From The Edge:

by

A. Razzak Memon

Understanding Temporary Tables and Views

R:BASE 7.x and V-8 “Turbo” for Windows

May 2, 2005

From the Edge:	Understanding Temporary Tables and Views
Section:	General
Chapter:	Running R:BASE Your Way!
Platform:	R:BASE 7.x and V-8 "Turbo" for Windows
Builds:	7.1.79.30128 or higher

Background:

- TEMPORARY TABLES were first introduced in R:BASE 6.1 (July 1997).

Features:

- CREATE TEMPORARY TABLE tablename creates a temporary table that disappears when the database is disconnected.
- CREATE TEMPORARY VIEW viewname creates a temporary view that disappears when the database is disconnected.
- PROJECT TEMPORARY newtablename FROM TableName USING ... creates a temporary table that disappears when the database is disconnected.
- You may define RULES, CONSTRAINTS and TRIGGERS on TEMPORARY tables
- You may define INDEX for any column in TEMPORARY table
- When creating TEMPORARY TABLE/VIEW, following temporary System Tables are also created:
    - SYSTMP_COMMENTS
    - SYSTMP_CONSTRAINTS
    - SYSTMP_DEFAULTS
    - SYSTMP_RULES
    - SYSTMP_SERVERS
    - SYSTMP_TRIGGERS
    - SYSTMP_VIEWS
- TEMPORARY Tables/Views are also supported when STATICDB is set to ON.
- You may use DROP tablename or DROP viewname to DROP TEMPORARY table or view.
- Always DROP the table or view before creating a temporary table or view. This only affect the user running the program. To avid the error message, use the following technique:

  Example 01:

  ```
  SET ERROR MESSAGE 2038 OFF
  DROP TABLE temptablename
  SET ERROR MESSAGE 2038 ON
  ```

  Example 02:

  ```
  SET ERROR MESSAGE 677 OFF

  DROP VIEW tempviewname
  SET ERROR MESSAGE 677 ON
  ```

- You may define Forms, Labels or Reports based on a TEMPorary Table/View

  Before Designing/Running that Form, Label or Report, you MUST create referenced TEMPorary Table(s)/View(s).

  Create a startup file, such as TempTables.RMD, to pre-define all required temporary tables/views as demonstrated in all sample applications bundled with R:BASE 7.x and V-8 "Turbo".

- TEMPORARY Tables and/or VIEWS along with temporary SYSTEM tables are DROPped when the database is DISConnected.

  You can simply DISCONNECT and CONNECT the database to DROP TEMPORARY Table(s)/View(s).

- How to differentiate between Regular and Temporary Tables/Views when using the LIST or LIST TABLES command at the R> prompt.

  To safely indicate which tables are temporary tables in the LIST or LIST TABLES command, you will now see a "(T)" in front of the table name.

  Here's how:

  A. Launch RBG7, RBG75 or RBG8

  B. CONNECT Concomp

  C. Switch to R> Prompt and create a TEMPorary tables as following:

     ```
     PROJECT TEMPORARY tCustomer FROM Customer USING ALL
     ```

     Or use the following block of code in a command file to create two temporary tables

     ```
     SET ERROR MESSAGE 2038 OFF
     DROP TABLE tInvoiceHeader
      CREATE TEMPORARY TABLE `tInvoiceHeader` +
     (`TransID` INTEGER, +
      `CustID` INTEGER, +
      `EmpID` INTEGER, +
      `TransDate` DATE, +
      `BillToCompany` TEXT (40), +
      `BillToAddress` TEXT (30), +
      `BillToCity` TEXT (20), +
      `BillToState` TEXT (2), +
      `BillToZip` TEXT (10), +
      `ShipToCompany` TEXT (40), +
      `ShipToAddress` TEXT (30), +
      `ShipToCity` TEXT (20), +
      `ShipToState` TEXT (2), +
      `ShipToZip` TEXT (10), +
      `NetAmount` CURRENCY, +
      `Freight`= ( netamount* .01) CURRENCY, +
      `Tax`= ( netamount* .081) CURRENCY, +
      `InvoiceTotal`= (NetAmount+Freight+Tax) CURRENCY)
     COMMENT ON TABLE tInvoiceHeader IS 'Invoice Header Information'
     ```

```
DROP TABLE tInvoiceDetail
CREATE TEMPORARY TABLE `tInvoiceDetail` +
 (`TransID` INTEGER, +
 `DetailNum` INTEGER, +
 `Model` TEXT (6), +
 `Units` INTEGER,  +
 `Price` CURRENCY, +
 `Discount` REAL, +
 `SalePrice`= (Price-(Price*Discount/100)) CURRENCY, +
 `ExtPrice`= (Units* SalePrice) CURRENCY)
 AUTONUM `DetailNum` IN `tInvoiceDetail` USING 1,1
COMMENT ON TABLE tInvoiceDetail IS 'Invoice Header Information'
SET ERROR MESSAGE 2038 ON

LIST TABLES
```

You will notice the (T) in front of the temporary tables.

D. Now create a TEMPorary VIEW as following:

```
SET ERROR MESSAGE 677 OFF
DROP VIEW tYTDInvoiceTotal
    CREATE TEMP VIEW `tYTDInvoiceTotal` +
    (CustID, YTDInvoiceTotal) AS  +
    SELECT CustID,(SUM(InvoiceTotal)) FROM TransMaster +
    GROUP BY CustID
    COMMENT ON VIEW `tYTDInvoiceTotal` IS +
    'Year-To-Date Invoice Total by Customer'
SET ERROR MESSAGE 677 ON

LIST VIEWS
```

You will notice (T) in front of the tYTDInvoiceTotal view.

- Always DISCONNECT and then CONNECT before using the AUTOCHK, PACK or RELOAD commands.

Advantages of Temporary Tables/View:

- ***Raw Speed***

  TEMPORARY Tables/Views are lightening fast! There is no multi-user checking going on.

  Find a report that prints from a view whose performance is extremely slow, project a temporary table containing only the rows needed and drive the report from the temporary table. The report that takes 15-20 minutes to print might print in under a minute.

- ***Flexibility***

  Because of the speed, you can do things you would never do with permanent tables. If you have systems that do an extraordinary amount of massaging of data placed in a temporary table. The work would take far longer (we are talking 10-100 times) to accomplish with a permanent table. And with permanent tables, deleting your scratch work takes a great deal of time and each record must have a user id in it to work correctly. With temp tables you just reconnect the database and all the temp tables are gone, just like that.

- ***No database growth***

  The data in the temp tables is just not part of the RB2 files. The most important thing about TEMP tables is that the actual data for a given user is written into a Scratch ($$$) file. With a REAL table that data is stored in the Data (RB2) File.

  Say for example you are using a Real Table and you start with a 100Meg RB2 and add 5Meg of "temporary data" to a database... then drop the working table... Now your RB2 is 105Meg... then run the procedure again and you have 110Meg... then run the procedure again and 115Meg... then 120Meg... and so on... Then Pack / Reload and NOW you're back to 100Meg.

  On the other hand if you use TEMP tables your RB2 doesn't grow at all. Then to "PACK" or "RELOAD" you drop the TEMP table (or DISCONNECT) and R:BASE deletes the $$$ file.

  Views don't really make much difference since the only thing that would be stored in the database itself (and they still might be with Temp Views) would be the structure... everything else is generated when you actually use the view.

- ***Independability***

  TEMPorary Table(s)/View(s) are specific to each session of R:BASE and that specific user.

  For example, five different users or sessions, can create the same temporary table with the same name and not interfere with the others. Only the user that created the table can see/use it. So what it means is that 5 different uses can be using a running a report on the TEMPORARY table/view and all 5 users have different data in the table.

  New Sales Order option in Running R:BASE Your Way! (Part 1 - Part 10), sample applications bundled with R:BASE 7.1 and 7.5, demonstrate the typical use of this feature.

- ***Usability***

  You can treat the TEMPORARY table/view as a regular table. You can create Forms, Reports and Labels based upon the TEMPORARY table/view.

  A powerful use of temporary tables is to PROJECT or CREATE a temporary table to collect (LOAD) data and allow easy editing prior to an insert. Since each session of R:BASE will project/create its own private temporary table (of the same name) this is an ideal solution, say for collecting some accounting data prior to allowing the user to post the transaction to the formal journal tables. As soon as the insert is done, a DISCONNECT/CONNECT will eliminate the temporary table and you are ready for next time.

  Temporary tables are great when you are trying to take a huge vertical table with hundreds of thousands of rows and farm it out to some aggregate tables.

  Sometimes when you need to insert row(s) into a table based on rows in the table, (the where clause cannot refer to the same table for the insert) You may project a temp table of the correct where conditions and insert where column in permanent table in (select column from temp table). You can do all kinds of variations of this one, such as using existing rows as a template which you house in a temp table, edit for the new values and re-insert back to the permanent table.

Disadvantages of Temporary Tables/View:

- Temporary tables/views advantage is also their disadvantage. They are not permanent. any data you wish to be persistent must go in real tables/views.