**Overview of Temporary Tables (Doc ID 68098.1)**

Introduction  
~~~~~~~~~~~~  
This is an overview of TEMPORARY TABLES introduced in Oracle8i. This   
new feature allows temporary tables to be created automatically in a   
users temporary tablespace.  
  
Syntax  
~~~~~~  
 CREATE GLOBAL TEMPORARY TABLE tablename ( columns )  
 [ ON COMMIT PRESERVE | DELETE ROWS ]  
  
 The default option is to delete rows on commit.  
  
What Happens  
~~~~~~~~~~~~  
 When you create a GLOBAL TEMPORARY table a dictionary definition of  
 the table is created. As soon as the table gets populated (on the first  
 INSERT or at creation time for CTAS operations) a temporary segment is  
 created in the users default TEMPORARY tablespace location. This temporary  
 segments contents are just like a normal table.  
  
 Different sessions using the same GLOBAL TEMPORARY table get allocated  
 different temporary segments. The temporary segments are cleaned up  
 automatically at session end or transaction end depending on the specified  
 duration (ON COMMIT PRESERVE ROWS or ON COMMIT DELETE ROWS).  
   
 Apart from the data visibility temporary tables can be used like ordinary  
 tables for most operations.  
  
  
   
Characteristics  
~~~~~~~~~~~~~~~  
   
1. Data exists only for the duration of either the session or   
 transaction.  
  
 This can be specified in the create table command.  
 For example:  
  
 SQL> Create global temporary table emp\_temp(eno number)   
 on commit delete rows;  
  
 - OR -  
  
 SQL> Create global temporary table emp\_temp(eno number)   
 on commit preserve rows;  
  
  
 ON COMMIT DELETE ROWS indicates a transaction level duration and  
 PRESERVE indicates a session level duration.   
   
2. Data is visible only at session or transaction level. Multiple   
 users using the same temporary table can see the definition  
 of the table and their own data segment and nothing else.  
  
3. Indexes, triggers and views can be created on these tables.  
  
4. If an Index is created on temporary tables then it MUST be created   
 when the table is empty - ie: When there are NO temporary segments for   
 incarnations of the table. Indexes are implemented as separate   
 temporary segments.  
  
5. No redo is generated for operations on the temporary table itself BUT  
 undo is generated. Redo \*IS\* generated for the undo so temporary tables  
 do indirectly generate redo.  
  
6. The keyword GLOBAL indicates the table definition can be viewed  
 by anybody with sufficient privileges - ie:using the same rules  
 that apply to normal user tables. Currently only GLOBAL TEMPORARY  
 tables are supported.  
  
7. TRUNCATE operations truncate only the current session's incarnation  
 of the table.  
  
8. You can only export or import the definition not the data.  
  
9. Segments get created only on the first insert (or CTAS) operation.  
  
  
Drawbacks  
~~~~~~~~~  
  
1. The table definition is not dropped automatically.   
  
2. Only GLOBAL tables are supported right now, not local ones.   
   
3. Can perform DDL only when no session is bound to it.   
  
4. There is no underlying support for STATISTICS on GLOBAL   
 TEMPORARY tables so CBO (Cost Based Optimizer) has no statistical  
 information to help determine an execution plan.  
 NB: "ANALYZE TABLE COMPUTE/ESTIMATE STATISTICS" returns success   
 even though no statistics are gathered in 8i. 9i and higher   
 provides support for STATISTICS.  
  
In order to gather statistics follow the directions provided in   
Note:351190.1 - How to Create Statistics on Global Temporary Tables  
  
   
  
Constraints  
~~~~~~~~~~~  
  
Constraints can be implemented on the table either at the session or   
transaction level depending on the scope of the temporary table and  
are not for the entire table even though the constraint is defined   
for the entire table.  
  
If there is a primary key or unique key constraint, it is applicable only at  
either session or transaction leve i.e. two users can enter the same values  
into the table from different sessions even though you have a primary / unique  
key in place for the entire table (if the scope is the session )  
  
In the case of a transaction level temporary table, the same values   
can be entered from different transactions.