Prerequisite :

A transaction is a unit of work that is performed against a database. Transactions are units or sequences of work accomplished in a logical order, whether in a manual fashion by a user or automatically by some sort of a database program.

A transaction is the propagation of one or more changes to the database. For example, if you are creating a record or updating a record or deleting a record from the table, then you are performing transaction on the table. It is important to control transactions to ensure data integrity and to handle database errors.

Properties of Transactions:

Transactions have the following four standard properties, usually referred to by the acronym ACID:

- **Atomicity:** ensures that all operations within the work unit are completed successfully; otherwise, the transaction is aborted at the point of failure, and previous operations are rolled back to their former state.
- **Consistency:** ensures that the database properly changes states upon a successfully committed transaction.
- **Isolation:** enables transactions to operate independently of and transparent to each other.
- **Durability:** ensures that the result or effect of a committed transaction persists in case of a system failure.

Transaction Control:

There are following commands used to control transactions:

- **COMMIT:** to save the changes.
- **ROLLBACK:** to rollback the changes.
- **SAVEPOINT:** creates points within groups of transactions in which to ROLLBACK
- **SET TRANSACTION:** Places a name on a transaction.

The COMMIT Command:

The COMMIT command is the transactional command used to save changes invoked by a transaction to the database.

The COMMIT command saves all transactions to the database since the last COMMIT or ROLLBACK command.

The SAVEPOINT Command:

A SAVEPOINT is a point in a transaction when you can roll the transaction back to a certain point without rolling back the entire transaction.

The syntax for SAVEPOINT command is as follows:

```
SAVEPOINT SAVEPOINT_NAME;
```

This command serves only in the creation of a SAVEPOINT among transactional statements. The ROLLBACK command is used to undo a group of transactions.

The syntax for rolling back to a SAVEPOINT is as follows:

```
ROLLBACK TO SAVEPOINT_NAME;
```

Example :-

```
SQL> select * from Account_master;

ACCOU CURRENT_BALANCE
----- ---------------
CA10             3000
CA11             2000
CA12             1000

SQL> select * from Transaction_Master;

no rows selected
```

In the example given we have withdrawn an amount of 1000. Then we have created savepoint no_update . Then deposite an amount of 140000 if the total amount exceeds 200000 then we have to rollback means undo the action of depositing money using ROLLBACK TO SAVEPOINT NO_UPDATE otherwise save the changes using COMMIT.

1. In this amount is less then 200000 even after depositing 140000 hence rollback action will not perform

```
BEGIN
insert into Transaction_Master VALUES('T1','CA10','26/sep/16',1000,'Withdraw');

update Account_master set current_balance=current_balance-1000 where Account_no='CA10';
SAVEPOINT no_update;

insert into Transaction_Master VALUES('T2','CA10','27/sep/16',140000,'deposite');
update Account_master set current_balance=current_balance+140000 where Account_no='CA10';

select SUM(Current_balance) into mbal from Account_master;
if(mbal>200000) then
rollback to SAVEPOINT no_update;
END IF;
COMMIT;
END;SP2-0158: unknown SET option "SERVER"
SQL>   2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17
 18  /

PL/SQL procedure successfully completed.

SQL> select * from Transaction_Master;

TRA ACC_N DATE_OF_T     AMOUNT DESCRIPTION
--- ----- --------- ---------- --------------------
T1  CA10  26-SEP-16       1000 Withdraw
T2  CA10  27-SEP-16     140000 deposite

SQL> select * from Account_master;

ACCOU CURRENT_BALANCE
----- ---------------
CA10           142000
CA11             2000
CA12             1000
```

2. In this amount exceeds limit of 200000 hence rollback action will perform.

```
SQL>
SET SERVER OUTPUT ON;
Declare
mbal number(8,2);
BEGIN
insert into Transaction_Master VALUES('T3','CA11','28/sep/16',1000,'Withdraw');

update Account_master set current_balance=current_balance-1000 where Account_no='CA11';
SAVEPOINT no_update;

insert into Transaction_Master VALUES('T4','CA11','29/sep/16',140000,'deposite');
update Account_master set current_balance=current_balance+140000 where Account_no='CA11';

select SUM(Current_balance) into mbal from Account_master;
if(mbal>200000) then
rollback to SAVEPOINT no_update;
END IF;
COMMIT;
END;SQL> SP2-0158: unknown SET option "SERVER"
SQL>   2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17
 18  /

PL/SQL procedure successfully completed.

SQL> select * from Transaction_Master;

TRA ACC_N DATE_OF_T     AMOUNT DESCRIPTION
--- ----- --------- ---------- --------------------
T1  CA10  26-SEP-16       1000 Withdraw
T2  CA10  27-SEP-16     140000 deposite
T3  CA11  28-SEP-16       1000 Withdraw

SQL> select * from Account_master;

ACCOU CURRENT_BALANCE
----- ---------------
CA10           142000
CA11             1000
CA12             1000
```