PL/SQL Transaction Commit, Rollback, Savepoint, Autocommit, Set Transaction

Oracle PL/SQL transaction oriented language. Oracle transactions provide a data integrity. PL/SQL transaction is a series of SQL data manipulation statements that are work logical unit. Transaction is an atomic unit all changes either committed or rollback.

At the end of the transaction that makes database changes, Oracle makes all the changes permanent save or may be undone. If your program fails in the middle of a transaction, Oracle detect the error and rollback the transaction and restoring the database.

You can use the COMMIT, ROLLBACK, SAVEPOINT, and SET TRANSACTION command to control the transaction.

1. [COMMIT](http://www.way2tutorial.com/plsql/plsql_transaction.php#commit) : COMMIT command to make changes permanent save to a database during the current transaction.
2. [ROLLBACK](http://www.way2tutorial.com/plsql/plsql_transaction.php#rollback) : ROLLBACK command execute at the end of current transaction and undo/undone any changes made since the begin transaction.
3. [SAVEPOINT](http://www.way2tutorial.com/plsql/plsql_transaction.php#savepoint) : SAVEPOINT command save the current point with the unique name in the processing of a transaction.
4. [AUTOCOMMIT](http://www.way2tutorial.com/plsql/plsql_transaction.php#autocommit) : Set AUTOCOMMIT ON to execute COMMIT Statement automatically.
5. [SET TRANSACTION](http://www.way2tutorial.com/plsql/plsql_transaction.php#set_transaction) : PL/SQL SET TRANSACTION command set the transaction properties such as read-write/read only access.



### Commit

The COMMIT statement to make changes permanent save to a database during the current transaction and visible to other users,

Commit Syntax :

SQL>COMMIT [COMMENT "comment text"];

Commit comments are only supported for backward compatibility. In a future release commit comment will come to a deprecated.

Commit Example :

SQL>BEGIN

UPDATE emp\_information SET emp\_dept='Web Developer'

WHERE emp\_name='Saulin';

COMMIT;

END;

/

### Rollback

The ROLLBACK statement ends the current transaction and undoes any changes made during that transaction. If you make a mistake, such as deleting the wrong row from a table, a rollback restores the original data. If you cannot finish a transaction because an exception is raised or a SQL statement fails, a rollback lets you take corrective action and perhaps start over.

ROLLBACK Syntax:

SQL>ROLLBACK [To SAVEPOINT\_NAME];

ROLLBACK Example :

SQL>DECLARE

emp\_id emp.empno%TYPE;

BEGIN

SAVEPOINT dup\_found;

UPDATE emp SET eno=1

WHERE empname = 'Forbs ross'

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

ROLLBACK TO dup\_found;

END;

/

Above example statement is exception raised because eno = 1 is already so DUP\_ON\_INDEX exception rise and rollback to the dup\_found savepoint named.

### Savepoint

SAVEPOINT savepoint\_names marks the current point in the processing of a transaction. Savepoints let you rollback part of a transaction instead of the whole transaction.

SAVEPOINT Syntax :

SQL>SAVEPOINT SAVEPOINT\_NAME;

SAVEPOINT Example :

SQL>DECLARE

emp\_id emp.empno%TYPE;

BEGIN

SAVEPOINT dup\_found;

UPDATE emp SET eno=1

WHERE empname = 'Forbs ross'

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

ROLLBACK TO dup\_found;

END;

/

### Autocommit

No need to execute COMMIT statement every time. You just set AUTOCOMMIT ON to execute COMMIT Statement automatically. It's automatic execute for each DML statement. set auto commit on using following statement,

AUTOCOMMIT Example :

SQL>SET AUTOCOMMIT ON;

You can also set auto commit off,

SQL>SET AUTOCOMMIT OFF;

### Set Transaction

SET TRANSACTION statement is use to set transaction are read-only or both read write. you can also assign transaction name.

SET TRANSACTION Syntax :

SQL>SET TRANSACTION [ READ ONLY | READ WRITE ]

[ NAME 'transaction\_name' ];

Set transaction name using the SET TRANSACTION [...] NAME statement before you start the transaction.

SET TRANSACTION Example :

SQL>SET TRANSACTION READ WRITE NAME 'tran\_exp';