# Transaction Control Language in Oracle

## What is a Transaction?

The transaction allows us to group a set of related tasks as one logical unit and all of these sets of related tasks are either get committed or get rollback if there is an error. In technical words, we can say that a transaction in Oracle is a set of SQL statements (mostly DML Statements) that should be executed as one unit.

So, a transaction in Oracle ensures that either all of the command succeeds or none of the commands succeeds. If one of the commands in the transaction fails, then all of the commands fail and any data that is modified in the database is rolled back. If all the commands are executed successfully, then the modification made to the database are committed.

## What is Transaction Management in Oracle?

The process of combining sets of inter-related Operations into a single unit and executing those operations by applying the do everything or do-nothing principle is called transaction management in Oracle. For example, the transfer money task is the combination of two operations

1. **Withdraw money from the Senders account**
2. **Deposit Money into the Receivers account.**

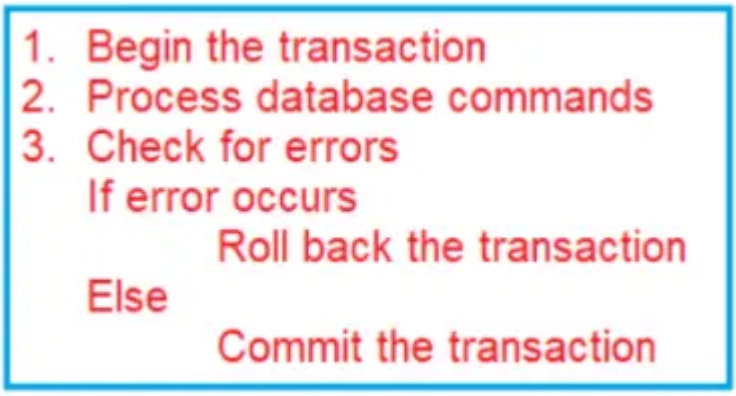
We need to execute the above two operations by applying the do-everything or do-nothing principle is nothing but transaction management. So, every transaction has two boundaries

1. **Beginning**
2. **Ending**

And controlling the boundaries of a transaction is nothing but transaction management.

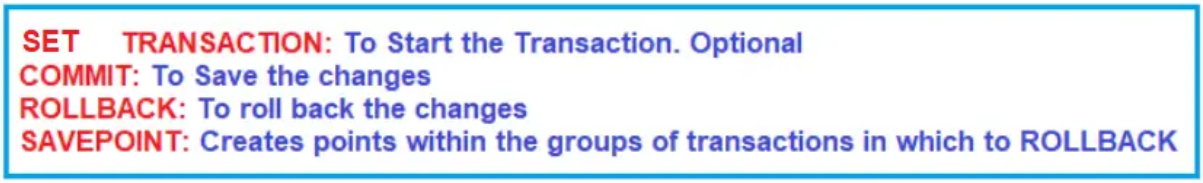
## How to implement Transaction Management in Oracle?

In order to understand how to implement transaction management in Oracle, please have a look at the below image. As you can see in the below image, transaction management involves three steps. First, we need to begin the transaction. Then we need to write the SQL Statements which we want to execute as a single unit. In the third step, we need to check for errors. If there is any error i.e. any of the SQL statements fails, then roll back the transaction (any data that is modified in the database will be rollback) else commit the transaction so that the data is saved permanently to the database.




## Transaction Control Language (TCL) Commands in Oracle:

A transaction is a unit of work that is performed against a database. If we are inserting / updating / deleting data to/from a table then we are performing a transaction on a table. To manage the transactions on database tables in Oracle, we are provided with transaction control language (TCL) commands. Transaction Control Language provides the following commands which we can use to implement transactions in Oracle.




1. **SET TRANSACTION:** It indicates that the transaction is started and is optional.
2. **COMMIT**: It indicates that the transaction was completed successfully and all the DML performed since the start of the transaction are committed to the database as well as frees the resources held by the transaction.
3. **ROLLBACK**: It will roll back the data to its previous state.

4. **SAVEPOINT**: This is used for dividing or breaking a transaction into multiple units so that the user has a chance of rolling back a transaction up to a point or location. It creates points within the groups of transactions in which to ROLLBACK.

```
CREATE TABLE Product
(
 ProductId INT PRIMARY KEY,
 ProductName VARCHAR2(20),
 Price INT,
 Quantity INT
);

 INSERT INTO Product VALUES(1001, 'Product-1', 1000, 100);
 INSERT INTO Product VALUES(1002, 'Product-2', 2000, 150);
 INSERT INTO Product VALUES(1003, 'Product-3', 3000, 200);
 INSERT INTO Product VALUES(1004, 'Product-4', 4000, 250);
```

```
INSERT INTO Product VALUES (1005, 'Product-5', 5000, 300);
UPDATE Product SET Price =3500 WHERE ProductID = 1003;
COMMIT;
```

## What is the need for the ROLLBACK command in Oracle?

The Rollback command in Oracle is used to undo the transactions that have not already been saved permanently to the database and get back to the initial state from where the transaction was started. So, if you want to restore the data into its previous state, then you need to use the ROLLBACK command at any time after the DML queries have been written but remember once the COMMIT statement has been executed then you cannot ROLLBACK the data. So, this command is used to cancel transactions. But once a transaction is committed then we cannot “rollback(cancel)”.

```
INSERT INTO Product VALUES(1006,'Product-6',6000, 300);
UPDATE Product SET Price =550 WHERE ProductID = 1005;
DELETE FROM Product WHERE ProductID = 1004;
  SELECT * FROM product;
```

`ROLLBACK;`

```
SET TRANSACTION READ WRITE;
INSERT INTO Product VALUES (1006,'Product-6',6000, 300);
UPDATE Product SET Price =550 WHERE ProductID = 1005;
```

```
DELETE FROM Product WHERE ProductID = 1004;
COMMIT;
```

## SET TRANSACTION TCL Command in Oracle:

Using **SET TRANSACTION** in Oracle to begin a transaction is optional. A new transaction begins
implicitly with the first DML statement that you execute after you make a database connection or with the first DML statement that you execute following a COMMIT or a ROLLBACK. You need to use SET TRANSACTION only when you want transaction attributes such as READ ONLY that are not the default.

The SET TRANSACTION marks the beginning of a transaction. Any changes you make to your data following the beginning of a transaction are not made permanent until you issue a COMMIT. Furthermore, those changes are not visible to other users until you've issued a COMMIT.

```
SET TRANSACTION READ WRITE;
INSERT INTO Product VALUES (1006,'Product-6',6000, 300);
UPDATE Product SET Price =550 WHERE ProductID = 1005;
DELETE FROM Product WHERE ProductID = 1004;
COMMIT;
```

## Transaction Types in Oracle:

1. **SET TRANSACTION READ WRITE:** It is the default transaction type specified as a read/write
   transaction. It allows us to issue statements such as UPDATE and DELETE.
2. **SET TRANSACTION READ ONLY:** It is a read-only transaction that does not allow the UPDATE and the DELETE DML statements.

## AUTOCOMMIT:

You don't require to execute a COMMIT statement every time whenever an INSERT, UPDATE or DELETE command is executed. You just need to set AUTOCOMMIT ON to execute COMMIT Statement automatically in Oracle. It is going to be executed for each DML statement. set auto-commit on using the following syntax.

**Syntax: SET AUTOCOMMIT ON;**

You can also set auto-commit off using the following syntax.

**Syntax: SET AUTOCOMMIT OFF;**