

EXPERIMENT-1

```
SQL> create table student(  
2  sid NUMBER,  
3  sname VARCHAR2(20),  
4  sage NUMBER,  
5  saddress VARCHAR2(20)  
6  );
```

Table created.

```
SQL> desc student
```

Name	Null?	Type
SID		NUMBER
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)

```
SQL> select * from student;
```

no rows selected

```
SQL> ALTER TABLE student ADD sphone NUMBER;
```

Table altered.

```
SQL> DESC STUDENT
```

Name	Null?	Type
SID		NUMBER
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)
SPHONE		NUMBER

```
SQL> ALTER TABLE student DROP COLUMN sphone;
```

Table altered.

```
SQL> desc student
```

Name	Null?	Type
SID		NUMBER
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)

```
SQL> ALTER TABLE student modify sid VARCHAR2(20);
```

Table altered.

```
SQL> desc student
```

Name	Null?	Type
SID		VARCHAR2(20)
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)

```
SQL> ALTER TABLE student RENAME COLUMN sid to rollno;
```

Table altered.

```
SQL> desc student
```

Name	Null?	Type
ROLLNO		VARCHAR2(20)
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)

```
SQL> ALTER TABLE student RENAME to students;
```

Table altered.

```
SQL> desc students
```

Name	Null?	Type
ROLLNO		VARCHAR2(20)
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)

```
SQL> ALTER TABLE students ADD PRIMARY KEY(rollno);
```

Table altered.

```
SQL> desc students
```

Name	Null?	Type
ROLLNO	NOT NULL	VARCHAR2(20)
SNAME		VARCHAR2(20)
SAGE		NUMBER
SADDRESS		VARCHAR2(20)

```
SQL> create table std(  
2  sid NUMBER,  
3  sname VARCHAR2(10),  
4  AGE INT  
5  );
```

Table created.

```
SQL> DROP TABLE STD;
```

Table dropped.

```
SQL> truncate table students;
```

Table truncated.

```
SQL> select * from students;
```

no rows selected

EXPERIMENT_2

```
SQL> CREATE TABLE employee(  
  2  eid NUMBER,  
  3  ename VARCHAR2(20),  
  4  eage INT,  
  5  esalary NUMBER  
  6  );
```

Table created.

```
SQL> INSERT INTO employee  
  2  VALUES(1,'HARSHA',18,50000);
```

1 row created.

```
SQL> INSERT INTO employee  
  2  VALUES(2,'ARUN',19,60000);
```

1 row created.

```
SQL> INSERT INTO employee  
  2  VALUES(3,'DINESH',21,61000);
```

1 row created.

```
SQL> INSERT INTO employee  
  2  VALUES(4,'NIVAS',20,51000);
```

1 row created.

EXPERIMENT_2

```
SQL> SELECT * FROM employee;
```

EID	ENAME	EAGE	ESALARY
1	HARSHA	18	50000
2	ARUN	19	60000
3	DINESH	21	61000
4	NIVAS	20	51000

```
SQL> SELECT eid FROM employee;
```

EID
1
2
3
4

```
SQL> SELECT ename FROM employee;
```

ENAME
HARSHA
ARUN
DINESH
NIVAS

EXPERIMENT_2

```
SQL> SELECT eid,esalary FROM employee;
```

EID	ESALARY
1	50000
2	60000
3	61000
4	51000

```
SQL> SELECT eid,ename,esalary FROM employee;
```

EID	ENAME	ESALARY
1	HARSHA	50000
2	ARUN	60000
3	DINESH	61000
4	NIVAS	51000

```
SQL> SELECT * from EMPLOYEE WHERE esalary>50000;
```

EID	ENAME	EAGE	ESALARY
2	ARUN	19	60000
3	DINESH	21	61000
4	NIVAS	20	51000

EXPERIMENT_2

```
SQL> UPDATE employee SET esalary=esalary+500 WHERE eid=1;
```

```
1 row updated.
```

```
SQL> SELECT * FROM employee;
```

EID	ENAME	EAGE	ESALARY
1	HARSHA	18	50500
2	ARUN	19	60000
3	DINESH	21	61000
4	NIVAS	20	51000

```
SQL> DELETE FROM employee WHERE eid=4;
```

```
1 row deleted.
```

```
SQL> SELECT * FROM employee;
```

EID	ENAME	EAGE	ESALARY
1	HARSHA	18	50500
2	ARUN	19	60000
3	DINESH	21	61000

TO perform create view we want to create a table

```
SQL> CREATE TABLE person(  
 2 pid NUMBER NOT NULL,  
 3 pname VARCHAR2(20),  
 4 pcity VARCHAR2(20)  
 5 );  
  
Table created.
```

INSERT VALUES INTO THE TABLE

```
SQL> INSERT INTO person  
 2 VALUES(1,'ABC','ATP');  
  
1 row created.  
  
SQL> INSERT INTO person  
 2 VALUES(1,'ABC','AP');  
  
1 row created.  
  
SQL> INSERT INTO person  
 2 VALUES(3,'AC','AP');  
  
1 row created.  
  
SQL> INSERT INTO person  
 2 VALUES(4,'SDC','CSDP');  
  
1 row created.
```

CREATE VIEWS FOR THE ABOVE TABLE USING BELOW SYNTAX

SYNTAX:

CREATE VIEW VIEW_NAME AS SELECT COLUMNS FROM TABLE;

EX:

```
SQL> CREATE VIEW employee AS SELECT pid ,pname FROM person;  
  
View created.
```


To see the view we use the following keyword

SELECT * FROM employee

```
SQL> SELECT * FROM employee
2 ;
```

PID	PNAME
1	ABC
1	ABC
3	AC
4	SDC

To add views to the table we use the folloeing syntax:

INSERT INTO VIEW_NAME(COL1,COL2)VALUES(value_list);

EX:

```
SQL> insert into employee(pid,pname)
2 VALUES(5,'AC');

1 row created.
```

```
SQL> SELECT * FROM employee
2 ;
```

PID	PNAME
1	ABC
1	ABC
3	AC
4	SDC
5	AC

We observe that above table is updated.

TO delete the table

EX:

```
SQL> DELETE person;  
  
5 rows deleted.
```

When you delete the table the views and the table is also deleted.

```
SQL> select*from person  
2 ;  
  
no rows selected  
  
SQL> select*from employee  
2 ;  
  
no rows selected
```

EXPERIMENT_4

```
SQL> CREATE TABLE instructor(  
  2  id NUMBER PRIMARY KEY,  
  3  name VARCHAR2(10),  
  4  dep_name VARCHAR2(10),  
  5  salary NUMBER  
  6  );
```

Table created.

```
SQL> CREATE TABLE department(  
  2  did NUMBER PRIMARY KEY,  
  3  dname VARCHAR2(15),  
  4  building VARCHAR2(15),  
  5  budget NUMBER  
  6  );
```

Table created.

```
SQL> INSERT ALL  
  2  INTO instructor VALUES(1,'HARSHA','CSE',50000)  
  3  INTO instructor VALUES(2,'ARUN','CSE',55000)  
  4  INTO instructor VALUES(3,'DINESH','EEE',52000)  
  5  INTO instructor VALUES(4,'BASHA','ECE',42000)  
  6  INTO instructor VALUES(5,'SUMANTH','CSM',32000)  
  7  INTO department VALUES(1,'CSE','B',35000000)  
  8  INTO department VALUES(2,'ECE','A',1780000)  
  9  INTO department VALUES(3,'MECH','MAIN',1734000)  
 10  SELECT * FROM dual;
```

8 rows created.

```
SQL> select * from department;
```

	DID	DNAME	BUILDING	BUDGET
1	CSE	B	35000000	
2	ECE	A	1780000	
3	MECH	MAIN	1734000	

EXPERIMENT_4

```
SQL> select * from instructor;
```

ID	NAME	DEP_NAME	SALARY
1	HARSHA	CSE	50000
2	ARUN	CSE	55000
3	DINESH	EEE	52000
4	BASHA	ECE	42000
5	SUMANTH	CSM	32000

```
SQL> SELECT dep_name from instructor
2 UNION
3 SELECT dname from department;
```

```
DEP_NAME
```

```
-----
CSE
EEE
ECE
CSM
MECH
```

```
SQL> select dep_name from instructor
2 UNION ALL
3 select dname from department;
```

```
DEP_NAME
```

```
-----
CSE
CSE
EEE
ECE
CSM
CSE
ECE
MECH
```

```
8 rows selected.
```

EXPERIMENT_4

```
SQL> select dep_name from instructor
 2  INTERSECT
 3  select dname from department;
```

DEP_NAME

CSE

ECE

```
SQL> select dname from department
 2  MINUS
 3  select dep_name from instructor;
```

DNAME

MECH

```
SQL> select i.name,d.dname,d.budget from instructor i,department d;
```

NAME	DNAME	BUDGET
------	-------	--------

-----	-----	-----
-------	-------	-------

HARSHA	CSE	35000000
--------	-----	----------

ARUN	CSE	35000000
------	-----	----------

DINESH	CSE	35000000
--------	-----	----------

BASHA	CSE	35000000
-------	-----	----------

SUMANTH	CSE	35000000
---------	-----	----------

HARSHA	ECE	1780000
--------	-----	---------

ARUN	ECE	1780000
------	-----	---------

DINESH	ECE	1780000
--------	-----	---------

BASHA	ECE	1780000
-------	-----	---------

SUMANTH	ECE	1780000
---------	-----	---------

HARSHA	MECH	1734000
--------	------	---------

NAME	DNAME	BUDGET
------	-------	--------

-----	-----	-----
-------	-------	-------

ARUN	MECH	1734000
------	------	---------

DINESH	MECH	1734000
--------	------	---------

BASHA	MECH	1734000
-------	------	---------

SUMANTH	MECH	1734000
---------	------	---------

15 rows selected.

EXPERIMENT_5

```
SQL> create table instructors(  
  2  id NUMBER PRIMARY KEY,  
  3  name VARCHAR2(19),  
  4  salary NUMBER  
  5  );
```

Table created.

```
SQL> CREATE TABLE departments(  
  2  id NUMBER PRIMARY KEY,  
  3  dname VARCHAR2(10)  
  4  );
```

Table created.

```
SQL> INSERT ALL  
  2 INTO instructors VALUES(1,'HARSHA',80000)  
  3 INTO instructors VALUES(2,'ARUN',90000)  
  4 INTO instructors VALUES(3,'DINESH',70000)  
  5 INTO instructors VALUES(4,'BASHA',75000)  
  6 INTO departments VALUES(1,'CSE')  
  7 INTO departments VALUES(2,'EEE')  
  8 INTO departments VALUES(3,'ECE')  
  9 SELECT * FROM dual;
```

7 rows created.

EXPERIMENT_5

```
SQL> select * from instructors;
```

ID	NAME	SALARY
1	HARSHA	80000
2	ARUN	90000
3	DINESH	70000
4	BASHA	75000

```
SQL> select * from departments;
```

ID	DNAME
1	CSE
2	EEE
3	ECE

```
SQL> select * from instructors
2  WHERE
3  salary IS NULL;
```

ID	NAME	SALARY
4	BASHA	

```
SQL> select * from instructors
2  where
3  salary between 80000 and 90000;
```

ID	NAME	SALARY
1	HARSHA	80000
2	ARUN	90000

EXPERIMENT_5

```
SQL> select * from instructors
  2  where
  3  name like 'B%';
```

ID	NAME	SALARY
4	BASHA	

```
SQL> select * from instructors
  2  where
  3  salary IN(10000,80000,90000);
```

ID	NAME	SALARY
1	HARSHA	80000
2	ARUN	90000

```
SQL> select * from instructors
  2  where
  3  EXISTS(SELECT * FROM departments WHERE instructors.id=departments.id);
```

ID	NAME	SALARY
1	HARSHA	80000
2	ARUN	90000
3	DINESH	70000

EXPERIMENT-6

```
SQL> create table student(  
  2  rollno NUMBER PRIMARY KEY,  
  3  name VARCHAR2(20) NOT NULL,  
  4  dname VARCHAR2(10) NOT NULL  
  5  );
```

Table created.

```
SQL> CREATE TABLE building(  
  2  dname VARCHAR2(10),  
  3  bname VARCHAR2(10)  
  4  );
```

Table created.

```
SQL> INSERT ALL  
  2  INTO student VALUES(1,'harsha','cse')  
  3  INTO student VALUES(2,'basha','ece')  
  4  INTO student VALUES(3,'dinesh','eee')  
  5  INTO student VALUES(4,'hari','csd')  
  6  INTO building VALUES('cse','b')  
  7  INTO building VALUES('eee','a')  
  8  INTO building VALUES('csd','c')  
  9  select * from dual;
```

7 rows created.

```
SQL> select * from student;
```

ROLLNO	NAME	DNAME
1	harsha	cse
2	basha	ece
3	dinesh	eee
4	hari	csd

```
SQL> select * from building;
```

DNAME	BNAME
cse	b
eee	a
csd	c

```
SQL> select * from student
2 JOIN building ON
3 student.dname=building.dname;
```

ROLLNO	NAME	DNAME	DNAME	BNAME
1	harsha	cse	cse	b
3	dinesh	eee	eee	a
4	hari	csd	csd	c

```
SQL> select * from student JOIN building
2 USING(dname);
```

DNAME	ROLLNO	NAME	BNAME
cse	1	harsha	b
eee	3	dinesh	a
csd	4	hari	c

```
SQL> select * from student
2 LEFT OUTER JOIN building ON
3 student.dname=building.dname;
```

ROLLNO	NAME	DNAME	DNAME	BNAME
1	harsha	cse	cse	b
3	dinesh	eee	eee	a
4	hari	csd	csd	c
2	basha	ece		

```
SQL> select * from student
2 RIGHT OUTER JOIN building ON
3 student.dname=building.dname;
```

ROLLNO	NAME	DNAME	DNAME	BNAME
1	harsha	cse	cse	b
3	dinesh	eee	eee	a
4	hari	csd	csd	c

```
SQL> select * from student
2 FULL OUTER JOIN building ON
3 student.dname=building.dname;
```

ROLLNO	NAME	DNAME	DNAME	BNAME
1	harsha	cse	cse	b
2	basha	ece		
3	dinesh	eee	eee	a
4	hari	csd	csd	c

Experiment-5

Aggregate functons(min,max,count,sum,avg)

To perform the aggregate functions you need to create a table and insert values in it

For example we can take a table name called employee

```
SQL> select * from employee;
```

EID	ENAME	SALARY
1	raju	1000
2	dinesh	3000
3	arun	4000
4	harsha	6000

Min function

It is used to find the minimum value int the column of a table

For example we are performing the above aggregate functions in the example.

Syntax :

```
Select min(column_name) from table_name;
```

Ex:

```
Select min(salary) from table_name;
```

```
SQL> select min(salary) from employee;
```

```
MIN(SALARY)
-----
          1000
```

```
SQL>
```

Max function :

It is used to find the maximum value in the column of a table.

Syntax:

Select max(column_name) from table_name;

Ex:

Select max(salary) from employee;

```
SQL> select max(salary) from employee;

MAX(SALARY)
-----
          6000
```

Count function :

It is used to count the how many rows in the column of a table.

Syntax:

Select count(column_name) from table_name;

Ex:

Select count(salary) from employee;

```
SQL> select count(salary) from employee;

COUNT(SALARY)
-----
              4

SQL>
```

Sum function :

It is used to find the sum of the values in a row of a table.

Syntax :

Select sum(column_name) from table_name;

Ex:

Select sum(salary) from employee;

```
SQL> select sum(salary) from employee;

SUM(SALARY)
-----
      14000
```

Avg function :

It is used to find the average of the column of a table.

Syntax:

Select avg(column_name) from table_name;

Ex:

Select avg(salary) from employee;

```
SQL> select avg(salary) from employee;

AVG(SALARY)
-----
       3500
```

EXPERIMENT-8

```
SQL> CREATE TABLE name(  
  2  fname VARCHAR2(20) NOT NULL,  
  3  lname VARCHAR2(20) NOT NULL  
  4  );
```

Table created.

```
SQL> INSERT ALL  
  2  INTO name VALUES('Harsha','Reddy')  
  3  INTO name VALUES('Dinesh','Reddy')  
  4  INTO name VALUES('Arun','Naik')  
  5  INTO name VALUES('Syed','Basha')  
  6  select * from dual;
```

4 rows created.

```
SQL> select LOWER(fname) from name;
```

```
LOWER(FNAME)
```

```
-----
```

```
harsha
```

```
dinesh
```

```
arun
```

```
syed
```



```
SQL> select UPPER(fname) from name;
```

```
UPPER(FNAME)
```

```
-----
```

```
HARSHA
```

```
DINESH
```

```
ARUN
```

```
SYED
```

```
SQL> select INITCAP(fname) from name;
```

```
INITCAP(FNAME)
```

```
-----
```

```
Harsha
```

```
Dinesh
```

```
Arun
```

```
Syed
```

```
SQL> select CONCAT(fname,lname) from name;
```

```
CONCAT(FNAME,LNAME)
```

```
-----
```

```
HarshaReddy
```

```
DineshReddy
```

```
ArunNaik
```

```
SyedBasha
```

```
SQL> select SUBSTR(fname,1,3) from name;
```

```
SUBSTR(FNAME
```

```
-----
```

```
Har
```

```
Din
```

```
Aru
```

```
Sye
```

```
SQL> select LENGTH(fname) from name;
```

```
LENGTH(FNAME)
```

```
-----
```

```
6  
6  
4  
4
```

```
SQL> select INSTR(fname,'a') from name;
```

```
INSTR(FNAME,'A')
```

```
-----
```

```
2  
0  
0  
0
```

```
SQL> select TRIM(' ' from fname) from name;
```

```
TRIM(' 'FROMFNAME)
```

```
-----
```

```
Harsha  
Dinesh  
Arun  
Syed
```

```
SQL> select ROUND(11.231,2) from dual;
```

```
ROUND(11.231,2)
```

```
-----
```

```
11.23
```

```
SQL> select MOD(25,2) from dual;
```

```
MOD(25,2)
-----
          1
```

```
SQL> select SYSDATE FROM dual;
```

```
SYSDATE
-----
17-DEC-23
```

```
SQL> select MONTHS_BETWEEN(SYSDATE,'17-DEC-2025') FROM DUAL;
```

```
MONTHS_BETWEEN(SYSDATE,'17-DEC-2025')
-----
                                -24
```

```
SQL> SELECT ADD_MONTHS(SYSDATE,12) FROM DUAL;
```

```
ADD_MONTH
-----
17-DEC-24
```

```
SQL> SELECT NEXT_DAY(SYSDATE,'MONDAY') FROM DUAL;
```

```
NEXT_DAY(
-----
18-DEC-23
```

```
SQL> SELECT LAST_DAY(SYSDATE) FROM DUAL;
```

```
LAST_DAY(
-----
31-DEC-23
```

```
SQL> SELECT CURRENT_TIMESTAMP(3) FROM DUAL;
```

```
CURRENT_TIMESTAMP(3)
-----
17-DEC-23 10.07.42.234 AM +05:30
```

EXPERIMENT-

Primary key :

```
SQL> create table college(  
  2  id varchar2(10) PRIMARY KEY,  
  3  name varchar2(20),  
  4  branch varchar2(10),  
  5  section varchar2(10)  
  6  );
```

Table created.

```
SQL> desc college
```

Name	Null?	Type
ID	NOT NULL	VARCHAR2(10)
NAME		VARCHAR2(20)
BRANCH		VARCHAR2(10)
SECTION		VARCHAR2(10)

Foreign key :

```
SQL> create table marks(  
  2  id varchar2(10) PRIMARY KEY,  
  3  num NUMBER NOT NULL,  
  4  marks varchar2(20) REFERENCES college(id)  
  5  );
```

Table created.

```
SQL> desc marks
```

Name	Null?	Type
ID	NOT NULL	VARCHAR2(10)
NUM	NOT NULL	NUMBER
MARKS		VARCHAR2(20)

```
SQL>
```

EXPERIMENT-10

```
SQL> ED
Wrote file afiedt.buf

 1 DECLARE
 2 n NUMBER;
 3 fac NUMBER:=1;
 4 n1 NUMBER;
 5 BEGIN
 6 n:=&n;
 7 n1:=n;
 8 WHILE N>0 LOOP
 9 fac:=n*fac;
10 n:=n-1;
11 END LOOP;
12 DBMS_OUTPUT.PUT_LINE('The factorial of '||n1||' is '||fac);
13* END;
SQL> /
Enter value for n: 5
old 6: n:=&n;
new 6: n:=5;

PL/SQL procedure successfully completed.

SQL> SET SERVEROUT ON
SQL> SET VERIFY OFF
SQL> /
Enter value for n: 5
The factorial of 5 is 120

PL/SQL procedure successfully completed.
```

EXPERIMENT-11

```
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  n NUMBER;
  3  i NUMBER;
  4  temp NUMBER;
  5  BEGIN
  6  n:=&n;
  7  i:=2;
  8  temp:=1;
  9  FOR I IN 2..n/2
10  LOOP
11  IF MOD(n,i)=0
12  THEN
13  temp:=0;
14  EXIT;
15  END IF;
16  END LOOP;
17  IF temp=1
18  THEN
19  DBMS_OUTPUT.PUT_LINE(n||' is a prime number');
20  ELSE
21  DBMS_OUTPUT.PUT_LINE(n||' is not a prime number');
22  END IF;
23* END;
SQL> /
Enter value for n: 12
12 is not a prime number

PL/SQL procedure successfully completed.

SQL> SET SERVEROUT ON
SQL> SET VERIFY OFF
SQL> /
Enter value for n: 3
3 is a prime number

PL/SQL procedure successfully completed.
```

EXPERIMENT-12

```
SQL> ED
Wrote file afiedt.buf

 1 DECLARE
 2 first NUMBER:=0;
 3 second NUMBER:=1;
 4 temp NUMBER;
 5 n NUMBER;
 6 i NUMBER;
 7 BEGIN
 8 n:=&n;
 9 DBMS_OUTPUT.PUT_LINE('SERIES');
10 DBMS_OUTPUT.PUT_LINE(first);
11 DBMS_OUTPUT.PUT_LINE(second);
12 FOR i IN 2..n
13 LOOP
14 temp:=first+second;
15 first:=second;
16 second:=temp;
17 DBMS_OUTPUT.PUT_LINE(temp);
18 END LOOP;
19* END;
20 /
Enter value for n: 5
SERIES
0
1
1
2
3
5

PL/SQL procedure successfully completed.
```


EXPERIMENT-13

```
SQL> CREATE TABLE sailor(  
  2  id NUMBER PRIMARY KEY,  
  3  name VARCHAR2(20) NOT NULL  
  4  );
```

Table created.

```
SQL> ED  
Wrote file afiedt.buf  
  
  1  CREATE OR REPLACE PROCEDURE insertuser(id IN NUMBER,name IN VARCHAR2)  
  2  AS  
  3  BEGIN  
  4  INSERT INTO sailor VALUES(id,name);  
  5  DBMS_OUTPUT.PUT_LINE('Record inserted successfully');  
  6* END;  
  7  /
```

Procedure created.

```
SQL> ED  
Wrote file afiedt.buf  
  
  1  DECLARE  
  2  co NUMBER;  
  3  BEGIN  
  4  insertuser(26,'Harsha');  
  5  select count(*) INTO co FROM sailor;  
  6  DBMS_OUTPUT.PUT_LINE(co||' Record is inserted successfully');  
  7* END;  
SQL> /  
Record inserted successfully  
1 Record is inserted successfully  
  
PL/SQL procedure successfully completed.
```



```
SQL> ED
Wrote file afiedt.buf

 1 DECLARE
 2 co NUMBER;
 3 BEGIN
 4 insertuser(25,'Harsha');
 5 select count(*) INTO co FROM sailor;
 6 DBMS_OUTPUT.PUT_LINE(co||' Record is inserted successfully');
 7* END;
SQL> /
Record inserted successfully
2 Record is inserted successfully

PL/SQL procedure successfully completed.
```

EXPERIMENT-

```
SQL> CREATE TABLE branch(  
  2  id NUMBER PRIMARY KEY,  
  3  name VARCHAR2(20) NOT NULL,  
  4  strength NUMBER  
  5  );
```

Table created.

```
SQL> INSERT ALL  
  2  INTO branch VALUES(1,'CSE',144)  
  3  INTO branch VALUES(2,'CSD',140)  
  4  INTO branch VALUES(2,'EEE',120)  
  5  SELECT * FROM DUAL;
```

INSERT ALL

*

ERROR at line 1:

ORA-00001: unique constraint (C##526.SYS_C008329) violated

```
SQL> INSERT ALL  
  2  INTO branch VALUES(1,'CSE',144)  
  3  INTO branch VALUES(2,'CSD',140)  
  4  INTO branch VALUES(3,'EEE',120)  
  5  SELECT * FROM DUAL;
```

3 rows created.

```
SQL> SET SERVEROUT ON
SQL> SET VERIFY OFF
SQL> CREATE OR REPLACE FUNCTION totalstrength RETURN NUMBER
  2 AS
  3 total NUMBER:=0;
  4 BEGIN
  5 SELECT sum(strength) INTO total FROM branch;
  6 return total;
  7 END;
  8 /
```

Function created.

```
SQL> DECLARE
  2 answer NUMBER;
  3 BEGIN
  4 answer:=totalstrength();
  5 DBMS_OUTPUT.PUT_LINE('Total strength of students is '||answer);
  6 END;
  7 /
```

Total strength of students is 404

PL/SQL procedure successfully completed.

EXPERIMENT-15

```
SQL> CREATE TABLE instruct(  
  2  id NUMBER PRIMARY KEY,  
  3  name VARCHAR2(10) NOT NULL,  
  4  dname VARCHAR2(10) NOT NULL,  
  5  salary NUMBER CHECK(salary>10000)  
  6  );
```

Table created.

```
SQL> INSERT ALL  
  2  INTO instruct VALUES(1,'HARSHA','CSE',50000)  
  3  INTO instruct VALUES(2,'ARUN','CSE',60000)  
  4  INTO instruct VALUES(3,'BASHA','ECE',55000)  
  5  INTO instruct VALUES(4,'DINESH','EEE',65000)  
  6  SELECT * FROM DUAL;
```

4 rows created.

```
SQL> CREATE OR REPLACE TRIGGER display_changes  
  2  BEFORE UPDATE ON instruct  
  3  FOR EACH ROW  
  4  WHEN(NEW.ID=OLD.ID)  
  5  DECLARE  
  6  sal_diff number;  
  7  BEGIN  
  8  sal_diff:=:NEW.salary-:OLD.salary;  
  9  DBMS_OUTPUT.PUT_LINE('OLD SALARY: '||:OLD.salary);  
 10  DBMS_OUTPUT.PUT_LINE('NEW SALARY: '||:NEW.salary);  
 11  DBMS_OUTPUT.PUT_LINE('Salary difference : '||sal_diff);  
 12  END;  
 13  /
```

Trigger created.

```
SQL> DECLARE
  2  tot_rows NUMBER;
  3  BEGIN
  4  UPDATE instruct
  5  SET salary=salary*1.5;
  6  IF sql%notfound THEN
  7  DBMS_OUTPUT.PUT_LINE('no instructors updated');
  8  ELSIF sql%found THEN
  9  tot_rows:=sql%rowcount;
 10  DBMS_OUTPUT.PUT_LINE(tot_rows||' instructors updated');
 11  END IF;
 12  END;
 13  /
```

PL/SQL procedure successfully completed.

```
SQL> SET SERVEROUT ON
SQL> SET VERIFY OFF
SQL> /
OLD SALARY: 75000
NEW SALARY: 112500
Salary difference : 37500
OLD SALARY: 90000
NEW SALARY: 135000
Salary difference : 45000
OLD SALARY: 82500
NEW SALARY: 123750
Salary difference : 41250
OLD SALARY: 97500
NEW SALARY: 146250
Salary difference : 48750
4 instructors updated
```

PL/SQL procedure successfully completed.

EXPERIMENT-

```
SQL> CREATE TABLE customers(  
  2  id NUMBER PRIMARY KEY,  
  3  name VARCHAR2(20) NOT NULL,  
  4  age NUMBER NOT NULL,  
  5  salary NUMBER NOT NULL  
  6  );
```

Table created.

```
SQL> INSERT ALL  
  2  INTO customers VALUES(1,'HARSHA',18,50000)  
  3  INTO customers VALUES(2,'ARUN',19,60000)  
  4  INTO customers VALUES(3,'BASHA',19,65000)  
  5  INTO customers VALUES(4,'DINESH',20,55000)  
  6  SELECT * FROM DUAL;
```

4 rows created.

```
SQL> DECLARE  
  2  tot_rows NUMBER;  
  3  BEGIN  
  4  UPDATE customers SET salary=salary*1.5;  
  5  IF sql%notfound THEN  
  6  DBMS_OUTPUT.PUT_LINE('No customers updated');  
  7  ELSIF sql%found THEN  
  8  tot_rows :=sql%rowcount;  
  9  DBMS_OUTPUT.PUT_LINE(tot_rows||' customers updated');  
 10  END IF;  
 11  END;  
 12  /
```

4 customers updated

PL/SQL procedure successfully completed.


```
SQL> DECLARE
  2  c_id customers.id%type;
  3  c_name customers.name%type;
  4  c_age customers.age%type;
  5  CURSOR c_customers IS
  6  SELECT id,name,age FROM customers;
  7  BEGIN
  8  OPEN c_customers;
  9  LOOP
 10  FETCH c_customers INTO c_id,c_name,c_age;
 11  EXIT WHEN c_customers%notfound;
 12  DBMS_OUTPUT.PUT_LINE(c_id||' '||c_name||' '||c_age);
 13  END LOOP;
 14  CLOSE c_customers;
 15  END;
 16  /
1 HARSHA 18
2 ARUN 19
3 BASHA 19
4 DINESH 20

PL/SQL procedure successfully completed.
```