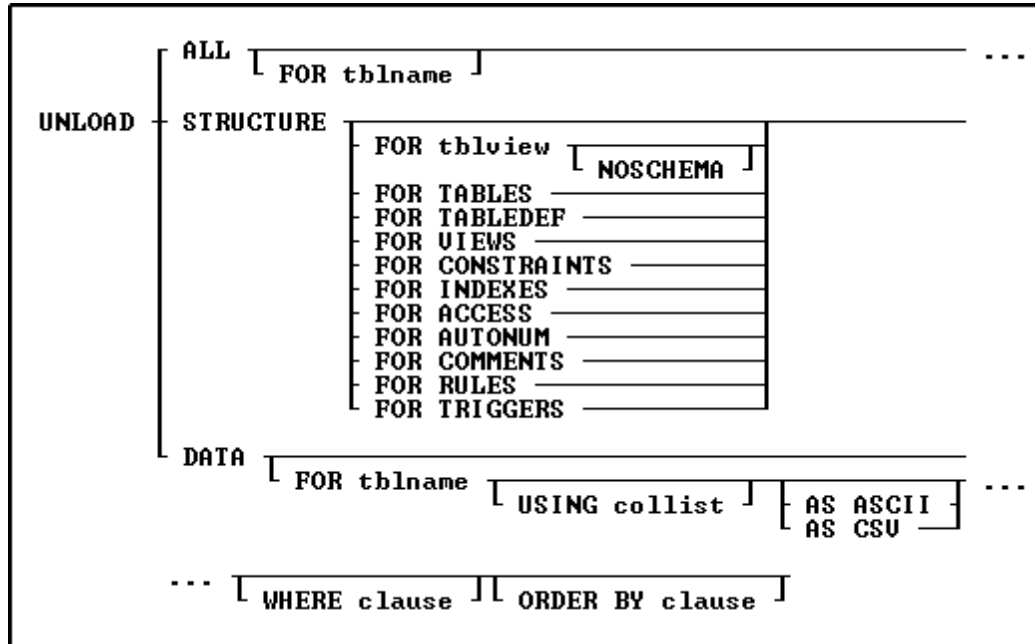
```
UNION employee WITH transmaster FORMING saleshist USING +
empid, empfname, emplname, netamount
```

The following command creates the *reporthistory* table by combining all columns and rows from the *employee* and *salesbonus* tables. If a USING clause is not specified, *reporthistory* includes all the columns from the *employee* and *salesbonus* tables, listing the columns from the *employee* table first.

```
UNION employee WITH salesbonus FORMING reporthistory
```

1.21.2 UNLOAD

Use the UNLOAD command to copy the data, structure, or data and structure of a database or table to a specified output device.



Options

ALL

Unloads both the data and the structure.

AS ASCII

Unloads data in ASCII-delimited format. Use only with the UNLOAD DATA command.

AS CSV

Unloads data in a minimally quoted comma separated format. Each field will be separated by the current DELIMIT character (usually the comma). Fields that contain the current DELIMIT character will be enclosed in the current QUOTES character.

DATA

Unloads just the data.

FOR ACCESS

Unloads all current access rights in the database.

FOR AUTONUM

Unloads the [AUTONUM](#) column formula for all autonumbered database columns in the database.

FOR COMMENTS

Unloads all comments assigned to tables, views, and columns in the database.

FOR CONSTRAINTS

Unloads all primary key, foreign key, and unique key relational constraints in the database, based on their order of creation.

FOR INDEXES

Unloads all indexes in the database.

FOR TABLES

Unloads the complete table structure for all tables in the database. The output includes the table

definition, relational constraints; based on their order of creation, access rights, autonumbered columns, comments, rules, and triggers.

FOR TABLEDEF

Unloads the table definition structure for all tables in the database. The output contains only the SQL command necessary to create all the tables.

FOR tblview

Specifies a single table/view to unload the SQL command necessary to create a specific table/view.

FOR TRIGGERS

Unloads all triggers in the database.

FOR RULES

Unloads all rules in the database.

FOR VIEWS

Unloads the SQL command necessary to create all views.

NOSCHEMA

Omits the character settings and [CREATE SCHEMA](#) syntax from the beginning of the output.

ORDER BY clause

Sorts rows of data. For more information, see [ORDER BY](#).

STRUCTURE

Unloads just the structure.

STRUCTURE FOR tblview

Unloads the SQL command necessary to create a specific table/view.

tblname

Specifies the table name to unload the data, structure, or both.

USING collist

Specifies the column(s) to use with the command.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

About the UNLOAD Command

Use UNLOAD to transfer tables or views from one database to another, or to back up a database. You can also use UNLOAD to free up space while using a temporary table.

The UNLOAD ALL and UNLOAD STRUCTURE commands require the database owner's user identifier if the database has had access rights granted with the [GRANT](#) command. R:BASE places the owner's user identifier and all the granted access rights in the file created by UNLOAD to ensure that the restored database file continues to be protected. Be sure to protect this file from unauthorized users.

The UNLOAD command creates a file with a .LOB extension for binary large objects, and the originating file that you specify for the data and/or structure. Your originating file can NOT have a .LOB file extension, otherwise, R:BASE will not be able to continue with the UNLOAD process.

Transferring Tables and Views

UNLOAD does not change the data or structure of the original database, and it does not unload computed column values unless you use the AS ASCII or AS CSV option.

You can also unload and transfer the data and structure of a view. To input the data from an unloaded view into a table, create a table to match the view's structure and use the [LOAD tblname FROM filespec](#) command, since views do not contain data. The UNLOAD command is useful when you want to create a file to transfer data to another database as a table.

To be able to reliably load data back into R:BASE from an unloaded file:

- [SET NULL](#) to -0- (the R:BASE default) before unloading data.
- Do not set a special character to the same setting as another special character.
- Precede UNLOAD with an [OUTPUT](#) command to direct the data to be unloaded to a file. You can edit the unloaded file with any ASCII text editor.

To transfer unloaded information back into R:BASE, use the [RUN](#), [RESTORE](#), or [GATEWAY](#) commands, if transferring only data. For example, you can use the UNLOAD DATA command to unload data, then the RUN command to transfer the data to a different database.

If the UNLOAD AS CSV syntax has been used you can use the [LOAD AS CSV](#) command to restore the data.

Backing up a Database

R:BASE unloads data in ASCII delimited format: values are separated by the current delimiter and all text strings are enclosed in quotation marks. UNLOAD creates a file containing commands that set special characters, such as commas and quotation marks. The setting of the [SET WIDTH](#) condition effects the width of data lines in the unloaded file

If the database has columns defined as binary or text large objects, then UNLOAD creates two files, one file containing the R:BASE commands, and a second file (with a .LOB extension) containing the large object data. Both files are needed to transfer the information back into R:BASE. Your originating file can NOT have a .LOB file extension, otherwise, R:BASE will not be able to continue with the UNLOAD process.

Note: The unloaded file cannot span multiple floppy disks.

The UNLOAD STRUCTURE or UNLOAD ALL commands write all the commands necessary to define the database or table, starting with [CREATE SCHEMA](#) AUTHORIZATION *dbname* near the beginning of the file. Before you input or restore the data or structure into a different database, you can use a text editor to change the database name in the unloaded file. If you use UNLOAD STRUCTURE, you can copy the table structure after you change the database name in the unloaded file.

Unloading Temporary Tables

Use the UNLOAD *tblname* command to backup individual temporary tables created when [STATICDB](#) is set on-which activates a read-only schema mode. When UNLOAD is used to backup temporary tables, it generates a SET STATICDB OFF command to be executed prior to the CREATE SCHEMA command.

Note:

- UNLOAD ALL does not back up temporary tables.

Examples

Example 01:

The following command lines unload only the data from the *product* table to a file named MYFILE.DBS. The data is in ASCII delimited format. The [OUTPUT SCREEN](#) command redirects the output back to the screen and closes the file.

```
OUTPUT myfile.dbs
UNLOAD DATA FOR product AS ASCII
OUTPUT SCREEN
```

Example 02:

In the example below the a file will be created that contains Comma Separated Values with no headings and no page breaks.

```
SET HEADINGS OFF
SET LINES 0
SET WIDTH 200
```

```
OUTPUT myfile.csv
UNLOAD DATA FOR Employees AS CSV
OUTPUT SCREEN
```

The commands above might create the file below. Notice that Jane Dough has Quotes surrounding her address. This is because the text contains an embedded comma.

```
Robert,Smith,123 Main St,Denver,CO,Support
Jane,Dough,'98 Folk St, Apt 1',Pittsburgh,PA,Sales
Matt,Follows,14 Arrowhead Ln,Portsmouth,RI,Services
```

Example 03:

The following command unloads the complete structure for the Contact table within the *ConComp* sample database.

```
R>UNLOAD STRUCTURE FOR contact
```

```
SET QUOTES=NULL
SET QUOTES='
SET DELIMIT=NULL
SET DELIMIT=','
SET LINEEND=NULL
SET LINEEND='p'
SET SEMI=NULL
SET SEMI=';'
SET PLUS=NULL
SET PLUS='+
SET SINGLE=NULL
SET SINGLE='_ '
SET MANY=NULL
SET MANY='%
SET IDQUOTES=NULL
SET IDQUOTES='^'
SET CURRENCY '$' PREF 2 B
DISCONNECT
SET STATICDB OFF
SET ROWLOCKS ON
SET FASTLOCK OFF
CREATE SCHEMA AUTHOR ConComp NONE
CREATE TABLE `Contact` +
(`CustID` INTEGER , +
`ContFName` TEXT (10) NOT NULL +
('Value for column Contfname cannot be NULL.'). , +
`ContLName` TEXT (16) NOT NULL +
('Value for column Contlname cannot be NULL.'). , +
`ContPhone` TEXT (12), +
`ContInfo` NOTE , +
`ContPhoto` LONG VARBIT )
ALTER TABLE `Contact` ADD FOREIGN KEY +
( `CustID` )+
REFERENCES `Customer`
COMMENT ON `CustID` IN `Contact` IS +
'Customer identification number'
COMMENT ON `ContFName` IN `Contact` IS +
```

```
'Customer contact first name'
COMMENT ON `ContLName` IN `Contact` IS +
'Customer contact last name'
COMMENT ON `ContPhone` IN `Contact` IS +
'Customer contact phone number'
COMMENT ON `ContInfo` IN `Contact` IS +
'Customer contact notes'
COMMENT ON `ContPhoto` IN `Contact` IS +
'Customer contact photo'
COMMENT ON TABLE `Contact` IS +
'Customer Contact Information'
```

Example 04:

The following command unloads the structure for the Contact table within the *ConComp* sample database, without any database schema at the beginning.

R>UNLOAD STRUCTURE FOR contact NOSCHEMA

```
CREATE TABLE `Contact` +
(`CustID` INTEGER , +
`ContFName` TEXT (10) NOT NULL +
(Value for column Contfname cannot be NULL.), +
`ContLName` TEXT (16) NOT NULL +
(Value for column Contlname cannot be NULL.), +
`ContPhone` TEXT (12), +
`ContInfo` NOTE , +
`ContPhoto` LONG VARBIT )
ALTER TABLE `Contact` ADD FOREIGN KEY +
( `CustID` )+
REFERENCES `Customer`
COMMENT ON `CustID` IN `Contact` IS +
'Customer identification number'
COMMENT ON `ContFName` IN `Contact` IS +
'Customer contact first name'
COMMENT ON `ContLName` IN `Contact` IS +
'Customer contact last name'
COMMENT ON `ContPhone` IN `Contact` IS +
'Customer contact phone number'
COMMENT ON `ContInfo` IN `Contact` IS +
'Customer contact notes'
COMMENT ON `ContPhoto` IN `Contact` IS +
'Customer contact photo'
COMMENT ON TABLE `Contact` IS +
'Customer Contact Information'
```

Example 05:

The following command unloads the indexes for the *ConComp* sample database.

R>UNLOAD STRUCTURE FOR INDEXES

```
CREATE INDEX CustState ON `Customer` +
(`CustState` ASC SIZE 2 )
```

Example 06:

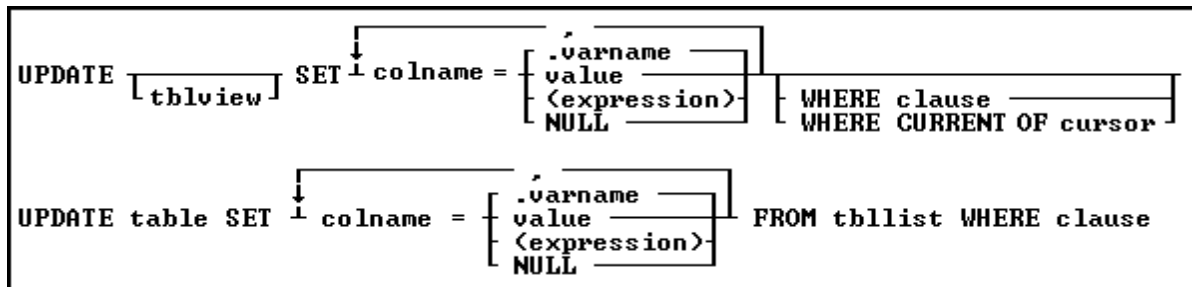
The following command unloads the rules for the *ConComp* sample database.

```
R>UNLOAD STRUCTURE FOR RULES
```

```
RULES 'Value for onhand cannot be less than minimum.' +
      FOR `ProdLocation` SUCCEEDS +
      WHERE ProdLocation.Onhand >= 1
```

1.21.3 UPDATE

Use the UPDATE command to change the data in one or more columns in a table or a view.



Options

,
Indicates that this part of the command is repeatable.

(expression)

Determines a value using a text or arithmetic formula. The expression can include other columns from the table, constant values, functions, or system variables such as *#date*, *#time*, and *#pi*.

FROM tbllist

Specifies a list of tables from which data can be retrieved and updated.

NULL

Sets the values in the column equal to null.

SET colname

Specifies the column to update.

table

Specifies a table.

tblview

Specifies a table or view. If no table or view name is included, columns will be updated in all tables containing the specified columns, according to the conditions of the WHERE clause.

value

Specifies a value to enter in the specified column.

.varname

Specifies a global variable that provides a value for a column.

WHERE clause

Limits rows of data. For more information, see [WHERE](#).

WHERE CURRENT OF cursor

Specifies a cursor that refers to a specific row to be affected by the UPDATE command. With this option, you must specify *tblview*.

About the UPDATE Command

This command is useful for adjusting values in columns that require uniform changes.

The UPDATE command only modifies data in columns in one table or view. You can also update a table by referencing values from another table. You can modify a column's value by doing the following:

- Entering a new value for the column as a constant or variable
- Entering an expression that calculates a value for the column
- Entering a null value

Notes:

- Only users that have been granted rights to update the table(s) or column(s) can run the UPDATE command.
- R:BASE complies with defined rules, even for columns not affected by the update. If an update breaks a rule, the update is not processed.
- You cannot use UPDATE with computed or autonumbered columns. To change a computed column value, change the values in the columns to which the computed column refers.
- The UPDATE command will not update data in a multi-table View (a View based on multiple tables), as the data is not editable.
- A View with a [GROUP BY](#) parameter is also not editable.

Updating Column Values

You can update a column with a specific value. The value you use must meet the requirements of the column's data type, for example, a numeric column cannot be loaded with a text value.

Use the current delimiter character (the default is a comma) to separate each column and its new value from the next column and value.

Use the following guidelines when modifying data with UPDATE:

- Do not embed commas within entries for CURRENCY, DATE, DATETIME, DOUBLE, INTEGER, NUMERIC, or REAL data types. R:BASE automatically inserts commas and the current currency symbol.
- When values for CURRENCY, DOUBLE, NUMERIC, or REAL or data types are decimal fractions, you must enter the decimal point. When values are whole numbers, R:BASE adds a decimal point for you at the end of the number. R:BASE adds zeros for subunits in whole currency values. For example, using the default currency format, R:BASE loads an entry of 1000 as \$1,000.00.
- When values for NOTE or TEXT data types contain commas, you can either enclose the entries within quotes, or use [SET DELIMIT](#) to change the default delimiter (comma) to another character.
- When values for NOTE or TEXT data types contain single quotes ('), and you are using the default QUOTES character ('), use two single quotes (') in the text string. For example, 'Walter Finnegan's order.'
- When values for NOTE or TEXT data types exceed the maximum length of a column, R:BASE truncates the value and adds it to the table. A message is displayed that tells you which row has been truncated.

Using an Expression or Variable

Enclose expressions in parentheses. If you use global variables in an expression, dot the variable (*varname*). If expressions contain values that have a TEXT data type, enclose the values within quotes. The default QUOTES character is the single quote (').

If you attempt to use a null value in an expression or computed column, the result of the expression is null. However, if you set [ZERO](#) to on, R:BASE treats null values as zeros and processes expressions as if the null value were zero.

Using the WHERE Clause

If an UPDATE command includes a table or view name, you do not need to specify a [WHERE](#) or WHERE CURRENT OF clause. All rows will be updated. If you use a [WHERE CURRENT OF](#) clause, you must include

a table or view name in the command.

If you omit a table or view name, you must use a WHERE clause with the UPDATE command so that you do not change values in more rows than you intended to change. The WHERE clause pinpoints the rows you want to change. If any columns exist in more than one table, all occurrences are changed if the column value meets the WHERE clause conditions. Test the WHERE clause by using the [SELECT](#) command before using the clause with UPDATE command. By using a WHERE clause with a SELECT command, you can view the rows you want to change before changing them.

R:BASE takes significantly less time to process a WHERE clause if one of the columns specified in the clause is an indexed column.

Using UPDATE with Transaction Processing

If more than one person at a time executes an UPDATE command and transaction processing is on, R:BASE might not execute the command concurrently. If you hold an UPDATE lock, you can read, modify, or delete any row in a table. R:BASE blocks any additional requests for UPDATE until other SELECT or UPDATE locks are cleared.

Examples

The following command changes values in the *company* and *custphone* columns of the *customer* table for the row where *custid* equals 100.

```
UPDATE customer SET company = 'Quality Computers', +
custphone = '617-341-3762' WHERE custid = 100
```

The following command changes the *invoicetotal* column in the *transmaster* table to the value of the expression (*invoicetotal* * .9) for rows where *transid* is greater than 5000.

```
UPDATE transmaster SET invoicetotal = ( invoicetotal * .9) +
WHERE transid > 5000
```

The following command changes the *listprice* column to the value of the expression (*1.1* * *listprice*) for every row in the *prodlocation* table containing an entry in the *listprice* column.

```
UPDATE prodlocation SET listprice = (1.1 * listprice) +
WHERE listprice IS NOT NULL
```

The following command adds to the set of conditions in the above command. The command below extracts all of the selling prices from the *transdetail* table and requires that *listprice* be changed only if it matches a current selling price in the table.

```
UPDATE product SET listprice = (1.1 * listprice) +
WHERE listprice IS NOT NULL AND model = 'CX3000' +
AND listprice IN (SELECT price FROM transdetail +
WHERE model = 'CX3000')
```

The following command changes the *onhand* column in the *prodlocation* table (specified by cursor *curs1*) to the value of the expression (*onhand* - 100). The changes are made only in the row currently referenced by the cursor.

```
UPDATE prodlocation SET onhand = (onhand - 100) +
WHERE CURRENT OF curs1
```

The following example shows interactive data updating in an application file. The first line defines the global variable *var1* as an INTEGER data type. The second command line allows you to enter a value for *var1*. The value of *var1* is used in the expression that is assigned to the *onhand* column of the *prodlocation* table. The UPDATE command changes values in *onhand* to the value of the expression (*onhand* - .*var1*) for all rows containing model numbers that begin with the letter C. The wildcard character % indicates one or more additional characters.

```
SET VARIABLE var1 INTEGER
```

```
FILLIN var1 USING 'Enter quantity by which to reduce inventory: '
```

```
UPDATE prodlocation SET onhand = (onhand - .var1) +  
WHERE model LIKE 'C%'
```

The following command changes the last names of two employees. This command omits the table name, thereby causing a global change to all tables that meet the WHERE clause criteria.

```
UPDATE SET emplname TO 'Smith-Simpson' WHERE +  
(empfname = 'Mary' AND emplname = 'Simpson') OR +  
(empfname = 'John' AND emplname = 'Smith')
```

The following example corrects a problem that can occur with an incorrect date sequence setting. For example, assume that you had the date sequence set to a four-digit year when you entered transactions, and you entered dates with a two-digit year (3/1/93). The dates would be stored as 3/1/0093. And, if you wanted the date to be in the 20th century, you could use the UPDATE command to modify the existing dates to 20th century dates by adding 693,960 to each date, which is the number of days between the beginning of the 1st and 20th centuries.

The [SET DATE](#) command makes sure that you are using a four-digit year. The UPDATE command changes all *transdate* values to 20th century dates, where the current value of the column is less than 1/1/1900. The last SET DATE command returns to a two-digit date sequence and format.

```
SET DATE MM/DD/YYYY
```

```
UPDATE transmaster SET transdate = (transdate + 693960) +  
WHERE transdate < 1/1/1900
```

```
SET DATE MM/DD/YY
```

Assume that you wanted to update the *inventory* table with the sum of the units sold from the *orders* table. Because there are many rows in the *orders* table for each part number, you cannot do this directly with the UPDATE command. The [CREATE VIEW](#) command creates a view containing the sum of the units sold from the *orders* table. The UPDATE command updates the *inventory* table by extracting the *totalsold* value from the view named *orders_view* for each part number.

```
CREATE VIEW orders_view (partid,totalsold) AS SELECT +  
partid, sum(sold) FROM orders GROUP BY partid
```

```
UPDATE inventory SET onhand = (T1.onhand - T2.totalsold) +  
FROM inventory T1, orders_view T2 +  
WHERE T1.partid = T2.partid
```

1.21.4 UPGRADE

Creates the R:BASE 7.x required system tables. You must be in single-user mode.

UPGRADE

About the UPGRADE command

Use UPGRADE command to create all required system tables for R:BASE 7.x, such as SYS_FORMS3, SYS_LABELS, SYS_LAYOUTS3, SYS_MERGE and SYS_REPORTS3.

Example: (at the R> Prompt)

DISCONNECT
 SET MULTI OFF
 CONNECT dbname
 UPGRADE
 LIST SYS%
 DISCONNECT
 SET MULTI ON
 CONNECT dbname

Databases that have already been converted to the 7.x format with above system tables will not need to be converted. However, if you are missing any one of the above system table(s), use UPGRADE command as explained above.

1.22 W

1.22.1 WHENEVER

Use the WHENEVER command in a program to check error conditions and run a set of commands designed to handle the error.

WHENEVER { <table> <tr> <td>SQLERROR</td> <td>]</td> <td>CONTINUE</td> </tr> <tr> <td>NOT FOUND</td> <td>]</td> <td>GOTO lblname</td> </tr> </table>	SQLERROR	]	CONTINUE	NOT FOUND	]	GOTO lblname
SQLERROR	]	CONTINUE				
NOT FOUND	]	GOTO lblname				

Options

CONTINUE

Turns off error checking by the WHENEVER command.

GOTO lblname

Passes control to the command following the indicated [LABEL](#) command.

NOT FOUND

Indicates that no rows were found by the command (*sqlcode* equals 100).

SQLERROR

Indicates that a processing error of any type other than *data-not-found* was detected (*sqlcode* is less than zero).

About the WHENEVER Command

Use WHENEVER to trap errors when R:BASE Structured Query Language (SQL) commands are run. WHENEVER traps all SQL commands. For more information about structured query language, see Structured Query Language.

R:BASE places the error value of the specific error that occurs in the R:BASE system variable *sqlcode*. You can check the value of *sqlcode* to determine what action to take. The NOT FOUND errors always have the value 100; other errors are the negative of the R:BASE error code number.

Errors in R:BASE commands-not R:BASE SQL commands- do not affect the value of the *sqlcode* error variable. Use an R:BASE error variable to check for R:BASE command errors.

Using the Options

Enter a WHENEVER SQLERROR command at the beginning of a command file or command block to check for any SQL processing error other than *data-not-found* errors. WHENEVER applies only to that command

or block. The system error variable *sqlcode* is set to the negative value of an error condition.

Enter a **WHENEVER NOT FOUND** command at the beginning of a command file or block to check for conditions such as *No rows satisfy the WHERE clause* or *End-of-data encountered* that occur within a program that is running.

The **CONTINUE** option turns off error checking and negates any previously entered **WHENEVER** command. That is, if you have entered a **WHENEVER NOT FOUND** with a **GOTO**, R:BASE runs the **GOTO** whenever data is not found. Then, if you enter a **WHENEVER NOT FOUND** command with a **CONTINUE**, data-not-found errors do not run the **GOTO** action. The **GOTO** action passes control to the specified label. You must have a matching **LABEL** command in the same command file or block within the procedure file running the **WHENEVER** command.

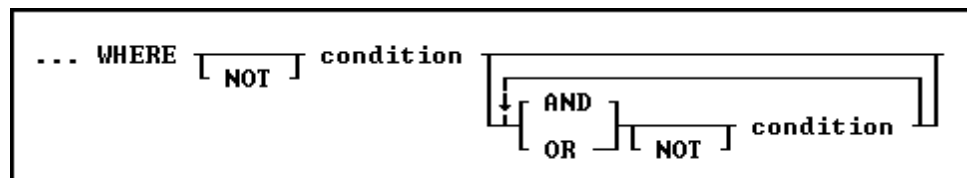
Example

In the following example, the [DECLARE CURSOR](#) and [OPEN](#) commands set a route to the *transmaster* table when the value of *transid* matches the value in the *vtran* variable. If the [FETCH](#) command does not find any rows, then R:BASE passes control to the commands that follow the [LABEL](#) command. The section after the **LABEL** command should contain commands to check the value of the *sqlcode* system variable and then perform some action according to the specific error condition that occurs.

```
WHENEVER NOT FOUND GOTO errors
.
.
.
DECLARE cur1 CURSOR FOR SELECT custid, netamount FROM +
    transmaster WHERE transid = .vtran
OPEN cur1
FETCH cur1 INTO vid, vamount
.
.
.
LABEL errors
*(error handling commands)
```

1.22.2 WHERE

Use a **WHERE** clause in commands to qualify or restrict the rows affected by a command.



Options

AND

Indicates the following condition must be met along with the preceding condition.

condition

Identifies requirements to be in the **WHERE** syntax.

NOT

Reverses the meaning of a connecting operator. **AND NOT**, for example, indicates that the first condition must be met and the following condition must not be met.

OR

Indicates the following condition can be met instead of the preceding condition.

About the WHERE Clause

In most commands, a WHERE clause follows the syntax diagram above. The [JOIN](#) command does not use that syntax diagram because it uses only comparison operators in a WHERE clause to compare two columns.

The two main elements in any WHERE clause are conditions and connecting operators.

We now support "COUNT = LAST" in two different ways. If the entire WHERE clause is "WHERE COUNT = LAST" then R:BASE works like it always has to quickly fetch the last row of the table. The NEW functionality is to have other conditions in the WHERE clause and you want the last row of whatever qualifies.

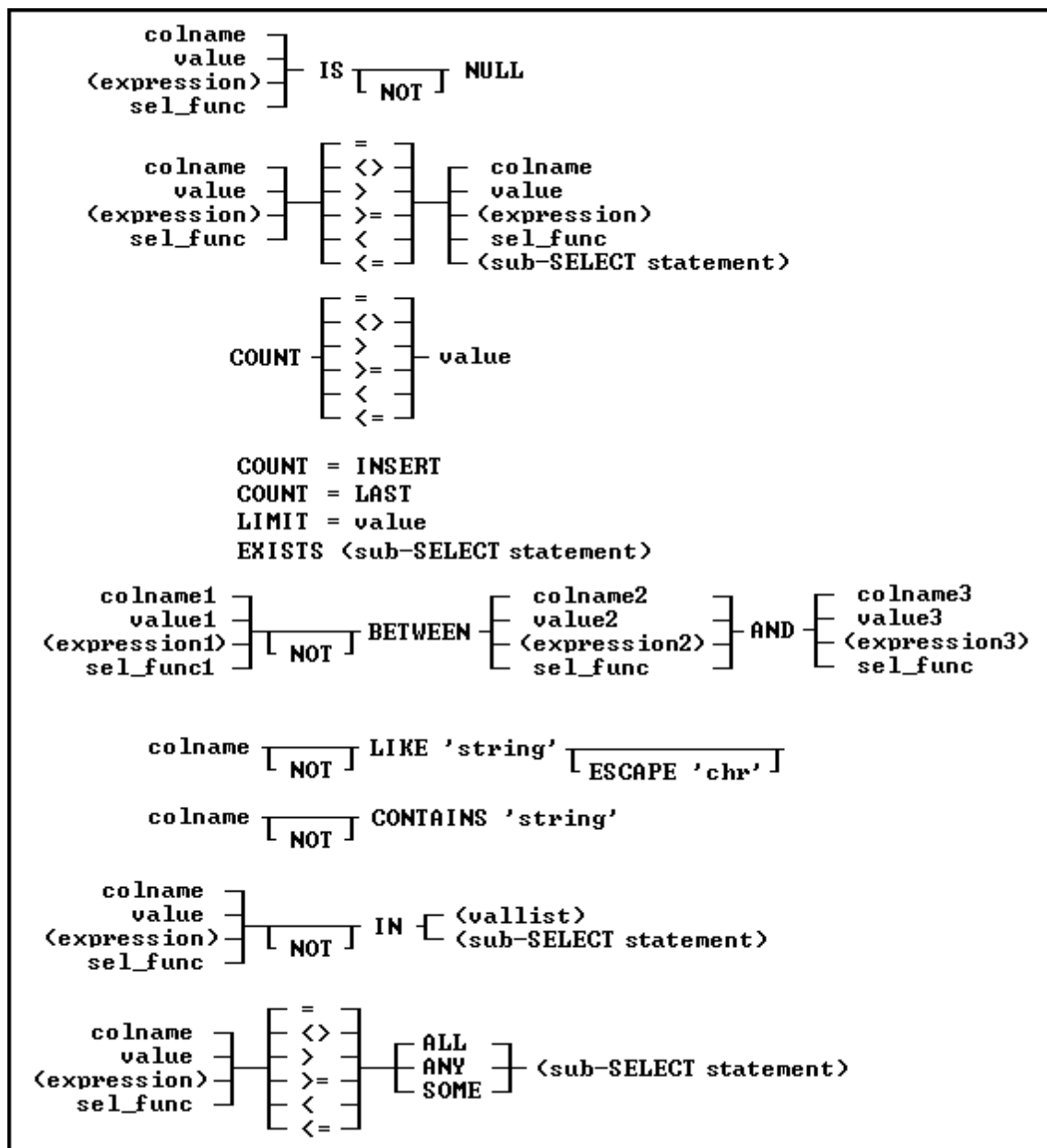
To make it work this way specify the other conditions and then add "AND COUNT = LAST".

Here is an example:

```
SELECT * FROM Customer WHERE CustID > 100 AND COUNT = LAST
```

WHERE Clause Conditions

The following syntax diagram and table show the basic formats for WHERE clause conditions, which can be used alone or together.



Basic WHERE Clause Conditions

Condition Syntax	Description
<code>colname op DEFAULT</code>	True if a column value compares correctly with the DEFAULT value for the column. <i>Op</i> can be <code>=</code> , <code><></code> , <code>>=</code> , <code>></code> , <code><=</code> , or <code><</code> .
<code>colname = USER</code>	True if a column value equals the current user identifier.
<code>item1 IS NULL</code>	True if <i>item1</i> has a null value. <i>Item1</i> can be a column name, value, or expression. A null value cannot be used in a comparison with an operator.
<code>item1 op item2</code>	True if the relationship between two items is true as defined by an operator. <i>Item1</i> can be a column name, value, or expression; <i>item2</i> can be a column name, value, expression, or sub-SELECT statement.
<code>COUNT=INSERT</code>	Refers to the last row inserted by the current user, even if it has been modified by another user.
<code>COUNT=LAST</code>	Refers to the last row in a table. COUNT cannot be used with a

	view.
COUNT <i>op value</i>	Refers to a number of rows defined by <i>op</i> and <i>value</i> .
LIMIT= <i>value</i>	Specifies a number of rows affected by a command. A LIMIT condition should be the last condition in a WHERE clause.
EXISTS (sub-SELECT statement)	True if sub-SELECT statement returns one or more rows.
<i>item1</i> BETWEEN <i>item2</i> AND <i>item3</i>	True if the value of <i>item1</i> is greater than or equal to the value of <i>item2</i> , and if the value of <i>item1</i> is less than or equal to the value of <i>item3</i> .
<i>colname</i> LIKE ' <i>string</i> '	True if a column value equals the text string. With LIKE, a string can also be a DATE, TIME, or DATETIME value. The text string can contain R:BASE wildcard characters.
<i>colname</i> LIKE ' <i>string</i> ' ESCAPE ' <i>chr</i> '	True if a column value equals a text string. If you want to use a wildcard character as a text character in the string, specify the ESCAPE character <i>1chr</i> . In the string, use <i>chr</i> in front of the wildcard character.
<i>colname</i> CONTAINS ' <i>string</i> '	True if a column value contains the text string.
<i>colname</i> SOUNDS ' <i>string</i> '	True if the soundex value of a column matches the soundex value of the text string.
<i>item1</i> IN (<i>vallist</i>)	True if <i>item1</i> is in the value list.
<i>item1</i> IN (sub-SELECT statement)	True if <i>item1</i> is in the rows selected by a sub-SELECT.
<i>item1 op</i> ALL (sub-SELECT statement)	True if the relationship between <i>item1</i> and every row returned by a sub-SELECT statement matches an operator.
<i>item1 op</i> ANY (sub-SELECT statement)	True if the relationship between <i>item1</i> and at least one value returned by a sub-SELECT statement matches an operator.
<i>item1 op</i> SOME (sub-SELECT statement)	ANY and SOME are equivalent.

Notes:

- Placing NOT before most text operators (such as NULL or BETWEEN) reverses their meaning.
- When a [SELECT](#) statement is part of a WHERE clause, it is called a [sub-SELECT](#) clause. A sub-SELECT clause can contain only one column name (not a column list or *), expression, or function. The [INTO](#) and [ORDER BY](#) clauses in a sub-SELECT are ignored.

You can only use the current wildcard characters to compare a column to a text value when using the LIKE comparison. The default wildcard characters are the percent sign (%), which is used for one or more characters, and the underscore (_), which is used for a single character.

If you compare a column with a value, you can either enter the value or specify a global variable. If you specify a variable, R:BASE compares the column with the current value of the variable.

To significantly reduce processing time for a WHERE clause, use INDEX processing. To use indexes, the following conditions must be met:

- A condition in the WHERE clause compares an indexed column.
- If the WHERE clause contains more than one condition, R:BASE selects the condition that places the greatest restriction on the WHERE clause.
- Conditions are not joined by the OR operator.
- The comparison value is not an expression.

Connecting Operators

When you use more than one condition in a WHERE clause, the conditions are connected using the connecting operators AND, OR, AND NOT, and OR NOT.

The connecting operator AND requires that both conditions it separates must be satisfied. The connecting operator OR requires that either condition it separates must be satisfied.

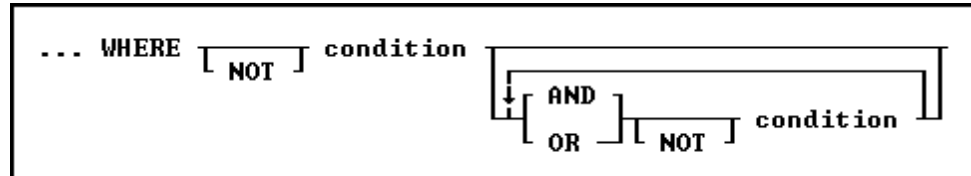
The connecting operator AND NOT requires that the preceding condition must be satisfied, and the following condition must not be satisfied. The connecting operator OR NOT requires that either the preceding condition must be satisfied, or any condition except the following condition must be satisfied.

In WHERE clauses with multiple conditions, conditions that are connected by AND or AND NOT are evaluated before those connected by OR or OR NOT. However, you can control the order in which

conditions are evaluated by either placing parentheses around conditions or using the [SET AND](#) condition. If you set AND off, conditions are always evaluated from left to right.

WHERE Builder

When launching the WHERE Clause Builder, the following window will appear:



Examples

The following WHERE clause chooses sales amounts that are less than the value of a variable containing the daily average.

```
... WHERE amount < .dailyave
```

The following WHERE clause specifies the seventh row.

```
... WHERE COUNT = 7
```

The following WHERE clause specifies each row from the *employee* table that contains both the first name *Jane* and the last name *Wilson*.

```
SELECT * FROM employee WHERE empfname = 'june' AND emplname = 'wilson'
```

The following WHERE clause selects dates in the *actdate* column that are greater than dates in the *begdate* column or are less than dates in the *enddate* column.

```
... WHERE actdate BETWEEN begdate AND enddate
```

The next three WHERE clauses use the following data:

```
empfname emplname
-----
Mary Jones
John Smith
Agnes Smith
John Brown
```

In both of the following clauses, R:BASE first evaluates the conditions connected by AND, selecting John Smith. Then R:BASE adds any Marys to the list because the connecting operator is OR. The final result includes John Smith and Mary Jones.

```
...WHERE empfname = 'Mary' OR empfname = 'John' +
AND emplname = 'Smith'
```

```
...WHERE empfname = 'Mary' OR (empfname = 'John' +
AND emplname = 'Smith')
```

By moving the parentheses around the conditions connected by OR, you can select only John Smith. In the following WHERE clause, the first name can be either Mary or John, but the last name must be Smith.

```
...WHERE (empfname = 'Mary' OR empfname = 'John') AND +
emplname = 'Smith'
```

The following example illustrates a sub-SELECT in a WHERE clause. Assume you wanted a list of all sales representatives that had transactions greater than \$100,000, and the information for such a list was contained in two tables, *employee* and *transmaster*. The relevant columns in these tables are:

employee		transmaster	
empid	emplname	empid	netamount
102	Wilson	133	\$32,400.00
129	Hernandez	160	\$9,500.00
133	Coffin	129	\$6,400.00
165	Williams	102	\$176,000.00
166	Chou	160	\$194,750.00
167	Watson	129	\$34,125.00
160	Smith	131	\$152,250.00
131	Simpson	102	\$87,500.00
102			\$22,500.00
102			\$40,500.00
131			\$108,750.00
131			\$80,500.00
129			\$56,250.00
102			\$57,500.00
160			\$140,300.00
129			\$95,500.00
129			\$155,500.00
133			\$88,000.00
131			\$130,500.00
102			\$3,060.00
165			\$3,060.00
167			\$3,830.00
133			\$12,740.00
165			\$26,310.00

To display a list of employees in the *transmaster* table with a transaction larger than \$100,000, enter the following command:

```
SELECT empid, emplname FROM employee WHERE empid IN +
  (SELECT empid FROM transmaster WHERE netamount > 100000)
```

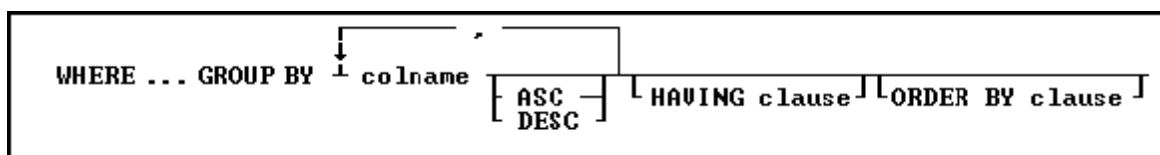
R:BASE displays the following list:

empid	emplname
102	Wilson
129	Hernandez
131	Simpson
160	Smith

Note: You can use a [sub-SELECT](#) in any command that allows a full WHERE clause.

1.22.2.1 GROUP BY

This clause determines which rows of data to include.



Options

'
Indicates that this part of the command is repeatable.

ASC DESC

Specifies whether to sort a column in ascending or descending order.

colname

Specifies a column name. In R:BASE eXtreme 9.0 (32), the column name is limited to 18 characters. In R:BASE eXtreme 9.0 (64), the column name is limited to 128 characters.

In a command, you can enter *#c*, where *#c* is the column number shown when the columns are listed with the [LIST TABLES](#) command. In an SQL command, a column name can be preceded by a table or correlation name and a period (*tblname.colname*).

GROUP BY

Returns a groups of rows as a summary resulting in only unique rows. This option is generally used with [SELECT](#) commands.

HAVING clause

Limits the rows affected by the GROUP BY clause.

ORDER BY clause

Sorts rows of data.

About the GROUP BY command

This optional clause groups rows according to the values in one or more columns and sorts the results. GROUP BY consolidates the information from several rows into one row. This results in a table with one row for each value in the named column or columns and one or more values per column.

The columns listed in the GROUP BY clause are related to those listed in the command clause. Any column named in the GROUP BY clause can also be named in the command clause, but any column not named in the GROUP BY clause can be used only in the command clause if the column is used in a SELECT command.

Examples

The [SELECT](#) command clause can contain the columns named in the GROUP BY clause, and SELECT functions that refer only to columns not named in the GROUP BY clause. Because the GROUP BY clause processes information resulting from a WHERE clause, you can add a GROUP BY clause to see the sales each employee has made:

```
SELECT empid FROM transmaster WHERE netamount < $100,000 +
GROUP BY empid
```

The following intermediate result table contains columns not named in the command clause because the command clause has not been processed yet (not all the columns fit in the display, however). The first part of the processing is to group the rows by *empid*. Because seven different employees are included, the intermediate result table includes seven rows.

Intermediate Result Table-GROUP BY empid

<u>transid</u>	<u>custid</u>	<u>empid</u>	<u>netamount</u>
4975, 4980, 5000,	101, 101, 101, 101, 100	102	\$87,500, \$22,500, \$40,500,
5060, 5045			\$57,500, \$3,060
4790, 4865, 5050,	104, 102, 104, 104	129	\$6,400, \$34,125, \$56,250,
5070			\$95,500
5015	103	131	\$80,500
4760, 5080, 5048	100, 100, 103	133	\$32,400, \$88,000, \$12,740

4780	105	160	\$9,500
5046, 5049	101, 102	165	\$3,060, \$26,310
5047	102	167	\$3,830

You can include more than one column in a GROUP BY clause. If you group the rows in the above example by *custid* as well as *empid*, the command looks like this:

```
SELECT empid, custid FROM transmaster +
WHERE netamount < $100,000 GROUP BY empid, custid
```

In the following table, rows are now grouped by both *empid* and *custid*, resulting in eleven groups.

Intermediate Result Table-GROUP BY empid and custid

<u>transid</u>	<u>custid</u>	<u>empid</u>	<u>netamount</u>
5045	100	102	\$3,060
4975, 4980, 5000, 5060	101	102	\$87,500, \$22,500, \$40,500, \$57,500
4865	102	129	\$34,125
4790, 5050, 5070	104	129	\$64,000, \$56,250, \$95,500
5015	103	131	\$80,500
4760, 5080	100	133	\$32,400, \$88,000
5048	103	133	\$12,740
4780	105	160	\$9,500
5046	101	165	\$3,060
5049	102	165	\$26,310
5047	102	167	\$3,830

If one or more of the columns named in the GROUP BY clause contain null values, R:BASE forms a separate group for null values. Review the result of this [SELECT](#) command for the *employee* table:

```
SELECT empid, empname, hiredate, emptitle FROM employee
```

<u>empid</u>	<u>empname</u>	<u>hiredate</u>	<u>emptitle</u>
102	Wilson	03/18/90	Manager
129	Hernandez	08/28/91	Manager
131	Smith	04/14/92	-0-
133	Coffin	11/26/93	Representative
160	Simpson	01/09/94	-0-
165	Williams	07/05/92	Representative
167	Watson	07/10/92	Representative
166	Chou	07/10/93	Sales Clerk

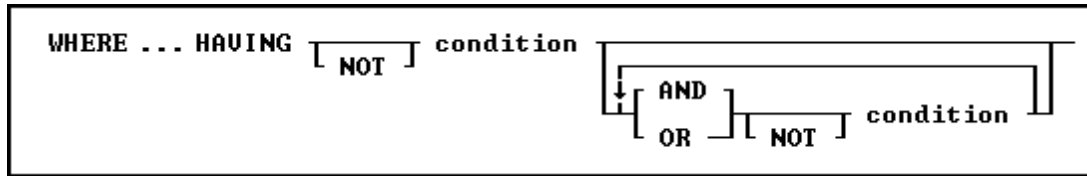
If you group these rows by the *emptitle* column, which contains null values, you get the following intermediate result table:

Intermediate Result Table-GROUP BY emptitle

<u>empid</u>	<u>empname</u>	<u>hiredate</u>	<u>emptitle</u>
102, 129	Wilson, Hernandez	03/18/90, 08/28/91	Manager
133, 165, 167	Coffin, Williams, Watson	11/26/93, 07/05/92, 07/10/92	Representative
166	Chou	07/10/93	Sales Clerk
131, 160	Smith, Simpson	04/14/94, 01/09/94	-0-

1.22.2.2 HAVING

This clause determines which rows of data to include based on the results of a prior [GROUP BY](#) clause.



Options

AND OR

AND indicates two conditions must both be true.

OR indicates either condition must be true.

condition

Specifies a combination of one or more expressions and/or operations that would evaluate to either true or false. See the "HAVING Conditions" below.

NOT

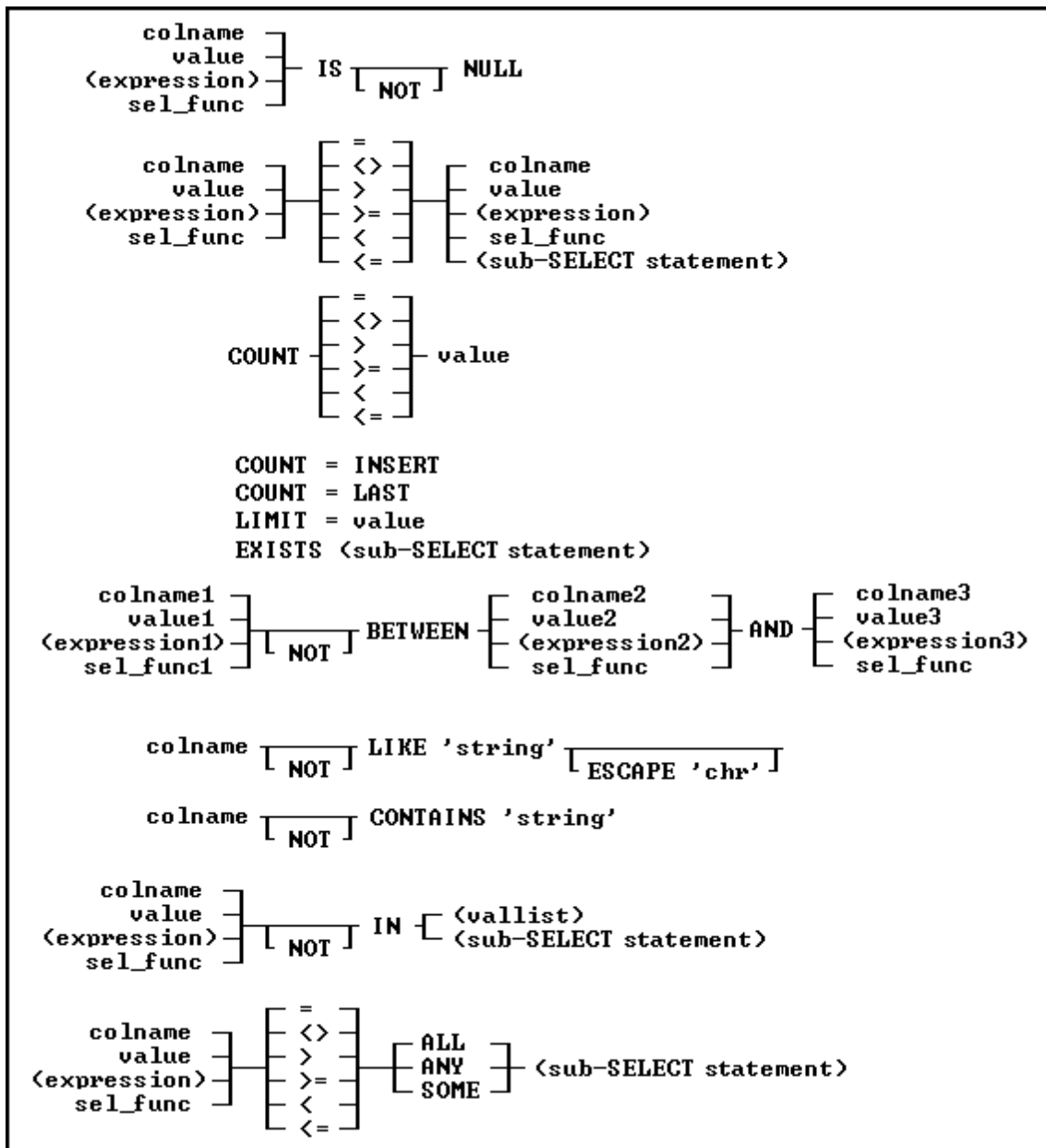
Reverses the meaning of an operator or indicates that a condition is not true.

About the HAVING command

The optional HAVING clause selects rows that meet one or more conditions from among the results of the GROUP BY clause. HAVING works the same as a WHERE clause with the following exceptions:

- A WHERE clause modifies the intermediate results of a [FROM](#) clause; a HAVING clause modifies the intermediate results of a GROUP BY clause.
- A HAVING clause can include [SELECT](#) functions.

HAVING Conditions:



Examples

To display sales information for only those employees who have made more than one sale to the same customer, add a HAVING clause such as the following to one of the examples shown previously in [GROUP BY](#). When used in a HAVING clause, [SELECT functions](#) compute results based on the values grouped in the specified column. In this HAVING clause, COUNT returns the number of values grouped in the *transid* column.

```
SELECT empid, custid FROM transmaster +
WHERE netamount < $100,000 +
GROUP BY empid, custid HAVING COUNT(transid) > 1
```

Intermediate Result Table-HAVING COUNT(transid) > 1

<u>transid</u>	<u>custid</u>	<u>empid</u>	<u>netamount</u>

4975, 4980, 5000, 5060	101	102	\$87,500, \$22,500, \$40,500, \$57,500
4790, 5050, 5070	104	129	\$6,400, \$56,250, \$95,500
5080	100	133	\$32,400, \$88,000

1.22.3 WHILE/ENDWHILE

Use the WHILE...ENDWHILE structure in a program to continuously run a set of commands based on a specified condition.

```
WHILE condlist THEN
  while-block
ENDWHILE
```

Options

condlist

Specifies a list of conditions that identify the requirements to be met.

while-block

Specifies commands to be executed if the WHILE condition is true.

About the WHILE...ENDWHILE Command

A WHILE ... ENDWHILE structure consists of conditions, commands, and an ENDWHILE statement. As long as WHILE conditions are true, R:BASE runs the commands repeatedly.

WHILE Conditions

The basic WHILE conditions are the same as those used in an [IF...ENDIF](#) structure and are as follows:

Condition	Description
<i>varname</i> IS NULL	The value of the variable is null.
<i>varname</i> IS NOT NULL	The value of the variable is not null.
<i>varname</i> CONTAINS ' <i>string</i> '	The variable has a TEXT data type and contains a ' <i>string</i> ' as a substring in the variable value.
<i>varname</i> NOT CONTAINS ' <i>string</i> '	The variable has a TEXT data type and a ' <i>string</i> ' is not contained as a substring in the variable value.
<i>varname</i> LIKE ' <i>string</i> '	The variable equals a ' <i>string</i> .' A ' <i>string</i> ' can contain wildcards.
<i>varname</i> NOT LIKE ' <i>string</i> '	The variable does not equal the ' <i>string</i> '. A ' <i>string</i> ' can contain wildcards.
<i>varname</i> BETWEEN <i>value1</i> AND <i>value2</i>	The value of the variable is greater than or equal to <i>value1</i> and less than or equal to <i>value2</i> . The variable and the values must be the same data type.
<i>varname</i> NOT BETWEEN <i>value1</i> AND <i>value2</i>	The value of the variable is less than <i>value1</i> or greater than <i>value2</i> . The variable and the values must be the same data type.
<i>item1</i> op <i>item2</i>	<i>Item1</i> has the specified relationship to <i>item2</i> . <i>Item1</i> can be a column name, value, or expression; <i>item2</i> can be a column name, value, or expression.

An expression can be substituted for the first variable in each of the condition formats shown above.

You can only use wildcard characters with the LIKE and NOT LIKE operators. For example, *varname* LIKE '*string*%'.

You can combine conditions from the WHILE condition list by using the connecting operators AND, OR, AND NOT, and OR NOT. Be careful when using these conditions in a condition list. Conditions connected

by AND are evaluated first, then conditions connected by OR are evaluated. However, you can use parentheses to set the evaluation order.

WHILE Loop Commands

All WHILE loop commands are retained in memory, so a WHILE loop runs more quickly than a [GOTO](#) or [LABEL](#) structure. A computer must have enough available memory to read all of the commands in a WHILE loop, or the program terminates abnormally.

R:BASE optimizes commands in a WHILE loop so that it runs more quickly. Use the following guidelines when constructing WHILE loops so they run more quickly.

- Do not clear variables in the WHILE loop. Rather, set those variables to null.
- Do not define variables within the WHILE loop. Only define variables outside of the loop because the values can change within the loop.
- If you issue multiple [SET VARIABLE](#) commands on a single command line, then those variables will not be optimized. If you want to increase the speed for that loop, you should put the SET VARIABLE commands on separate lines.

To turn off WHILE loop optimization, set [WHILEOPT](#) off.

The ENDWHILE Statement

ENDWHILE indicates the end of the loop. Place an ENDWHILE statement at the end of each WHILE loop. Each time R:BASE reaches the ENDWHILE statement, R:BASE returns to the WHILE command at the top of the loop and checks to see whether the conditions are still true or false. If true, R:BASE again runs the commands between the WHILE and the ENDWHILE. If false, R:BASE runs the command line immediately following the ENDWHILE.

Exiting from a WHILE Loop

To exit from a WHILE loop before the WHILE condition becomes false, use an [IF...ENDIF](#) structure to check other conditions, then use [BREAK](#) to exit from the WHILE loop. The BREAK command causes the WHILE loop to terminate when the conditions specified in the IF statement become true.

Never use GOTO to exit from a WHILE loop; use BREAK instead. BREAK clears the WHILE loop. When you do not use BREAK or the naturally occurring exit (that is, when the WHILE loop conditions are no longer true) to exit from a WHILE loop, R:BASE continues to read commands into memory. If you have a large command or procedure file, you can run out of memory and your program terminates abnormally.

Skip to the next WHILE Occurrence

Use the [CONTINUE](#) command to move to the next occurrence of the WHILE loop and run the code.

In the following example, when the code is run, processing returns to line 3 after it completes the CONTINUE command on line 6. The while-block commands in line 8 are not run.

```
SET VARIABLE v1=0
SET VARIABLE v2=1
WHILE v1 = 0 THEN
  *(while-block commands)
  IF v2 <> 0 THEN
    CONTINUE
  ENDIF
  *(while-block commands)
ENDWHILE
```

Example

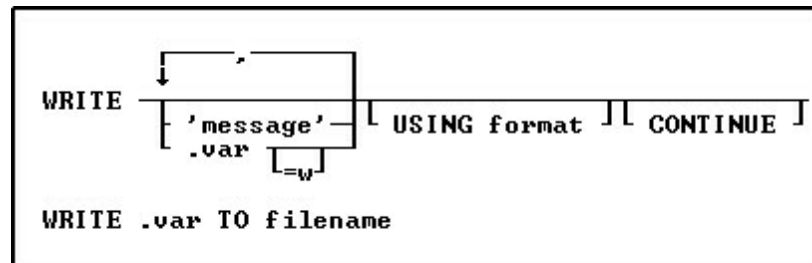
In the following example, R:BASE runs the commands in the WHILE block and evaluates the v2 condition in the IF statement. If v2 is not equal to zero, R:BASE runs the BREAK command and terminates the WHILE loop. R:BASE then runs the commands immediately following the ENDWHILE statement. As long as the WHILE condition (v1) is true and the IF condition (v2) remains false, the WHILE loop continues

processing.

```
SET VARIABLE v1 = 0
WHILE v1 = 0 THEN
  *(while-block commands)
  IF v2 <> 0 THEN
    BREAK
  ENDIF
  *(while-block commands)
ENDWHILE
*(next command outside the while-block)
```

1.22.4 WRITE

Use the WRITE command to send a message to an output device or to display a message on R> Prompt screen. In this context, the WRITE command is primarily used with R:BASE for DOS. To display messages in R:BASE for Windows, use the [PAUSE](#) command. Another use for the WRITE command is to recreate binary or large ASCII data to a file.



Options

'
Indicates that this part of the command is repeatable.

'message'

Specifies a message to be displayed on the screen.

CONTINUE

Suppresses the carriage return line feed.

filename

The name of the file for writing variable data to a file.

USING format

Specifies a display format for the message or variable.

.var

Specifies a dot variable of any data type, which can be used instead of a message.

=w

Specifies the display width for the variable.

About the WRITE Command

The WRITE command allows concatenation of several items in one command with embedded variables.

The WRITE command displays a message or variable that you place within the current QUOTES character. The default QUOTES character is the single quote ('). The quotes are not displayed when WRITE runs.

After you place binary or large ASCII data into your R:BASE database, the original file can be recreated at any time using the WRITE command. The data is read from the table into a variable, then the variable is written to a file. This process recreates the file exactly.

If the current output device is not the screen, the AT and screen attributes options do not have any effect unless [PAGEMODE](#) is set on.

To display a message or variable containing line end characters, you must specify a width to write multiple lines.

Symbols for Formatting the Display

You can specify a format for displaying either a message or a variable. When using a display format, you can only display one item, and you must allow enough space between the quotes to properly display the formatted information. For example, if you wanted to center a text message, the format must include the centering symbol ([^]) and the space within which the text will be centered.

The following tables summarize the formatting symbols.

String Modifiers for All Messages

Use this symbol...	To do the following...
[<]	Left justify the data in the field.
[>]	Right justify the data in the field.
[^]	Center the data in the field.

String Modifiers for Messages with Numbers

Use this symbol...	To do the following...
[-]	Place a negative sign to the right of a negative number.
[CR]	Place "CR" to the right of a positive number.
[DB]	Place "DB" to the right of a negative number.
[()]	Enclose a negative number in parentheses.

Character Modifiers for Text Messages

Character	Modifier	If character is a letter	If character is a number	If character is neither
		uppercase	blank	blank
		lowercase	blank	blank
%		uppercase	number	character
?		lowercase	number	character

Character Modifiers for Numeric Messages

Character	Modifier	If character is a number	If character doesn't fill the specified format
9		number	blank
0		number	0
*		number	* if blank

Examples

The following command automatically concatenates the items and puts a space between them. An ampersand (&) or plus (+) character cannot be used for concatenations.

```
WRITE 'Today's date is', .#DATE
```

The following command displays the contents of the *vmmsg* variable.

```
WRITE .vmmsg
```


by default in the C:\WINNT\SYSTEM32 directory.

/C 'batchfile'

Specifies a DOS Batch File to be executed by the command shell.

About ZIP

Zip allows you to call an external program from within R:BASE. ZIP operates slightly differently in DOS R:BASE or Windows R:BASE as noted above. In R:BASE eXtreme if you are running the Windows version of R:BASE you can specify a Windows program to be executed.

In either Windows or DOS you can call a batch file that uses the Windows Command START to launch a Windows program. For example START /WAIT NOTEPAD in a Batch File will start the Notepad Program and wait for Notepad to close before returning control to R:BASE. If you do not use the /WAIT option then as soon as Start had spawned the Notepad program IT would finish and control would pass back to R:BASE. This is simply because R:BASE is only watching START. Use START /? at a Command Prompt to see the other options available.

Example

The following command runs an external program named *SmallEd*. When you exit from *SmallEd*, R:BASE continues at the point where you ran the ZIP command.

ZIP smallEd

The following command, when run in R:BASE Windows, runs Notepad and specifies that the ABC.TXT file be opened.

ZIP Notepad ABC.TXT

Index

- - -

-- 43

- # -

#FEEDBACK 432

#TABLEORDER 436

- { -

{ } 43

- | -

||format 82

- A -

ABORT 3

Alias 383

ALTER TABLE 3

AND 418

ANSI 418

APPEND 8

APPLICATION 155, 359

Application Designer 364

ApplicationFormProperty 155, 359

ApplicationProperty 155

AS 408

ASCII 148

ATTACH 9, 387

AUTOCHK 11

AUTOCOMMIT 419

AUTOCONVERT 419

AUTODROP 419

AUTONUM 14

AUTORECOVER 420

AUTOROWVER 420

AUTOSKIP 420

AUTOSYNC 420

AUTOUPGRADE 420

- B -

BACKUP 17

BEEP 20

BELL 421

BLANK 421

BMP 179, 291

BREAK 20

BROWSE 21

BROWSE USING 24

Builder 267

Buttons 90

- C -

CALCULATOR 102

CALL 25

CAPTION 421

CASCADE 26

CASE 421

CHDIR 26

CHDRV 27

CHKDSK 28

CHOOSE 29, 267

CLBR 141

CLEAR 39, 421

CLIPBOARD 422

CLOSE 40

CLOSEWINDOW 40

CLS 41

CODELOCK 41

COLOR 422

Command 2, 371, 393

Commands 3, 8, 9, 11, 14, 17, 20, 21, 24, 25, 26,
 27, 28, 29, 39, 40, 41, 43, 44, 46, 48, 50, 51, 52, 55,
 57, 61, 69, 70, 71, 74, 75, 76, 77, 78, 102, 104, 105,
 106, 109, 112, 114, 115, 116, 117, 119, 120, 154, 155,
 156, 157, 161, 162, 164, 166, 167, 169, 170, 172, 173,
 177, 179, 181, 183, 184, 186, 193, 196, 203, 207, 209,
 210, 212, 220, 223, 225, 228, 230, 236, 237, 238, 239,
 240, 241, 244, 246, 256, 282, 283, 284, 288, 291, 293,
 294, 297, 298, 300, 308, 311, 312, 321, 323, 325, 329,
 331, 333, 335, 337, 339, 347, 350, 352, 355, 357, 359,
 360, 362, 363, 364, 365, 366, 367, 369, 370, 371, 372,
 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 386,
 387, 389, 390, 396, 398, 399, 400, 401, 403, 408, 410,
 412, 414, 415, 416, 417, 456, 460, 464, 465, 466, 467,
 468, 469, 471, 474, 476, 481, 484, 485, 486, 491, 494,
 496, 498, 500
 COMMENT 43
 COMMENT ON 44
 COMMIT 46
 COMPATIB 423
 Component ID 155, 359
 COMPUTE 46
 CONNECT 48
 CONTINUE 50, 498
 ControlProperty 155, 359
 CONVERT 50
 COPY 51
 COUNT 393
 Create Custom Plugins 276
 CREATE INDEX 52
 CREATE SCHEMA 55
 CREATE TABLE 57
 CREATE VIEW 61
 CREATEOBJECT 64
 CROSSTAB 69
 CSV 124, 148
 CURRENCY 424
 CUS 148
 Custom Plugin 276
 CustomFormActionName 359

- D -

Data Designer 365
 Data Interchange Format 293
 Data Source 383, 387
 DATE 98, 425
 DATETIME 100
 DB 122, 147

DBF 123, 148
 DEBUG 70, 426
 DECLARE CURSOR 71
 DEFF 74
 DELETE 74
 DELETE DUPLICATES 75
 DELETEOBJECT 76
 DELIMIT 426
 DELROW 76
 DETACH 77
 DIALOG 78, 88, 90, 93, 94, 95, 98, 99, 100, 102,
 257, 260
 Dialog Builder 257
 Dialog With Button Builder 260
 DIF 135, 293
 DIR 102
 DISCONNECT 104
 DISPLAY 105
 DISTINCT 393
 DOC 131
 DROP 106
 DSN 387
 DSN-Less 387, 389, 390
 DUPROW 109

- E -

ECHO 427
 EDIT 109
 EDIT USING 112
 EditMask 82
 EDITOR 366, 427
 EMAIL 223, 350
 EMF 181, 294
 ENDIF 162
 ENDSW 469
 ENDWHILE 496
 ENTER 114
 ENTER USING 114
 EOFCHAR 428
 EQNULL 428
 ERASE 115
 ERROR MESSAGE 429
 ERROR MESSAGES 429
 ERROR VARIABLE 429
 ESCAPE 431
 ETXT 183, 297
 EXIT 115

EXITFORM 115
EXPLODE 431
EXPORT 121
external themes 359

- F -

FASTFK 431
FASTLOCK 432, 442
FEEDBACK 432
FETCH 116, 398
File Gateway 120
FILES 433
FILLIN 117
FIX 126
FIXED 433
fixed width 126
FLUSH 119
FONT 433
foreign data source 387, 390
foreign database 383
foreign table 383
format mask 82
FormProperty 155, 359
FORMS 119
FormTableName 359
FROM 399

- G -

GATEWAY 120, 121, 146
GAUGE Progress Bar 246
GET 154
GETPROPERTY 155
GIF 184, 298
GOTO 156
GRANT 157
GROUP BY 410, 491

- H -

HAVING 412, 494
HEADINGS 433
HELP 161
HTM 127, 151
HTML 151, 186, 300, 416

- I -

IDQUOTES 383, 433
IF 162
IF/ENDIF 162
IMPORT 146
INDEXONLY 434
INNER 396
INPUT 164
INSERT 164, 434
INTERSECT 166
INTERVAL 434
INTO 398
ISV 148

- J -

JOIN 167, 396
JPG 193, 308

- K -

KEYMAP 434

- L -

LABEL 169
Labels 366
LAUNCH 170
LAYOUT 435
LBLPRINT 172, 173, 177, 179, 181, 183, 184, 186,
193, 196, 203, 207, 209, 210, 212, 220, 223
LDIF 145
LIMIT 400
LINEEND 435
LINES 435
LIST 157, 225
LIST ACCESS 228
LISTOF 393
LOAD 230
load directory 271
load file 272, 273, 274
load folder 271
LoadDirectoryName.rbl 271
LoadFileName.rbl 272
LoadFileNamePlus.rbl 273

LoadFileNamePlusPlus.rbl 274
LOADWINDOW 155, 359
LOCK 435
LOOKUP 95, 436
Lotus 311

- M -

MANOPT 436
MANY 437
MAPI 270
MAPI Mail 270
mask 82
MAX 393
MAXIMIZE 236
MAXTRANS 437
MDB 153
MDI 437
message 88
MESSAGES 437
Meter 246
Meter Progress Bar 246
MIN 393
MINIMIZE 236
MIRROR 438
MKDIR 236
MODAL 119, 169, 374
MOUSE 438
MULTI 438, 442
Multi-Line 246
MultiLine PAUSE 246
Multi-Line PAUSE 246

- N -

NAME 438
NAMEWIDTH 439
NEWPAGE 237
NEWROW 237
NEXTROW 237
NEXTTAB 237
NORMALIZE 237
Note/BLOB Field Viewer/Editor 364
NOTE_PAD 439
NULL 439

- O -

ODBC 387
ON 223, 350
ON CONNECT 238
ON CONNECT/PACK 238
ON ERROR 239
ONLINE 439
OPEN 239
ORDER BY 240, 414
OUTER JOIN 401
OUTPUT 241
OWNER 473

- P -

PACK 244
PAGEMODE 440
Parameters 359
PASSTHROUGH 440
password 78
PAUSE 246, 263
PDF 144, 196, 312
PLUGINS 256, 257, 260, 263, 266, 267, 270, 271, 272, 273, 274, 276
PLUS 441
PNG 321
popup 95
Pop-up 95
Portable Network Graphics 321
POSFIXED 441
PREVROW 282
PREVTAB 282
PRINT 283, 284, 288, 291, 293, 294, 297, 298, 300, 308, 311, 312, 321, 323, 325, 329, 331, 333, 335, 337, 339, 347, 350
PRINTER 177, 288, 441
PRNSETUP 352
PROCEDURE 441
ProcessMessage 355
PROGRESS 442
Progress bar 246
PROJECT 357
PROPERTY 359
PUT 360

- Q -

QPRO 323
QSV 148
QUALCOLS 442
Qualkey 383
Quattro Pro 323
QUERY 362
QUIT 363
QUOTES 359, 443

- R -

R:BASE Editor 366
R:Docs 155, 359
RBA_FORM 155, 359
RADMIN 443
RBAPP 364
RBASE_FORM 155, 359
RBASE_FORM_ACTION 359
RBBEDIT 364
RBDEFINE 365
RBEDIT 366
RBLABELS 366
RBSYNC 367
RBTI_TIMEOUT 449
RCalculator 276
RCalculator.rbl 276
RECALC 369
RECOVER 370
RECYCLE 444
REFF 371
REFRESH 444
RELOAD 371
RENAME 372
REPORT 155, 359
ReportProperty 155, 359
REPORTS 374
RESETROW 374
RESTORE 375
RETURN 376
REVERSE 444
REVOKE 377
RHELP 378
RHide 379
RMDIR 379

ROLLBACK 380
ROWLOCKS 442, 445
RPHONE 380
RSHOW 381
RTF 142, 203, 325
RULES 381, 445
RUN 382
RWhois 266
RWhois.rbl 266

- S -

SATTACH 383, 387, 389, 390
SAV 143
SAVEROW 386
Scalable Vector Graphics 329
SCONNECT 383, 387, 389, 390
SCRATCH 445
SCREEN 173, 284
SCREEN RESTORE 389
SDETACH 383, 389, 390
SDISCONNECT 383, 387, 389, 390
SELECT 390, 393, 396, 398, 399, 400, 401, 403, 408, 410, 412, 414, 415, 416
SELECT Functions 393
SELMARGIN 446
SEMI 446
SERVER 446
SET 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456
SET VARIABLE 456
SETFOCUS 460
SHORTNAME 447
SHOW 460
SHOW BUILD 464
SHOW ERROR 464
SHOW VARIABLE 465
SHOW VERSION 466
SINGLE 447
SKIP 467
SLK 134
SORT 448
SORTMENU 448
SQL 138, 383
SSQL 468
STATICDB 442, 448

Sub-SELECT 408
SUBTRACT 468
SUM 393
SVG 329
SWITCH 469
SWITCH/ENDSW 469
SYLK 331
Symbolik Link 331
Syntax 2

- T -

TAB 133, 148
TABLE 359
TALLY 471
TEMPORARY 383
theme 359
ThemeName 359
ThemeProperty 359
Themes 29, 246
TIFF 207, 333
TILE 471
TIME 99, 448
TIMEOUT 449
TOLERANCE 450
TOP 396
TRACE 450, 471
TRANSACT 450
TURBO 473
TXT 148, 209, 335
TYPE 474

- U -

UNION 415, 474
UNLOAD 476
UPDATE 481
UPGRADE 484
USER 451

- V -

Value 359
VarName 155
VERIFY 452

- W -

WAIT 453
WALKMENU 453
WB1 323
WHENEVER 485
WHERE 403, 486, 491, 494
WHERE CURRENT 71
WHILE 496
WHILE/ENDWHILE 496
WHILEOPT 454
WIDTH 454
WINBEEP 454
Window 94
WindowProperty 155, 359
WK1 136, 311
WKS 150
WMF 210, 337
WQ1 137, 152
WRAP 455
WRITE 498
WRITECHK 455

- X -

XHTML 212, 339
XLS 128, 129, 149, 220, 347
XML 140, 152

- Z -

ZERO 456
ZIP 500
ZOOMEDIT 456

Notes