DATABASE MANAGEMENT SYSTEM LABORATORY

COURSE CODE: BCSE2073

Lab Manual

for

BACHELOR OF

Engineering & Technology

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA

UTTAR PRADESH

NAME-NEERAJ SINGH

SEC-6

21SCSE1011675

SUBMITTED TO-

SWATI SHARMA MA'AM

Sr. No.	Title of Lab Experiments
1.	Implement Data Definition language Statements.
2.	Implement Data Manipulation Statements.
3.	Implement SELECT command with different clauses.
4.	Implement various type of Integrity Constraints on database.
5.	Implement SINGLE ROW functions (Character, Numeric, Date functions) and GROUP functions (avg, count, max, min, sum).
6.	Implement various type of SET OPERATORS (Union, Intersect, Minus) and JOINS.
7.	Implement the concept of grouping of Data and Subqueries.
8.	Implement the concept of Data Control Language (DCL), Transaction Control Language (TCL).
9.	Implement Simple and Complex View.
10.	Write a PL/SQL block to satisfy some conditions by accepting input from the user.
11.	Write a PL/SQL block for greatest of three numbers using IF AND ELSEIF
12.	Write a PL/SQL block for summation of odd numbers using for LOOP
13.	Write a PL/SQL Procedure for GCD Numbers
14.	Write a PL/SQL Procedure for cursor implementation
15.	Write a PL/SQL block to implementation of factorial using function
Value Added Experiments	
16.	Create a Database for Banking Sector and implement various queries on it.
17.	Create a Database for Customer Sale/purchase and implement various queries on it.

EXPERIMENT DETAILS

Experiment 1

Title **Data Definition Language**

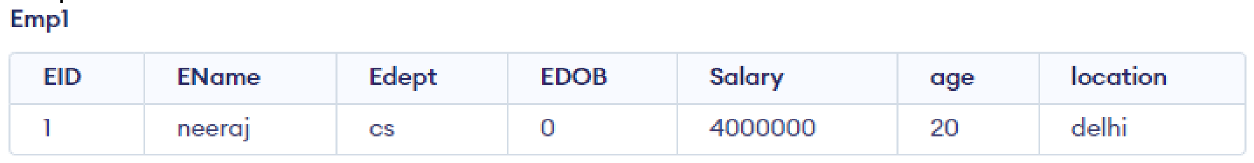
Objective **Study of Data Definition language commands. - Create table, Alter Table, Drop Table, Rename Table.**

Syntax

<u>CREATE TABLE</u>

```
CREATE TABLE Emp1 (
  EID int,
  EName Char,
  Edept char,
  EDOB date,
  Salary int
);
ALTER TABLE Emp1 ADD location int;
DROP TABLE Emp1;
RENAME Emp1 To Empolyee;
TRUNCATE TABLE  Empolyee;
```

output-



Emp1

EID	EName	Edept	EDOB	Salary	age	location
1	neeraj	cs	0	4000000	20	delhi

Experiment 2

Title **Data Manipulation Language Statements.**

Objective **Study of Data Manipulation Statements.**

Syntax

```
SELECT * FROM Student;
INSERT INTO Student (Stu_id, Stu_Name, Stu_Marks, Stu_Age) VALUES (104, Anmol, 89, 19);
```

UPDATE Product **SET** Product_Price = 80 **WHERE** Product_Id = 'P102' ;
DELETE FROM Product **WHERE** Product_Id = 'P202' ;

Student_ID	Student_Name	Student_Marks
BCA1001	Abhay	85
BCA1002	Anuj	75
BCA1003	Bheem	60
BCA1004	Ram	79
BCA1005	Sumit	80

Experiment 3

Title **SELECT Command**

Objective **Study of SELECT command with different clauses.**

Syntax **Syntax of SQL SELECT Statement:**

```
SELECT * FROM Student WHERE Stu_Marks = 80;
```

Student_ID	Student_Name	Student_Marks
BCA1001	Abhay	80
BCA1003	Bheem	80
BCA1005	Sumit	80

```
SELECT COUNT (Car_Name), Car_Price FROM Cars_Details GROUP BY Car_Price;
```

Count (Car_Name)	Car_Price
2	1000000
2	900000

```
SELECT * FROM Employee_Order ORDER BY Emp_Salary DESC;
```

Emp_Id	Emp_Name	Emp_Salary	Emp_City
204	Anuj	90000	Goa
203	Rashet	80000	Jaipur
205	Sumit	50000	Delhi

Experiment 4

Title **Keys**

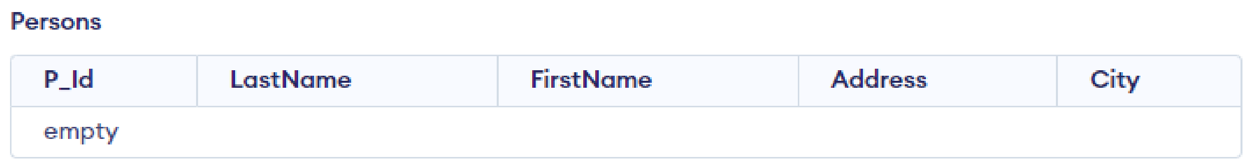
Objective **Study of various type of Integrity Constraints.**

Syntax

SQL CREATE TABLE + CONSTRAINT Syntax

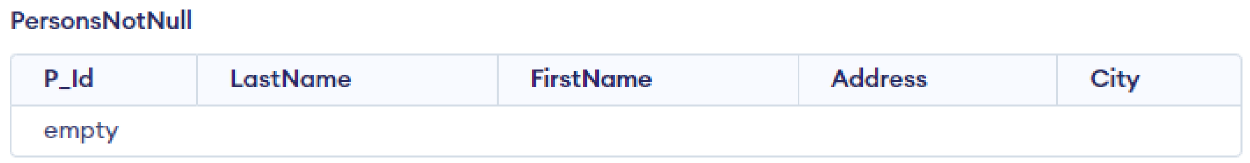
```
CREATE TABLE PersonsNotNull
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)

CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
PRIMARY KEY (P_Id)
)

CREATE TABLE Orderr
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
PRIMARY KEY (O_Id),
FOREIGN KEY (P_Id) REFERENCES Persons(P_Id)
)
```

output



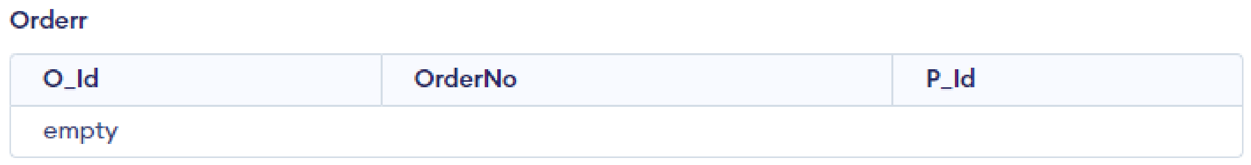
Persons

P_Id	LastName	FirstName	Address	City
empty				



PersonsNotNull

P_Id	LastName	FirstName	Address	City
empty				



Orderr

O_Id	OrderNo	P_Id
empty		

Experiment 5

Title: SINGLE ROW functions and Group functions

Objective: Study of SINGLE ROW functions (Character, Numeric, Date functions) and GROUP functions (avg, count, max, min, sum).

Syntax

```
SELECT first_name, last_name, salary, NVL (commission_pct,0)
FROM employees
WHERE rownum < 5;
```

output

```
FIRST_NAME     LAST_NAME       SALARY     NVL(COMMISSION_PCT,0)
-------------------- ------------------------- ---------- ---------------------
Steven            King                 24000              0
Neena             Kochhar                17000              0
Lex              De Haan               17000              0
Alexander          Hunold                 9000              0
```

Some of the commonly used aggregate functions are as below -

SUM([ALL | DISTINCT] expression)

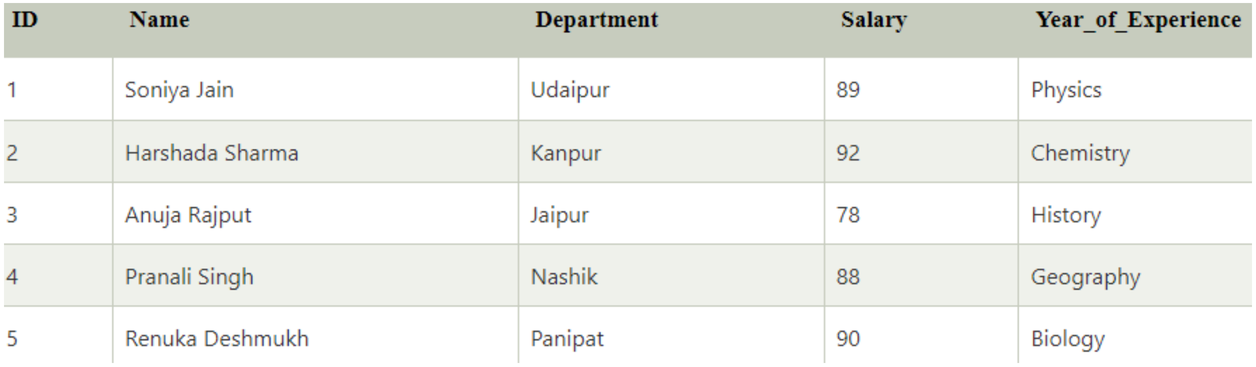
AVG([ALL | DISTINCT] expression)

COUNT([ALL | DISTINCT] expression)
COUNT(*)
MAX(expression)
MIN(expression)
Post Lab Assignment (If Any)

Experiment 6(a)

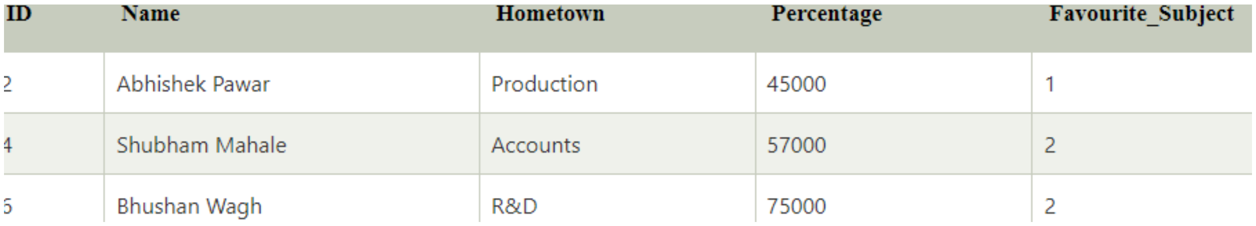
Title **SET Operators.**
Objective **Study of various type of SET OPERATORS (Union, Intersect, Minus) and Various type of JOINS.**
Syntax

```
mysql> SELECT *FROM t_students UNION SELECT *FROM t2_students;
```



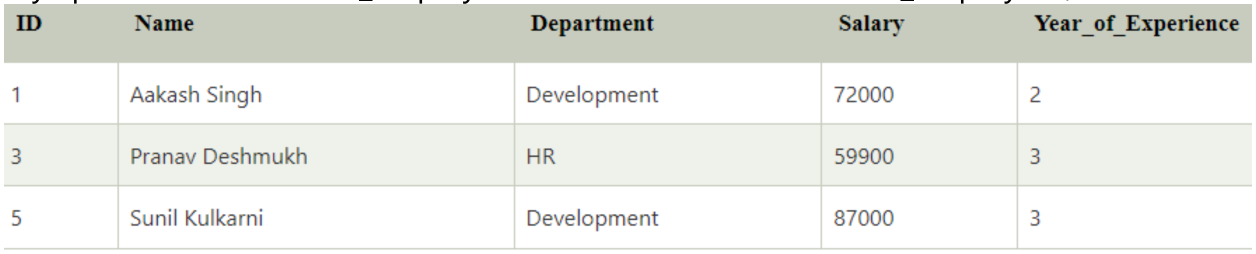
ID	Name	Department	Salary	Year_of_Experience
1	Soniya Jain	Udaipur	89	Physics
2	Harshada Sharma	Kanpur	92	Chemistry
3	Anuja Rajput	Jaipur	78	History
4	Pranali Singh	Nashik	88	Geography
5	Renuka Deshmukh	Panipat	90	Biology

```
mysql> SELECT *FROM t_employees INTERSECT SELECT *FROM t2_employees;
```



ID	Name	Hometown	Percentage	Favourite_Subject
2	Abhishek Pawar	Production	45000	1
4	Shubham Mahale	Accounts	57000	2
5	Bhushan Wagh	R&D	75000	2

```
mysql> SELECT *FROM t_employees MINUS SELECT *FROM t2_employees;
```



ID	Name	Department	Salary	Year_of_Experience
1	Aakash Singh	Development	72000	2
3	Pranav Deshmukh	HR	59900	3
5	Sunil Kulkarni	Development	87000	3

Experiment 7

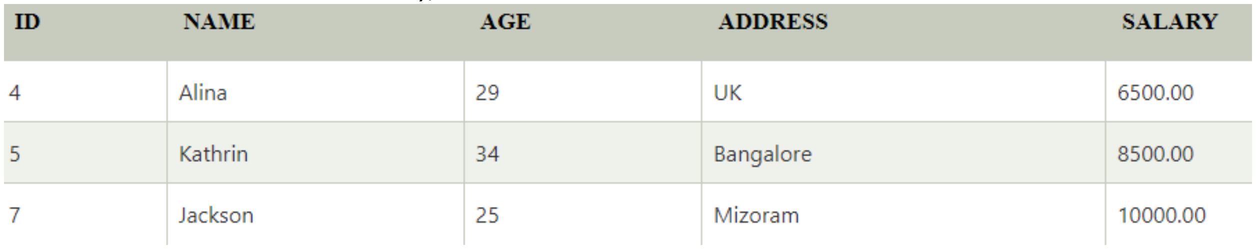
Title **Subqueries**

Objective **Study and implement the concept of sub queries.**

Syntax:

The subquery with a SELECT statement will be:

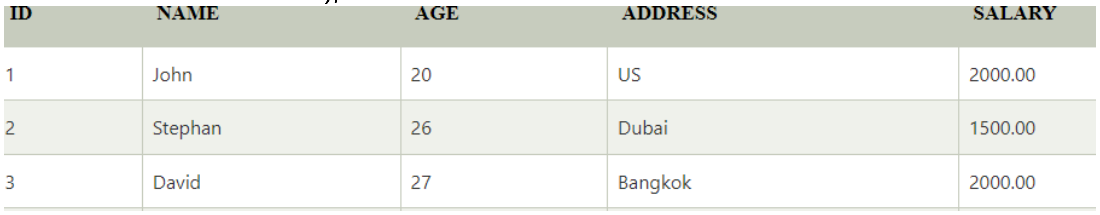
```
    SELECT *
       FROM EMPLOYEE
       WHERE ID IN (SELECT ID
       FROM EMPLOYEE
       WHERE SALARY > 4500);
```



ID	NAME	AGE	ADDRESS	SALARY
4	Alina	29	UK	6500.00
5	Kathrin	34	Bangalore	8500.00
7	Jackson	25	Mizoram	10000.00

```
INSERT INTO table_name (column1, column2, column3....)
SELECT *
FROM table_name
WHERE VALUE OPERATOR
```

```
UPDATE EMPLOYEE
  SET SALARY = SALARY * 0.25
  WHERE AGE IN (SELECT AGE FROM CUSTOMERS_BKP
    WHERE AGE >= 29);
```

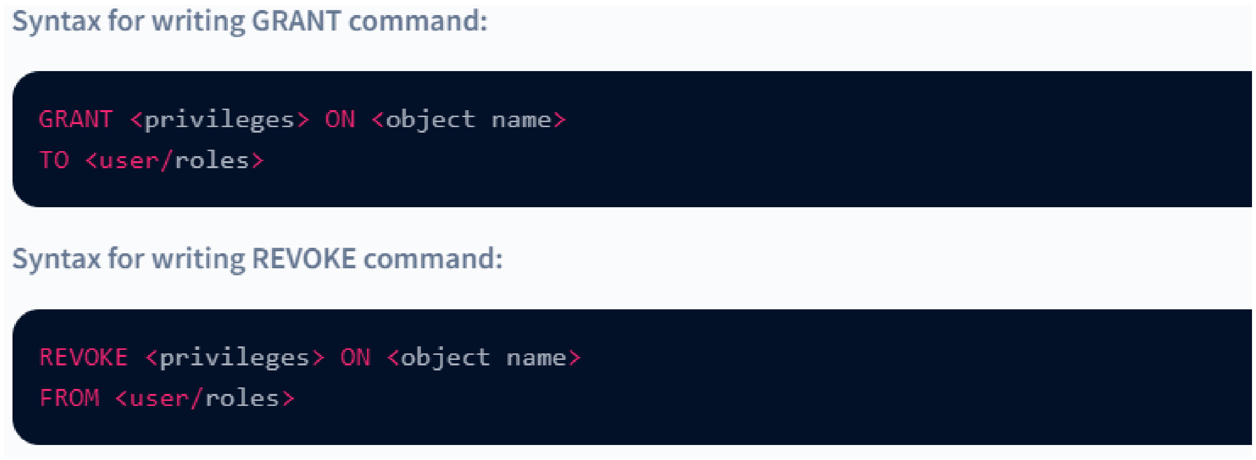


ID	NAME	AGE	ADDRESS	SALARY
1	John	20	US	2000.00
2	Stephan	26	Dubai	1500.00
3	David	27	Bangkok	2000.00

Experiment 8

Title **Control languages**

Objective **Study and implement the concept of Data Control Language (DCL), Transaction Control Language (TCL).**

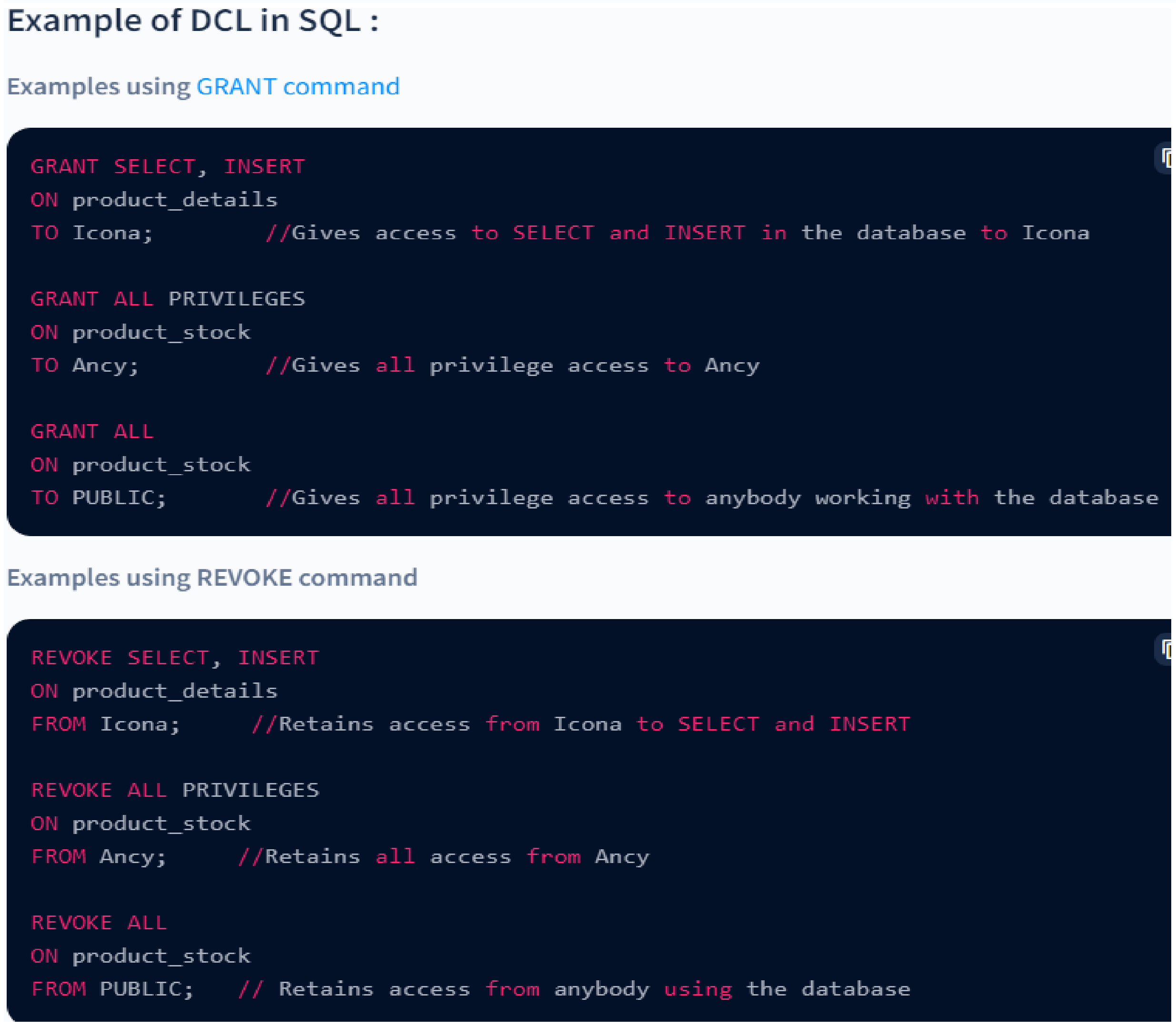


Syntax for writing GRANT command:

```
GRANT <privileges> ON <object name>
TO <user/roles>
```

Syntax for writing REVOKE command:

```
REVOKE <privileges> ON <object name>
FROM <user/roles>
```



Example of DCL in SQL :

Examples using GRANT command

```
GRANT SELECT, INSERT
ON product_details
TO Icona;         //Gives access to SELECT and INSERT in the database to Icona

GRANT ALL PRIVILEGES
ON product_stock
TO Ancy;          //Gives all privilege access to Ancy

GRANT ALL
ON product_stock
TO PUBLIC;        //Gives all privilege access to anybody working with the database
```

Examples using REVOKE command

```
REVOKE SELECT, INSERT
ON product_details
FROM Icona;     //Retains access from Icona to SELECT and INSERT

REVOKE ALL PRIVILEGES
ON product_stock
FROM Ancy;     //Retains all access from Ancy

REVOKE ALL
ON product_stock
FROM PUBLIC;   // Retains access from anybody using the database
```

Experiment 9

Title **Views**
Objective **Study of Simple and Complex View.**
Syntax

SQL CREATE VIEW Syntax

```
CREATE VIEW view_name AS
SELECT column_name(s)
FROM table_name
WHERE condition
```

Renaming the columns of a view:-

Syntax:-

```
CREATE VIEW viewname AS
SELECT newcolumnname….
FROM tablename
WHERE columnname=expression_list;
```

Selecting a data set from a view-

Syntax:-

```
SELECT columnname, columnname
FROM viewname
WHERE search condition;
```

Destroying a view-
Syntax:-

```
DROP VIEW viewname;
```

Experiment 10

Title **PL/SQL Program for Addition of Two numbers**
Objective **PL/SQL Control Structure provides conditional tests, loops, flow control and branches that let to produce well-structured programs.**
Syntax

```
SQL>set serveroutput on
SQL>declare
1 a number;
2 b number;
3 c number;
 4 begin
 5 a: =&a;
6 b: =&b;
7 c: =a+b;
 8 dbms_output.put_line ('sum of'||a||'and'||b||'is'||c);
9 end;
10 /
```

INPUT

```
Enter value for a: 23
old 6: a:=&a;
new 6: a:=23;
Enter value for b: 12
old 7: b:=&b;
new 7: b:=12;
```

OUTPUT sum of 23 and 12 is 35
PL/SQL procedure successfully completed.

Experiment 11

Title **PL/SQL block for greatest of three numbers using IF AND ELSEIF**
Objective **PL/SQL Control Structure provides conditional tests**
Syntax

```
SQL>set server output on
SQL> declare
2 a number;
3 b number;
4 c number;
5 begin
```

```
6 a:=&a;
 7 b:=&b;
8 c:=&c;
9 if(a>b)and(a>c) then
10 dbms_output.put_line('A is maximum');
11 else if(b>a)and(b>c)then
12 dbms_output.put_line('B is maximum');
13 else
14 dbms_output.put_line('C is maximum');
15 end if;
16 end;
17 /
```

INPUT

```
Enter value for a: 21
old 7: a:=&a;
 new 7: a:=21;
Enter value for b: 12
old 8: b:=&b;
new 8: b:=12;
Enter value for b: 45
old 9: c:=&b;
 new 9: c:=45;
```

OUTPUT

```
 C is maximum PL/SQL procedure successfully completed.
```

Experiment 12

Title   PL/SQL block for summation of odd numbers using for LOOP
Objective   PL/SQL Control Structure provides conditional tests, loops, flow control
and branches that let to produce well-structured programs.
Syntax

```
SQL>set server output on
SQL> declare
```

```
2 n number;
3 sum1 number default 0;
4 end value number;
5 begin
6 end value:=&end value;
 7 n:=1;
8 for n in 1..endvalue
9 loop
10 if mod(n,2)=1
11 then
12 sum1:=sum1+n;
13 end if;
14 end loop;
15 dbms_output.put_line('sum ='||sum1);
16 end;
17 /
```

INPUT

```
Enter value for end value: 4
old 6: end value:=&end value;
new 6: end value:=4;
```

OUTPUT

```
sum =4
PL/SQL procedure successfully completed.
```

Experiment 13

Title **PL/SQL Procedure for GCD Numbers**

Objective **PL/SQL Control Structure provides conditional tests.**

Syntax

create or replace procedure pro is

```
a number(3);
b number(3);
c number(3);
d number(3);
```

```
begin a:=&a;
b:=&b;
if(a>b) then c:=mod(a,b);
if(c=0) then
dbms_output.put_line('GCD is');
dbms_output.put_line(b);
 else
dbms_output.put_line('GCD is');
dbms_output.put_line(c);
 end if;
else d:=mod(b,a);
if(d=0) then
dbms_output.put_line('GCD is');
dbms_output.put_line(a);
else
dbms_output.put_line('GCD is');
dbms_output.put_line(d);
 end if;
end if;
end;
 /
```

INPUT

```
 Enter value for a: 8
old 8: a:=&a;
new 8: a:=8;
 Enter value for b: 16
old 9: b:=&b;
 new 9: b:=16;
Procedure created.
 SQL> set serveroutput on;
SQL> execute pro;
```

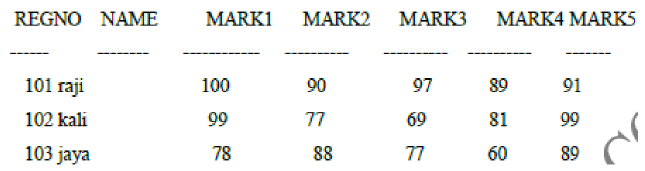
OUTPUT

```
GCD is 8
PL/SQL procedure successfully completed
```

Experiment 14

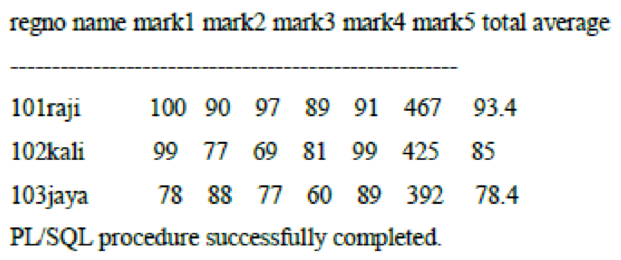
Title **PL/SQL Procedure for cursor implementation.**
Objective **To understand the concept of cursor.**
Syntax

```
insert into st13 values(101,'raji',100,90,97,89,91);
insert into a13 values(102,'kali');
 insert into a13 values(103,'jaya');
 select * from st13;
```



```
REGNO  NAME      MARK1    MARK2    MARK3    MARK4 MARK5
------ --------  -------- -------- -------- -------- -------
  101 raji        100      90       97       89      91
  102 kali         99      77       69       81      99
  103 jaya         78      88       77       60      89
```

```
SQL>set server output on
declare
ave number(5,2);
 tot number(3);
cursor c_mark is select * from st13 where mark1>=40 and mark2>=40 and mark3>=40 and
mark4>=40 and mark5>=40;
begin
dbms_output.put_line('regno name mark1 mark2 mark3 mark4 mark5 total average');
dbms_output.put_line('----------------------------------------------------');
for student in c_mark
loop
tot:=st13.mark1+st13.mark2+st13.mark3+st13.mark4+st13.mark5;
ave:=tot/5;
dbms_output.put_line(st13.regno||rpad(st13.name,15)||rpad(st13.mark1,6)||rpad(st13.mark2,6)||rp
ad(st13.mark3,6)||rpad(st13.mark4,6)||rpad(st13.mark5,6)||rpad(tot,8)||rpad(ave,5)); end loop;end;
/
```



```
regno name mark1 mark2 mark3 mark4 mark5 total average
----------------------------------------------------
101raji      100  90   97   89   91   467    93.4
102kali       99  77   69   81   99   425    85
103jaya       78  88   77   60   89   392    78.4
PL/SQL procedure successfully completed.
```

Experiment 15

Title **FUNCTION TO FIND FACTORIAL**

Objective **To find factorial using function**

Syntax

```
declare
n number;
fac number:=1;
i number;

begin
n:=&n;

for i in 1..n
loop
fac:=fac*i;
end loop;

dbms_output.put_line('factorial='||fac);
end;
/
```

Output

```
Enter value for n: 10
old 7: n:=&n;
new 7: n:=10;
factorial=3628800
```