DML

INSERT

It inserts a record to a table.
Let us observe how it is done,

```
SQL>  INSERT INTO products
  2   values (1001, 'CAMERA' , 10, 'Digital') ;

1 row created.

SQL> INSERT INTO products
  2  values (1002, 'Laptop', 23, 'Dell') ;

1 row created.
```

This is how we insert values into a table. All characters and alpha-numeric characters(ex – 10023sdf78) must be enclosed in single quotes („ „) and each value must be separated by comma. Also we must be careful in entering the data without violating the primary key, foreign key , unique constraints.

Now let us see the table in which the data in has been inserted,

```
SQL> select * from products ;

    PRODID PRODNAME   QTY_AVAILABLE DESCRIPTION
---------- ---------- ------------- --------------------
      1001 CAMERA                 10 Digital
      1002 Laptop                 23 Dell
```

Now, let us insert data into the table orders in which a foreign key is referencing primary key,

```
SQL> INSERT INTO orders
  2  values (1001, 9001, 2, 9867.1, sysdate ) ;

1 row created.
```

Here, we see that 1001 is the same prodid as of the earlier table. Sysdate – it displays the current date set in the system .

```
SQL> INSERT INTO orders
  2  values (1002, 9023, 2, 98756.23, ' 02 - Oct - 2010 ' ) ;

1 row created.
```

Now, let us see the table,

```
SQL> select * from orders ;

    PRODID    ORDERID   QTY_SOLD      PRICE ORDER_DT
---------- ---------- ---------- ---------- ---------
      1001       9001          2     9867.1 06-APR-11
      1002       9023          2   98756.23 02-OCT-10
```

Another way of inserting data into the table is shown below,

```
SQL> INSERT INTO orders (prodid, orderid, qty_sold, price, order_dt)
  2  values (1002, 99, 7, 23678.9, '02 - Oct - 1987' ) ;

1 row created.
```

Now, let us see the table,

```
SQL> select * from orders ;

    PRODID    ORDERID   QTY_SOLD      PRICE ORDER_DT
---------- ---------- ---------- ---------- ---------
      1001       9001          2     9867.1 06-APR-11
      1002       9023          2   98756.23 02-OCT-10
      1002         99          7    23678.9 02-OCT-87
```

UPDATE :-

It updates one or more records.

For ex – 1) Let us update salary by increasing it by Rs200 and also give commission of Rs100 where empno = 7369.

```
SQL> select * from emp ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.
```

Now, let us update the said record as shown below,

```
SQL> update emp set sal = sal + 200, comm = 100 where empno = 7369 ;

1 row updated.
```

Let us verify if the record has been updated,

```
SQL> select * from emp ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 17-DEC-80       1000        100         20
      7499 ALLEN      SALESMAN         7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER          7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER          7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER          7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT             17-NOV-81       5000                    10
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK            7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK            7698 03-DEC-81        950                    30
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK            7782 23-JAN-82       1300                    10

14 rows selected.
```

Thus, the record(empno – 7369) has been updated.

2) Increase all salary by 10%

```
SQL> update emp set sal = sal + sal * 0.1 ;

14 rows updated.
```

Let us verify it,

```
SQL> select * from emp ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 17-DEC-80       1100        100         20
      7499 ALLEN      SALESMAN         7698 20-FEB-81       1760        300         30
      7521 WARD       SALESMAN         7698 22-FEB-81       1375        500         30
      7566 JONES      MANAGER          7839 02-APR-81     3272.5                    20
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1375       1400         30
      7698 BLAKE      MANAGER          7839 01-MAY-81       3135                    30
      7782 CLARK      MANAGER          7839 09-JUN-81       2695                    10
      7788 SCOTT      ANALYST          7566 19-APR-87       3300                    20
      7839 KING       PRESIDENT             17-NOV-81       5500                    10
      7844 TURNER     SALESMAN         7698 08-SEP-81       1650          0         30
      7876 ADAMS      CLERK            7788 23-MAY-87       1210                    20
      7900 JAMES      CLERK            7698 03-DEC-81       1045                    30
      7902 FORD       ANALYST          7566 03-DEC-81       3300                    20
      7934 MILLER     CLERK            7782 23-JAN-82       1430                    10

14 rows selected.
```

DELETE

It deletes one / some / all the records.

Let us create a table test from table emp – and see how to delete 1 record and how to delete all records from it,

```
SQL> select * from test ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

```
14 rows selected.
```

Thus, we have created the table test.

```
SQL> delete from test where empno = 7934 ;

1 row deleted.
```

Thus 1 row, „miller" has been deleted.

```
SQL> select * from test ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20

```
13 rows selected.
```

Thus, the deletion has been confirmed.

TCL

Any DML change on a table is not a permanent one.
We need to save the DML changes in order to make it permanent
We can also undo (ignore) the same DML changes on a table.

The DDL changes cannot be undone as they are implicitly saved.

ROLLBACK

It undoes the DML changes performed on a table.
Let us see in the below example how rollback works,

```
SQL> delete from emp ;

14 rows deleted.

SQL> select * from emp ;

no rows selected
```

Let us delete the employee table. When we perform select operation on emp, we can see that all the rows have been deleted.
We now perform the rollback operation,

```
SQL> rollback ;

Rollback complete.
```

Now let us perform the select operation,

```
SQL> select * from emp ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.
```

Thus performing the rollback operation, we can retrieve all the records which had been deleted.

COMMIT

It saves the DML changes permanently to the database.

Committing after rollback & vice versa will not have any effect

Let us explain the above statement with an example,

```
SQL> select * from test ;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

SQL> delete from test ;

4 rows deleted.

SQL> select * from test ;

no rows selected

SQL> rollback ;

Rollback complete.

SQL> commit  ;

Commit complete.

SQL> select * from test ;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
```

We can see that commit has no effect after rollback operation.

```
SQL> select * from test ;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

SQL> delete from test ;

4 rows deleted.

SQL> commit ;

Commit complete.

SQL> rollback ;

Rollback complete.

SQL> select * from test ;

no rows selected
```

Thus, from above – we can see that rollback has no effect after commit operation.

During an abnormal exit – i.e, shutdown or if the SQL window is closed by mouse click – then all the DML"s will be rolled back automatically.

During a normal exit – exit ; - all the DML"s will be auto-committed – and there will be no rollback.

Ex – 1) INSERT
UPDATE
ALTER
DELETE
ROLLBACK

When we perform the following operations in the same order for a table – then INSERT, UPDATE will be committed – because ALTER is a DDL – and thus all the DML"s above it will also be committed – because DDL operations cannot be undone.
Here – only DELETE will be rolled back because it"s a DML.

2) INSERT
UPDATE
DELETE
ROLLBACK

Here, all are rolled back.

SAVEPOINT :

It is like a pointer (break-point) till where a DML will be rolled back.
Ex :-
Insert …

Save point x ;
Update …
Delete ..

Rollback to x ;
…
…

Here, only DELETE & UPDATE are rolled back.
INSERT is neither rolled back nor committed.

Assignments

1) Create the following tables

a) Table name :-
STUDENTS regno (PK)
name (NN)
semester
DOB Phone

b) Table name :-
BOOKS bookno (PK)
bname
author

c) Table name :- LIBRARY
regno (FK from students)
bookno (FK from books)
DOI –date of issue
DOR – date of return

2) Insert 5 records to each of these tables

3) Differentiate between,
a) Delete and Truncate
b) Truncate and Drop
c) Char and Varchar
d) Drop and Delete

Delete and Truncate

a) Delete – deletes whichever records we want to delete from the table Truncate – deletes all the records whether we want it or not

b) Delete – can be undone
Truncate – cannot be undone.

NOTE – The Primary Key created using more than 1 column is called as *composite primary key*. Ex – alter table lib
Add primary key (regno, bookno, DOI) ;