

Question 1

Create the Following Tables with Necessary Constrains. APPLICANT (AID, ANAME, ADDR, ABIRTH_DT)

ENTRANCE TEST(ETID, ETNAME, MAX SCORE, CUT_SCORE)

ETEST CENTRE(ETCID, LOCATION, INCHARGE, CAPACITY)

ETEST DETAILS(AID, ETID, ETCID, ETEST_DT, SCORE)

This database is for a common entrance test, which is being conducted at a number of (centers and can be taken by an applicant on any day except holidays)

1. Modify the APPLICANT table so that every applicant id has an 'AP' before its value.(e.g. if value is '1123', it should become 'AP1123')
2. Display test centre details where no tests were conducted.
3. Display details about applicants who have same score as that of Ronak in 'Java Programming Language'.
4. Display details of applicants who appeared for all tests.
5. Display those tests where no applicant passed.
6. Display details of the applicants who scored more than the cut score in the tests they appeared in.
7. Display average and maximum score test wise of tests conducted at Bardoli.
8. Display the number of applicants who have appeared for each test, test centre wise.
9. Display details of applicants who have passed.

Answer

- **Create Table Queries**

```
CREATE TABLE APPLICANT (  
  AID VARCHAR(10) PRIMARY KEY,  
  ANAME VARCHAR(20),  
  ADDR VARCHAR(20),  
  ABIRTH_DT DATE
```

);
CREATE TABLE ENTRANCE_TEST (ETID INT PRIMARY KEY, ETNAME VARCHAR(30), MAX_SCORE INT, CUT_SCORE INT);
CREATE TABLE ETEST_CENTRE (ETCID INT PRIMARY KEY, LOCATION VARCHAR(20), INCHARGE VARCHAR(20), CAPACITY INT);
CREATE TABLE ETEST_DETAILS (AID VARCHAR(10), ETID INT, ETCID INT, ETEST_DT DATE, SCORE INT, FOREIGN KEY (AID) REFERENCES APPLICANT(AID), FOREIGN KEY (ETID) REFERENCES ENTRANCE_TEST(ETID), FOREIGN KEY (ETCID) REFERENCES ETEST_CENTRE(ETCID));

Insert Queries

INSERT INTO APPLICANT VALUES ('1001', 'Ronak', 'Surat', '01-Jan-2000'); INSERT INTO APPLICANT VALUES ('1002', 'Raj', 'Ahmedabad', '02-Feb-1999'); INSERT INTO APPLICANT VALUES ('1003', 'Priya', 'Vadodara', '03-Mar-2001'); INSERT INTO APPLICANT VALUES ('1004', 'Ankit', 'Rajkot', '04-Apr-2002'); INSERT INTO APPLICANT VALUES ('1005', 'Sneha', 'Bhavnagar', '05-Jan-2001');
INSERT INTO ENTRANCE_TEST VALUES (1, 'Java Programming Language', 100, 33);

```

INSERT INTO ENTRANCE_TEST VALUES (2, 'Python Programming
Language', 100, 33);
INSERT INTO ENTRANCE_TEST VALUES (3, 'Data Structures', 100, 33);
INSERT INTO ETEST_CENTRE VALUES (1, 'Bardoli', 'Miss. Riya', 100);
INSERT INTO ETEST_CENTRE VALUES (2, 'Surat', 'Mr. Patel', 100);
INSERT INTO ETEST_CENTRE VALUES (3, 'Ahmedabad', 'Mr. Mehul', 100);
INSERT INTO ETEST_DETAILS VALUES ('1001', 1, 1, '16-Mar-2024', 89);
INSERT INTO ETEST_DETAILS VALUES ('1002', 2, 3, '17-Mar-2024', 75);
INSERT INTO ETEST_DETAILS VALUES ('1003', 3, 2, '16-Mar-2024', 78);
INSERT INTO ETEST_DETAILS VALUES ('1004', 1, 2, '17-Mar-2024', 98);
INSERT INTO ETEST_DETAILS VALUES ('1005', 2, 1, '16-Mar-2024', 31);

```

```
SQL> SELECT * FROM APPLICANT;
```

AID	ANAME	ADDR	ABIRTH_DT
1001	Ronak	Surat	01-JAN-00
1002	Raj	Ahmedabad	02-FEB-99
1003	Priya	Vadodara	03-MAR-01
1004	Ankit	Rajkot	04-APR-02
1005	Sneha	Bhavnagar	05-JAN-01

```
SQL> SELECT * FROM ENTRANCE_TEST;
```

ETID	ETNAME	MAX_SCORE	CUT_SCORE
1	Java Programming Language	100	33
2	Python Programming Language	100	33
3	Data Structures	100	33

```
SQL> SELECT * FROM ETEST_CENTRE;
```

ETCID	LOCATION	INCHARGE	CAPACITY
1	Bardoli	Miss. Riya	100
2	Surat	Mr. Patel	100
3	Ahmedabad	Mr. Mehul	100

```
SQL> SELECT * FROM ETEST_DETAILS;
```

AID	ETID	ETCID	ETEST_DT	SCORE
1001	1	1	16-MAR-24	89
1002	2	3	17-MAR-24	75
1003	3	2	16-MAR-24	78
1004	1	2	17-MAR-24	98
1005	2	1	16-MAR-24	31

1. Modify the APPLICANT table so that every applicant id has an 'AP' before its value.(e.g. if value is '1123', it should become 'AP1123')

```
UPDATE APPLICANT  
SET AID = 'AP' || AID;
```

2. Display test centre details where no tests were conducted.

```
SELECT ETCID, LOCATION, INCHARGE, CAPACITY  
FROM ETEST_CENTRE  
WHERE ETCID NOT IN (SELECT DISTINCT ETCID FROM ETEST_DETAILS);
```

3. Display details about applicants who have same score as that of Ronak in 'Java Programming Language'.

```
SELECT *  
FROM ETEST_DETAILS  
WHERE ETID = 'Java Programming Language' AND SCORE = :ronak_score;
```

4. Display details of applicants who appeared for all tests.

```
SELECT AID, COUNT(ETID) AS TotalTests  
FROM ETEST_DETAILS  
GROUP BY AID  
HAVING TotalTests = (SELECT COUNT(DISTINCT ETID) FROM  
ENTRANCE_TEST);
```

5. Display those tests where no applicant passed.

```
SELECT ETID, ETNAME  
FROM ENTRANCE_TEST  
WHERE ETID NOT IN (SELECT DISTINCT ETID FROM ETEST_DETAILS WHERE  
SCORE >= CUT_SCORE);
```

6. Display details of the applicants who scored more than the cut score in the tests they appeared in.

```
SELECT * FROM ETEST_DETAILS  
WHERE SCORE > (SELECT CUT_SCORE FROM ENTRANCE_TEST WHERE  
ETID = ETEST_DETAILS.ETID);
```

7. Display average and maximum score test wise of tests conducted at Bardoli.

```
SELECT ETID, AVG(SCORE) AS AverageScore, MAX(SCORE) AS MaxScore  
FROM ETEST_DETAILS
```

```
WHERE ETCID = 'Bardoli'  
GROUP BY ETID;
```

8. Display the number of applicants who have appeared for each test, test centre wise.

```
SELECT ETID, ETCID, COUNT(DISTINCT AID) AS NumberOfApplicants  
FROM ETEST_DETAILS  
GROUP BY ETID, ETCID;
```

9. Display details of applicants who have passed.

```
SELECT * FROM ETEST_DETAILS  
WHERE SCORE >= (SELECT CUT_SCORE FROM ENTRANCE_TEST WHERE  
ETID = ETEST_DETAILS.ETID);
```

Question 2

Write a PL/SQL block to transfer all employees who are working in account department into Sales & Personal dept. according to following designation. (Take appropriate table and columns as per requirement)

IF Manager -> Sales

Otherwise -> Personal

Answer

Create Table Query :

```
SQL> CREATE TABLE EMP(  
  2  EID NUMBER PRIMARY KEY,  
  3  ENAME VARCHAR(10),  
  4  DEP  VARCHAR(10)  
  5  );  
  
Table created.
```

Insert Queries :

INSERT INTO EMP VALUES (1, 'John', 'Manager');

INSERT INTO EMP VALUES (2, 'Meet', 'Marketing');

INSERT INTO EMP VALUES (3, 'Riya', 'Production');

INSERT INTO EMP VALUES (4, 'Ronak', 'Manager');

INSERT INTO EMP VALUES (5, 'Charlie', 'IT');

INSERT INTO EMP VALUES (6, 'Mehul', 'Manager');

INSERT INTO EMP VALUES (7, 'Atul', 'HR');

INSERT INTO EMP VALUES (8, 'Sneha', 'IT');

INSERT INTO EMP VALUES (9, 'Ankit', 'Manager');

INSERT INTO EMP VALUES (10, 'Priya', 'Marketing');

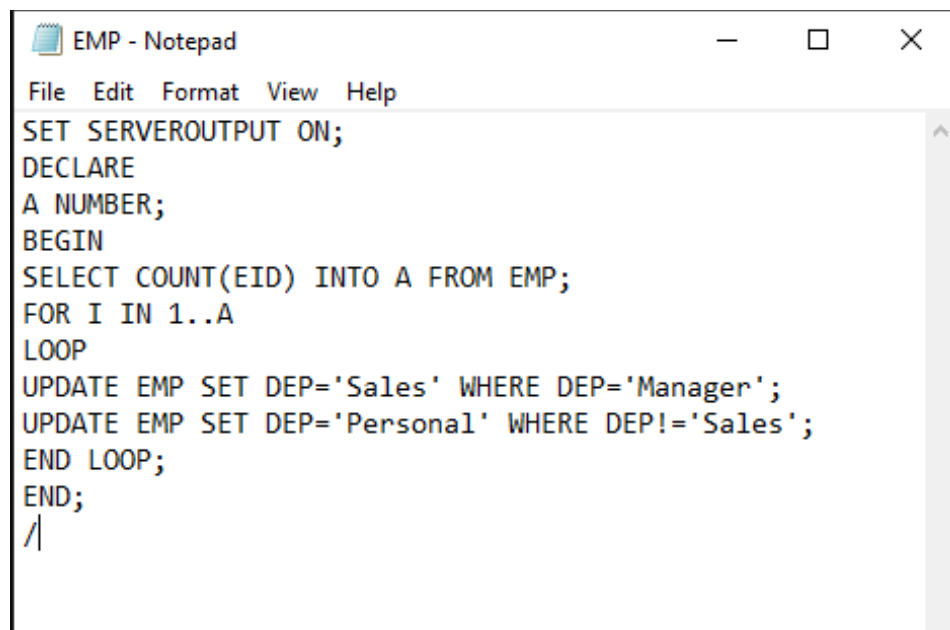
Preview :

```
SQL> SELECT * FROM EMP;
```

EID	ENAME	DEP
1	John	Manager
2	Meet	Marketing
3	Riya	Production
4	Ronak	Manager
5	Charlie	IT
6	Mehul	Manager
7	Atul	HR
8	Sneha	IT
9	Ankit	Manager
10	Priya	Marketing

```
10 rows selected.
```

PL/SQL Code :



```
EMP - Notepad
File Edit Format View Help
SET SERVEROUTPUT ON;
DECLARE
A NUMBER;
BEGIN
SELECT COUNT(EID) INTO A FROM EMP;
FOR I IN 1..A
LOOP
UPDATE EMP SET DEP='Sales' WHERE DEP='Manager';
UPDATE EMP SET DEP='Personal' WHERE DEP!='Sales';
END LOOP;
END;
/
```

Output :

```
SQL> select * from emp;
```

EID	ENAME	DEP
1	John	Sales
2	Meet	Personal
3	Riya	Personal
4	Ronak	Sales
5	Charlie	Personal
6	Mehul	Sales
7	Atul	Personal
8	Sneha	Personal
9	Ankit	Sales
10	Priya	Personal

```
10 rows selected.
```


Question 3

Write a PL/SQL code block that will accept an employee number from the user and deduct a salary by Rs.1000 from the input employee number if employee has a minimum salary of Rs.500 after salary is deducted, otherwise, display message "SALARY IS NOT ENOUGH TO DEDUCT". (Take appropriate table and columns as per requirement)

Answer

Create Table Query :

```
SQL> CREATE TABLE EMP1(
  2  EID NUMBER PRIMARY KEY,
  3  ENAME VARCHAR(15),
  4  SALARY NUMBER
  5  );

Table created.
```

Insert Queries :

```
INSERT INTO EMP1 VALUES (1, 'Harsh', 5000);
```

```
INSERT INTO EMP1 VALUES (2, 'Anurag', 6000);
```

```
INSERT INTO EMP1 VALUES (3, 'Deepak', 1000);
```

```
INSERT INTO EMP1 VALUES (4, 'Kiran', 800);
```

```
INSERT INTO EMP1 VALUES (5, 'Manish', 2000);
```

Preview :

```
SQL> SELECT * FROM EMP1;
```

EID	ENAME	SALARY
1	Harsh	5000
2	Anurag	6000
3	Deepak	1000
4	Kiran	800
5	Manish	2000

PL/SQL Code :

```

EMP1 - Notepad
File Edit Format View Help
DECLARE
    TEID EMP1.EID%TYPE := &TEID;
    tsalary EMP1.SALARY%TYPE;
BEGIN
    SELECT SALARY INTO tsalary FROM EMP1 WHERE EID = TEID ;
    IF tsalary - 1000 >= 500 THEN
        UPDATE EMP1 SET SALARY = SALARY - 1000 WHERE EID = TEID;
        DBMS_OUTPUT.PUT_LINE('Salary deducted successfully. ');
    ELSE
        DBMS_OUTPUT.PUT_LINE('SALARY IS NOT ENOUGH TO DEDUCT');
    END IF;
END;
/
    
```

Output :

If Employee Has Minimum Rs 500 After Deduct =

```

SQL> @ EMP1;
Enter value for teid: 1
old 2:      TEID EMP1.EID%TYPE := &TEID;
new 2:      TEID EMP1.EID%TYPE := 1;
Salary deducted successfully.

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMP1;

      EID ENAME      SALARY
-----
      1 Harsh      4000
      2 Anurag      6000
      3 Deepak      1000
      4 Kiran        800
      5 Manish      2000
    
```

If Employee Don't Has Minimum Rs 500 After Deduct =

```

SQL> @ EMP1;
Enter value for teid: 4
old 2:      TEID EMP1.EID%TYPE := &TEID;
new 2:      TEID EMP1.EID%TYPE := 4;
SALARY IS NOT ENOUGH TO DEDUCT

PL/SQL procedure successfully completed.
    
```

Question 4

Write a PL/SQL block to display the top five highest paid employees who are specialized in 'development'. Employee (emp_id, name, wage_per_hour, specialised_in, manager_id)(use explicit cursor)

Answer

Create Table Query :

```
SQL> CREATE TABLE Employee (
  2     emp_id NUMBER,
  3     name VARCHAR2(10),
  4     wage_per_hour NUMBER,
  5     specialised_in VARCHAR2(20),
  6     manager_id NUMBER
  7 );

Table created.
```

Insert Queries :

```
INSERT INTO Employee VALUES (1, 'John', 50, 'development', 3);
```

```
INSERT INTO Employee VALUES (2, 'Ankit', 55, 'development', 1);
```

```
INSERT INTO Employee VALUES (3, 'Bob', 60, 'design', 1);
```

```
INSERT INTO Employee VALUES (4, 'Atul', 65, 'development', 2);
```

```
INSERT INTO Employee VALUES (5, 'Priya', 70, 'development', 2);
```

```
INSERT INTO Employee VALUES (6, 'David', 75, 'design', 3);
```

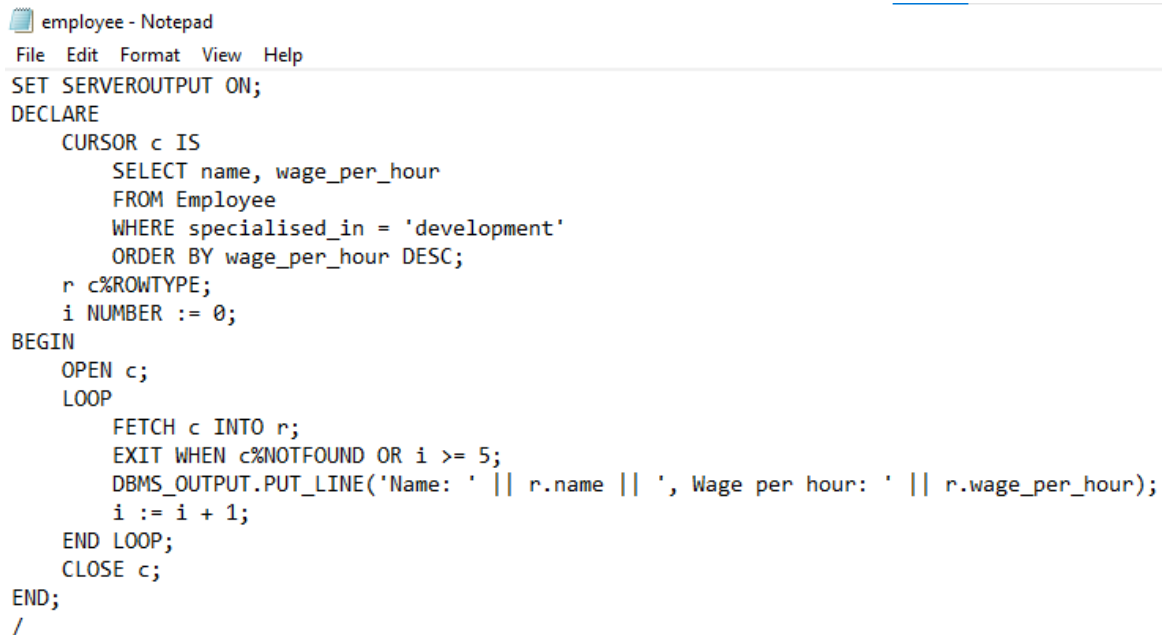
Preview :

```
SQL> SELECT * FROM EMPLOYEE;
```

EMP_ID	NAME	WAGE_PER_HOUR	SPECIALISED_IN	MANAGER_ID
1	John	50	development	3
2	Ankit	55	development	1
3	Bob	60	design	1
4	Atul	65	development	2
5	Priya	70	development	2
6	David	75	design	3

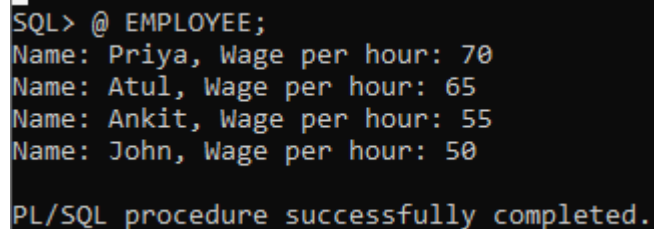
```
6 rows selected.
```

PL/SQL Code :



```
employee - Notepad
File Edit Format View Help
SET SERVEROUTPUT ON;
DECLARE
    CURSOR c IS
        SELECT name, wage_per_hour
        FROM Employee
        WHERE specialised_in = 'development'
        ORDER BY wage_per_hour DESC;
    r c%ROWTYPE;
    i NUMBER := 0;
BEGIN
    OPEN c;
    LOOP
        FETCH c INTO r;
        EXIT WHEN c%NOTFOUND OR i >= 5;
        DBMS_OUTPUT.PUT_LINE('Name: ' || r.name || ', Wage per hour: ' || r.wage_per_hour);
        i := i + 1;
    END LOOP;
    CLOSE c;
END;
/
```

Output :



```
SQL> @ EMPLOYEE;
Name: Priya, Wage per hour: 70
Name: Atul, Wage per hour: 65
Name: Ankit, Wage per hour: 55
Name: John, Wage per hour: 50

PL/SQL procedure successfully completed.
```

Question 5

Write a function, which returns the total number of incomplete jobs. (No parameters being passed) (job: jobid, type_of_job,status)

Answer

Create Table Query :

```
SQL> CREATE TABLE job (  
2     jobid NUMBER,  
3     type_of_job VARCHAR2(10),  
4     status VARCHAR2(10)  
5 );  
  
Table created.
```

Insert Queries :

```
INSERT INTO job VALUES (1, 'Desinger', 'Incomplete');
```

```
INSERT INTO job VALUES (2, 'Salesman', 'Complete');
```

```
INSERT INTO job VALUES (3, 'Manager', 'Incomplete');
```

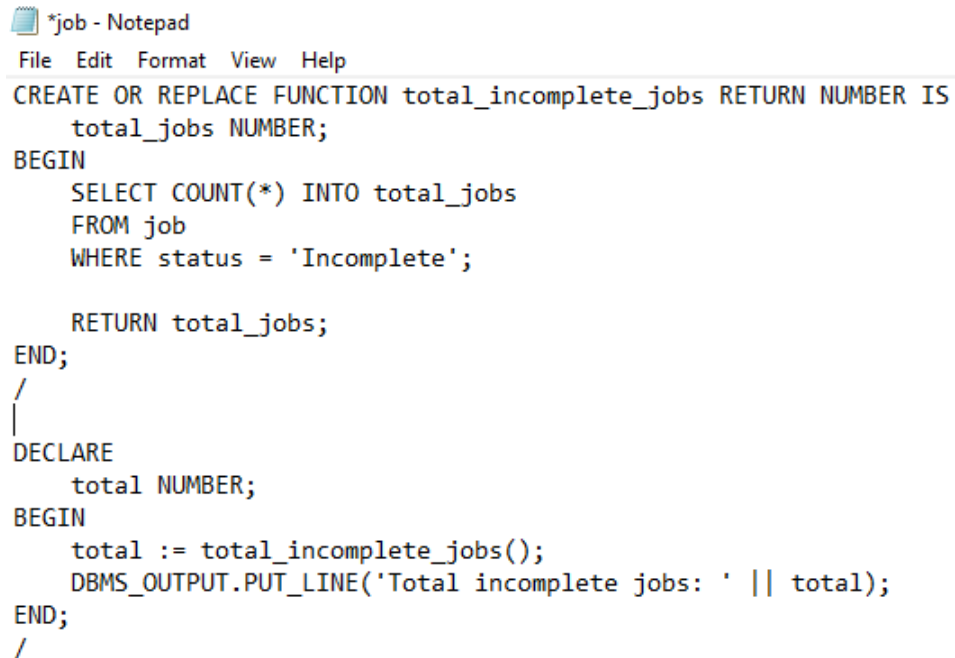
```
INSERT INTO job VALUES (4, 'Accountant', 'Complete');
```

```
INSERT INTO job VALUES (5, 'Developer', 'Incomplete');
```

Preview :

```
SQL> SELECT * FROM JOB;  
  
  JOBID TYPE_OF_JO STATUS  
-----  
    1 Desinger  Incomplete  
    2 Salesman   Complete  
    3 Manager    Incomplete  
    4 Accountant Complete  
    5 Developer  Incomplete
```

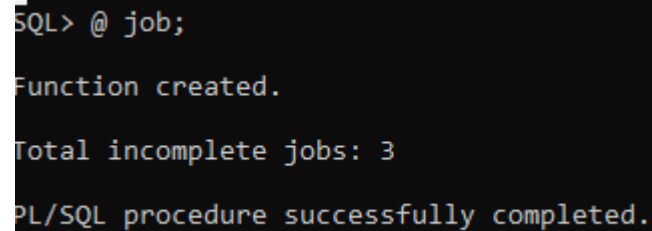
PL/SQL Code :



```
*job - Notepad
File Edit Format View Help
CREATE OR REPLACE FUNCTION total_incomplete_jobs RETURN NUMBER IS
    total_jobs NUMBER;
BEGIN
    SELECT COUNT(*) INTO total_jobs
    FROM job
    WHERE status = 'Incomplete';

    RETURN total_jobs;
END;
/
DECLARE
    total NUMBER;
BEGIN
    total := total_incomplete_jobs();
    DBMS_OUTPUT.PUT_LINE('Total incomplete jobs: ' || total);
END;
/
```

Output:



```
SQL> @ job;

Function created.

Total incomplete jobs: 3

PL/SQL procedure successfully completed.
```

Question 6

Write a PL/SQL block which accepts applicants whose age is between 18 and 45 only, if age is in range then print appropriate user defined message otherwise create user defined exception and manage it.

Answer

Create Table Query :

```
SQL> CREATE TABLE App (
  2     applicant_id NUMBER,
  3     name VARCHAR2(50),
  4     age NUMBER
  5 );
Table created.
```

Insert Queries :

```
INSERT INTO App VALUES (1, 'John', 20);
```

```
INSERT INTO App VALUES (2, 'Ankit', 50);
```

```
INSERT INTO App VALUES (3, 'Bob', 59);
```

```
INSERT INTO App VALUES (4, 'Atul', 45);
```

```
INSERT INTO App VALUES (5, 'Priya', 18);
```

- **Preview :**

```
SQL> SELECT * FROM APP;

APPLICANT_ID NAME          AGE
-----
1 John          20
2 Ankit         50
3 Bob           59
4 Atul          45
5 Priya         18
```

PL/SQL Code :

```
app - Notepad
File Edit Format View Help
DECLARE
    age_out_of_range EXCEPTION;
    v_applicant App%ROWTYPE;
    v_applicant_id App.applicant_id%TYPE;
BEGIN
    v_applicant_id := &applicant_id;
    SELECT *
    INTO v_applicant
    FROM App
    WHERE applicant_id = v_applicant_id;

    IF v_applicant.age < 18 OR v_applicant.age > 45 THEN
        RAISE age_out_of_range;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Applicant is within the age range.');
```

Output :

If Applicant's Age Is Between 18 And 45

```
SQL> @ app;
Enter value for applicant_id: 4
old 6:      v_applicant_id := &applicant_id;
new 6:      v_applicant_id := 4;
Applicant is within the age range.

PL/SQL procedure successfully completed.
```

If Applicant's Age Is Not Between 18 And 45

```
SQL> @ app;
Enter value for applicant_id: 3
old 6:      v_applicant_id := &applicant_id;
new 6:      v_applicant_id := 3;
Applicant is not within the age range.

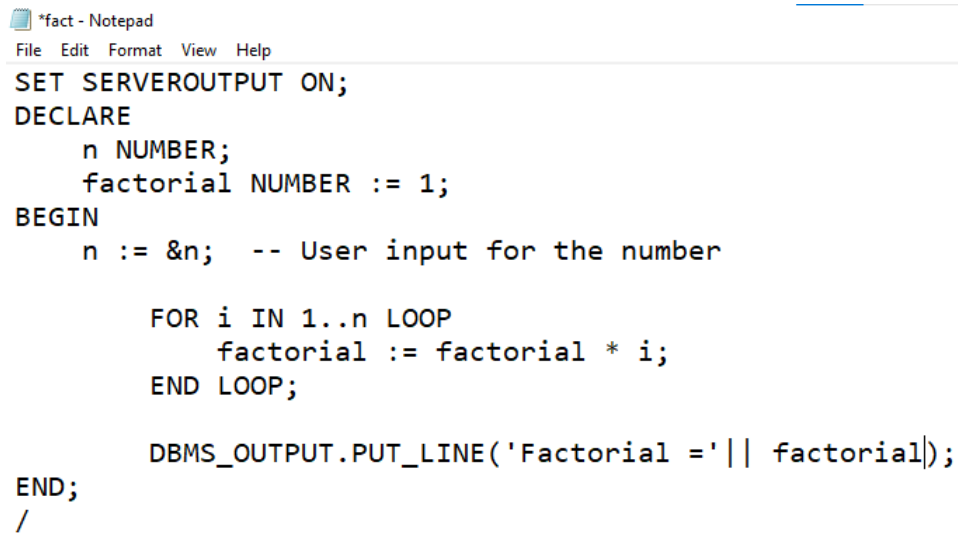
PL/SQL procedure successfully completed.
```


Question 7

PL/SQL Program to Find Factorial of a Number.

Answer

PL/SQL Code :

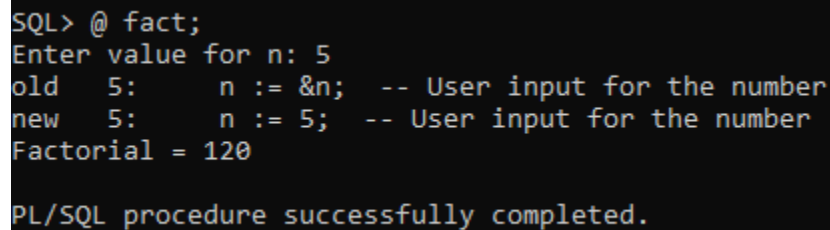
A screenshot of a Notepad window titled '*fact - Notepad'. The window contains the following PL/SQL code:

```
File Edit Format View Help
SET SERVEROUTPUT ON;
DECLARE
    n NUMBER;
    factorial NUMBER := 1;
BEGIN
    n := &n;  -- User input for the number

    FOR i IN 1..n LOOP
        factorial := factorial * i;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Factorial = ' || factorial);
END;
/
```

Output :

A screenshot of a SQL*Plus session showing the execution of the PL/SQL program. The output is as follows:

```
SQL> @ fact;
Enter value for n: 5
old   5:      n := &n;  -- User input for the number
new   5:      n := 5;   -- User input for the number
Factorial = 120

PL/SQL procedure successfully completed.
```