

**PCA 2**

**MAKAUT ODD SEMESTER 2024**

|  |  |
| --- | --- |
| **NAME:** | **RUPAK SARKAR** |
| **STREAM:** | **MCA** |
| **SEMESTER:** | **1ST** |
| **SUBJECT:** | **RELATIONAL DATABASE MANAGEMENT SYSTEM** |
| **SUBJECT** | **CODE: MCAN-192** |

**Q.1. Write a PL/SQL to print Hello World.**

Ans:

**SQL:** DECLARE

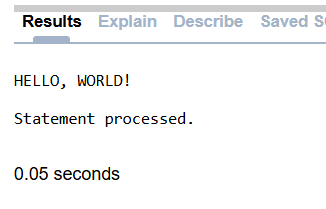
message VARCHAR2(20):= 'HELLO, WORLD!';

BEGIN

dbms\_output.put\_line(message);

END;

**Output:**



**Q.2. Write a PL/SQL program to find addition, subtraction, multiplication, division of two numbers.**

Ans:

**SQL:** DECLARE

A NUMBER(6);

B NUMBER(6);

S NUMBER(6);

SUB NUMBER(6);

M NUMBER(6);

D NUMBER(6);

BEGIN

A:=:A;

B:=:B;

S:=A+B;

SUB:=A-B;

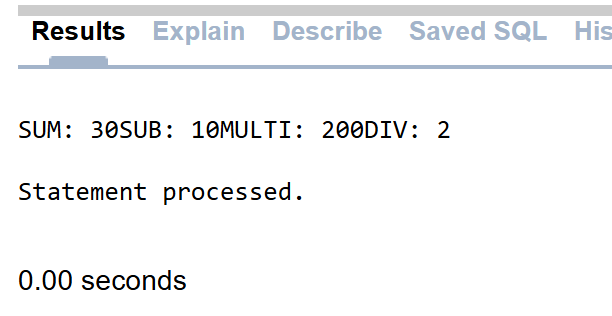
M:=A\*B;

D:=A/B;

DBMS\_OUTPUT.PUT\_LINE('SUM: '||S|| 'SUB: '||SUB||'MULTI: '||M||'DIV: '||D);

END;

**Output:**

****

**Q.3. Write a PL/SQL program to print the maximum among three numbers.**

Ans:

**SQL:** DECLARE

A NUMBER(3); B NUMBER(3); C NUMBER(3);

BEGIN

A:=:A;

B:=:B;

C:=:C;

IF A>B AND A>C THEN

DBMS\_OUTPUT.PUT\_LINE(A||' IS GREATER THAN '||B||' AND '||C);

ELSE

IF B>C AND B>C THEN

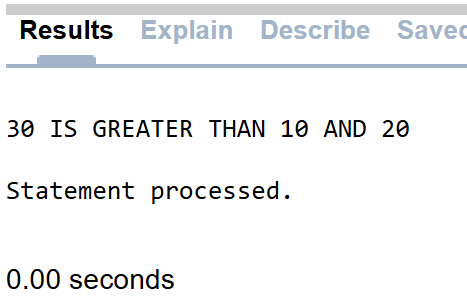
DBMS\_OUTPUT.PUT\_LINE(B||' IS GREATER THAN '||A||' AND '||C); ELSE

DBMS\_OUTPUT.PUT\_LINE(C||' IS GREATER THAN '||A||' AND '||B); END IF;

END IF;

END;

**Output:**

****

**Q.4. Write a PL/SQL program to print numbers from 1 -10 using different types of loop.**

Ans:

**SQL:** DECLARE

A NUMBER(6):=0;

BEGIN

LOOP

IF A>=10 THEN

EXIT;

END IF;

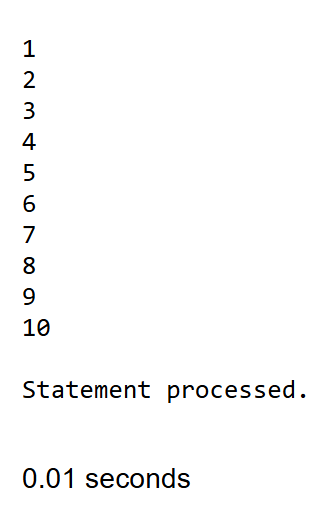
A:=A+1;

DBMS\_OUTPUT.PUT\_LINE(A);

END LOOP;

END;

**Output:**

****

**WHILE LOOP:**

**SQL:** DECLARE

A NUMBER(6):=1;

BEGIN

WHILE A<=10

LOOP

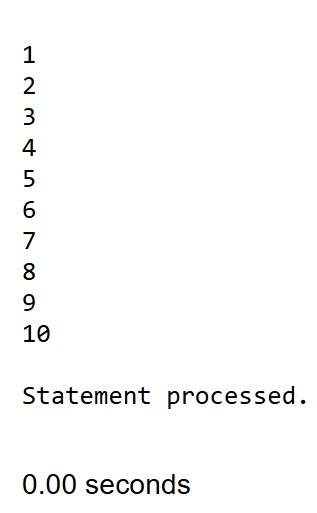
DBMS\_OUTPUT.PUT\_LINE(A);

A:=A+1;

END LOOP;

END;

**Output:**

****

**FOR LOOP:**

**SQL:** DECLARE

A NUMBER(6):=0;

BEGIN

FOR A IN 0..10

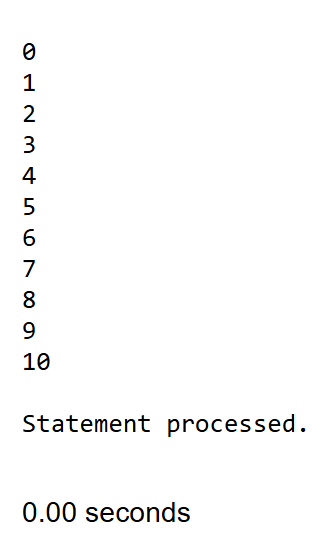
LOOP

DBMS\_OUTPUT.PUT\_LINE(A);

END LOOP;

END;

**Output:**

****

**Q.5. Write a PL/SQL program to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and the corresponding values of calculated area in an empty table named AREAS consisting of two columns RADIUS and AREA.**

Ans:

**SQL:** DECLARE

PI CONSTANT NUMBER(4,2):=3.14;

V\_RADIUS CIRCLE.RADIUS%TYPE;

V\_AREA CIRCLE.AREA%TYPE;

BEGIN

V\_RADIUS:=3;

WHILE V\_RADIUS<=7

LOOP

V\_AREA:=PI \* POWER(V\_RADIUS, 2);

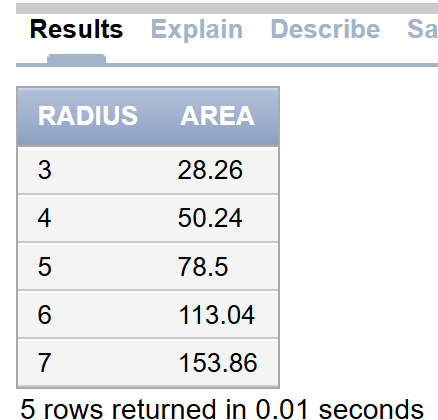
INSERT INTO CIRCLE VALUES(V\_RADIUS, V\_AREA);

V\_RADIUS:=V\_RADIUS + 1;

END LOOP;

END;

**Output:**

****

**Q.6. Write a PL/SQL program to Factorial of a number using function.**

Ans:

**SQL:** DECLARE

NUM NUMBER;

FACTORIAL NUMBER;

FUNCTION FACT(X NUMBER)

RETURN NUMBER

IS

F NUMBER;

BEGIN

IF X=0 THEN

F:=1;

ELSE

F:=X \* FACT(X-1);

END IF;

RETURN F;

DBMS\_OUTPUT.PUT\_LINE(F);

END;

BEGIN

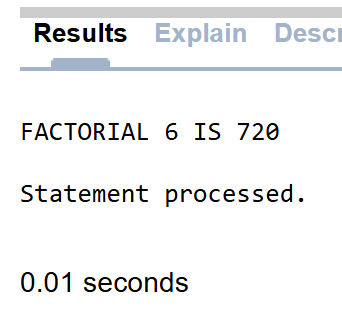
NUM:=6;

FACTORIAL:=FACT(NUM);

DBMS\_OUTPUT.PUT\_LINE('FACTORIAL '||NUM|| ' IS '||FACTORIAL);

END;

**OUTPUT:**



**Q.7. Print HELLO WORLD using Procedure.**

Ans:

**SQL:** CREATE OR REPLACE PROCEDURE GREETINGS

AS

BEGIN

DBMS\_OUTPUT.PUT\_LINE('HELLO WORLD');

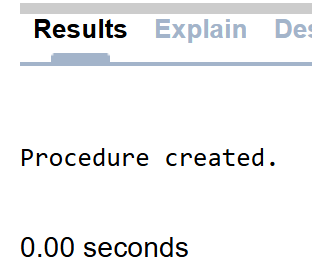
END;

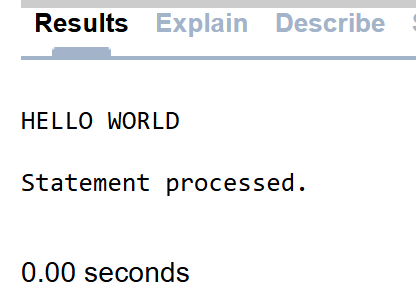
BEGIN

GREETINGS;

END;

**OUTPUT:**



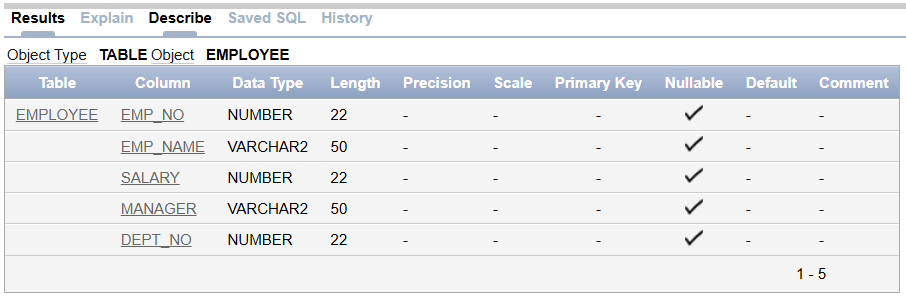


**Q.8. Write a PL/SQL script to create a table employee (emp\_no, emp\_name, salary, manager, dept\_no). Write a program to create a row-level trigger for the employee table that would fire for INSERT or UPDATE operations performed on the employee table. This trigger will display the salary difference between the old values and new values.**

Ans:

**TABLE CREATION:**

**SQL:** CREATE TABLE EMPLOYEE(EMP\_NO NUMBER, EMP\_NAME VARCHAR2(50), SALARY NUMBER, MANAGER VARCHAR2(50), DEPT\_NO NUMBER);



**VALUE INSERTION:**

**SQL:**

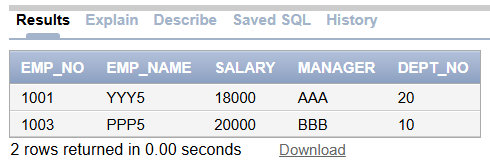
BEGIN

INSERT INTO employee VALUES(1001,'YYY5',18000,'AAA',20) ;

INSERT INTO employee VALUES(1003,'PPP5',20000,'BBB',10);

COMMIT;

END;



**TRIGGER CREATION:**

**SQL:** CREATE OR REPLACE TRIGGER salary\_difference

BEFORE INSERT OR UPDATE ON employee

FOR EACH ROW

WHEN (NEW.emp\_no > 0)

DECLARE

sal\_diff number;

BEGIN

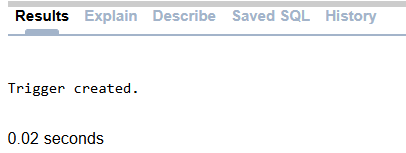
sal\_diff := :NEW.salary - :OLD.salary;

dbms\_output.put\_line('Old salary: ' || :OLD.salary);

dbms\_output.put\_line('New salary: ' || :NEW.salary);

dbms\_output.put\_line('Salary difference: ' || sal\_diff);

END;

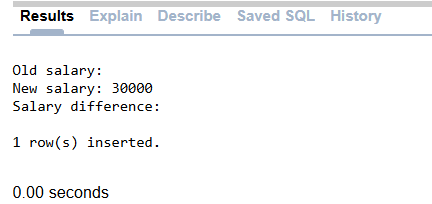


**TRIGGER CHECKING:**

**SQL:** INSERT INTO employee (emp\_no,emp\_name,salary,manager,dept\_no)

VALUES(1005,'RRR5',30000,'BBB',10);

**OUTPUT:**

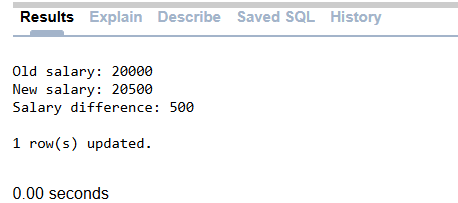


**SQL:** UPDATE employee

SET salary = salary + 500

WHERE emp\_no=1003;

**OUTPUT:**

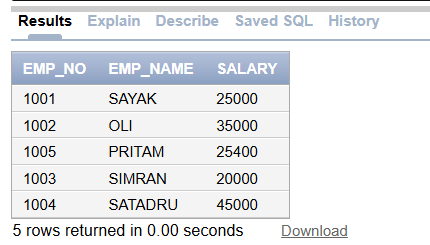


**Q.9. Write a PL/SQL script to print the details of the employee having the 2nd highest salary from employee table using explicit cursor.**

Ans:

**TABLE CREATION:**

**SQL:** CREATE TABLE EMP\_EC(emp\_no number(8),emp\_name varchar2(10),salary number(8));



**CURSOR IMPLEMENTION:**

**SQL:** DECLARE

emp\_no EMP\_EC.emp\_no%type;

emp\_name EMP\_EC.emp\_name%type;

salary EMP\_EC.salary%type;

CURSOR employee is

SELECT emp\_no,emp\_name,salary from EMP\_EC

where salary = (select max(salary) from EMP\_EC where salary <

(select max(salary) from EMP\_EC));

BEGIN

OPEN employee;

LOOP

FETCH employee INTO emp\_no,emp\_name,salary;

EXIT WHEN employee%NOTFOUND;

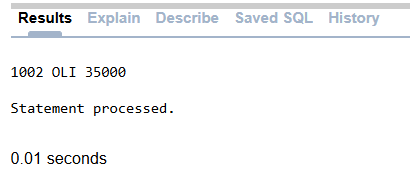
dbms\_output.put\_line(emp\_no||' '||emp\_name||' '||salary);

END LOOP;

CLOSE employee;

END;

**OUTPUT:**



**Q.10. Write a PL/SQL script to implement Package.**

Ans:

**SQL:** CREATE OR REPLACE PACKAGE my\_package AS

PROCEDURE greet(name VARCHAR2);

END my\_package;

CREATE OR REPLACE PACKAGE BODY my\_package AS

PROCEDURE greet(name VARCHAR2) IS

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Hello, ' || name);

END greet;

END my\_package;

BEGIN

my\_package.greet('HEY THERE');

END;

**OUTPUT:**

