Assignment 6

Q1. 1. The bank manager has decided to activate all those accounts which were previously marked as inactive for performing no transaction in the last 365 days. Write a PL/SQL block (using implicit cursor) to update the status of the account, display an approximate message based on the no. of rows affected by the update.

Ans:-

```
CREATE TABLE acc_details(Acc_no NUMBER, Name VARCHAR(9), status VARCHAR(8));
  INSERT INTO acc_details VALUES(1520, 'XYZ', 'Inactive');
  INSERT INTO acc_details VALUES(1521, 'abc', 'Active');
  INSERT INTO acc_details VALUES(1522, 'pqr', 'Inactive');
  INSERT INTO acc_details VALUES(1523, 'lmn', 'Inactive');
  INSERT INTO acc_details VALUES(1524, 'Xab', 'Active');
```

ACC_NO	NAME	STATUS
1520	XYZ	Inactive
1521	abc	Active
1522	pqr	Inactive
1523	lmn	Inactive
1524	Xab	Active

```
DECLARE
 Row_count NUMBER;
BEGIN
  UPDATE acc_details SET status='Active' WHERE status='Inactive';
  Row_count := SQL%ROWCOUNT;
  DBMS_OUTPUT.PUT_LINE('Number of updated rows are ' || Row_count);
EXCEPTION
  WHEN Others THEN
```

```
    DBMS_OUTPUT.PUT_LINE('ERROR!');
END;
```

OUTPUT:-

Number of updated rows are 3

PL/SQL procedure successfully completed.

After execution of PL/SQL Block:-

ACC_NO	NAME	STATUS
1520	XYZ	Active
1521	abc	Active
1522	pqr	Active
1523	lmn	Active
1524	Xab	Active

Q2. Organization has decided to increase the salary of employees by 10% of existing salary, who are having salaries less than the average salary of the organization. Whenever such salary updates take place, a record for the same is maintained in the increment_salary table.

EMP (E_no , Salary)

Increment_salary(E_no , Salary)

Ans:- (NOT COMPLETED)

```
CREATE TABLE EMP(E_no NUMBER, Salary NUMBER);
  INSERT INTO EMP VALUES(1361,25000);
  INSERT INTO EMP VALUES(1362,21000);
  INSERT INTO EMP VALUES(1363,35000);
  INSERT INTO EMP VALUES(1364,45000);
  INSERT INTO EMP VALUES(1365,19000);
CREATE TABLE Increment_salary(E_no NUMBER, Salary NUMBER);
```

```
DECLARE
  Avg_sal NUMBER;
  Updated_sal NUMBER;
  Sal NUMBER;
  emp_no NUMBER;
  CURSOR C1 IS SELECT E_no,Salary FROM EMP;
BEGIN
  SELECT AVG(Salary) INTO Avg_sal FROM EMP;
  OPEN C1;
   LOOP
    FETCH C1 INTO emp_no, sal;
    EXIT WHEN C1%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('E_no:  ' || emp_no || '  Salary:  ' || sal);
    IF sal < Avg_sal THEN
      Updated_sal := sal * 1.1;
      UPDATE EMP SET Salary = Updated_sal WHERE E_no = emp_no;
      INSERT INTO Increment_salary VALUES (emp_no, Updated_sal);
    END IF;
  END LOOP;
  CLOSE C1;
END;
```

OUTPUT:-

```
E_no:  1361  Salary:  25000
E_no:  1362  Salary:  21000
E_no:  1363  Salary:  35000
E_no:  1364  Salary:  45000
E_no:  1365  Salary:  19000
E_no:  1365  Salary:  19000
PL/SQL procedure successfully completed.
```

Q3. Write PL/SQL block using explicit cursor for following requirements: College has decided to mark all those students detained (D) who are having attendance less than 75%. Whenever such an update takes place, a record for the same is maintained in the D_Stud table. create table

stud19(roll number(4), att number(4), status varchar(1)); create table d_stud(roll number(4), att number(4));

Ans:-

```
CREATE TABLE stud19(roll number(4), att number(4), status varchar(1));
   INSERT INTO stud19 VALUES (1,89,'-' );
   INSERT INTO stud19 VALUES (2,69,'-' );
   INSERT INTO stud19 VALUES (3,76,'-' );
   INSERT INTO stud19 VALUES (4,55,'-' );
   INSERT INTO stud19 VALUES (5,99,'-' );

CREATE TABLE d_stud(roll number(4), att number(4));


DECLARE
  Stud_att stud19.att%type;
  Stud_roll stud19.roll%type;
  CURSOR c IS SELECT roll,att FROM stud19;
BEGIN
  OPEN c;
  LOOP
    FETCH c INTO Stud_roll,Stud_att;
    EXIT WHEN c%NOTFOUND;
    IF (Stud_att < 75) THEN
      UPDATE stud19 SET status = 'D' WHERE roll = Stud_roll;
      INSERT INTO d_stud VALUES (Stud_roll,Stud_att);
    END IF;
  END LOOP;
  CLOSE c;
END;
```

OUTPUT:-
PL/SQL procedure successfully completed.
—----------

Q4. Write a PL/SQL block of code using parameterized Cursor, that will merge the data available in the newly created table N_RollCall with the data available in the table O_RollCall. If the data in the first table already exists in the second table then that data should be skipped.

Ans:-

```
CREATE TABLE N_RollCall (id NUMBER,name VARCHAR2(10));
  INSERT INTO N_RollCall  VALUES (2, 'XYZ');
  INSERT INTO N_RollCall  VALUES (4, 'ABC');

CREATE TABLE O_RollCall (id NUMBER,name VARCHAR2(10));
  INSERT INTO O_RollCall  VALUES (1, 'lmn');
  INSERT INTO O_RollCall  VALUES (2, 'XYZ');
  INSERT INTO O_RollCall  VALUES (3, 'pqr');
  INSERT INTO O_RollCall  VALUES (4, 'ABC');
```

```
DECLARE
        CURSOR old_csr is select id,name from O_RollCall;
        CURSOR new_csr(id1 number) is select * from N_RollCall where id=id1;
        s_id O_RollCall.id%type;
        s_name O_RollCall.name%type;
        v  N_RollCall%rowtype;
BEGIN
        open old_csr;
        if old_csr%isopen then
        loop
                fetch old_csr into s_id,s_name;
                exit when old_csr%notfound;
                open new_csr(s_id);
                if new_csr%isopen then
                                fetch new_csr into v;
                                if new_csr%notfound then
                                INSERT INTO N_RollCall  VALUES (s_id, s_name);
                                end if;
                end if;
                close new_csr;
        end loop;
        end if;
        close old_csr;
END;
```

OUTPUT:-

```
PL/SQL procedure successfully completed.

SQL> select * from N_RollCall;

        ID NAME
---------- ----------
         2 XYZ
         4 ABC
         1 lmn
         3 pqr
```

Q5. Write the PL/SQL block for following requirements using parameterized Cursor:
Consider table EMP(e_no, d_no, Salary), department wise average salary should be inserted into the new table dept_salary(d_no, Avg_salary).

Ans:-

```
create table emp (emp number,d_no number,salary number);
   insert into emp values(1,101,4000);
   insert into emp values(2,101,5000);
   insert into emp values(3,103,5000);
   insert into emp values(4,104,10000);
   insert into emp values(5,105,8000);
```

```
CREATE TABLE dept_salary(d_no NUMBER, Avg_salary NUMBER);

Declare
  Cursor crsr1 is select d_no from emp;
  Cursor crsr2(dept_no number) is select avg(salary) from emp where d_no=dept_no;
  Cursor crsr3(dept_no number) is select avg_salary from dept_salary where d_no=dept_no;
  dept_id emp.d_no%type;
  avg_sal number;
Begin
  open crsr1;
  loop
    fetch crsr1 into dept_id;
    exit when crsr1%notfound;
    open crsr2(dept_id);
    fetch crsr2 into avg_sal;
    open crsr3(dept_id);
    fetch crsr3 into avg_sal;
    if crsr3%notfound then
      insert into dept_salary values(dept_id,avg_sal);
    end if;
  close crsr3;
  close crsr2;
  end loop;
  close crsr1;
End;

PL/SQL procedure successfully completed.

SQL> SELECT * FROM dept_salary;

D_NO        AVG_SALARY
----------  ----------
       101       4500
       103       5000
       104      10000
       105       8000
```

Q6. Write PL/SQL block using explicit cursor: Cursor FOR Loop for following requirements:

College has decided to mark all those students detained (D) who are having attendance less than 75%.
Whenever such update takes place, a record for the same is maintained in the D_Stud table.
create table stud21(roll number(4), att number(4), status varchar(1));
create table d_stud(roll number(4), att number(4));

Ans:-

```
CREATE TABLE stud19(roll number(4), att number(4), status varchar(1));
   INSERT INTO stud19 VALUES (1,89,'-' );
```

```
   INSERT INTO stud19 VALUES (2,69,'-' );
   INSERT INTO stud19 VALUES (3,76,'-' );
   INSERT INTO stud19 VALUES (4,55,'-' );
   INSERT INTO stud19 VALUES (5,99,'-' );

CREATE TABLE d_stud(roll number(4), att number(4));

Declare
  cursor update_status is select roll,att from stud19;
Begin
  for demo in update_status
  loop
    if demo.att< 75 then
      update stud19 set status='D' where roll=demo.roll;
      insert into d_stud values(demo.roll,demo.att);
    else
      update stud19 set status='-' where roll=demo.roll;
    end if;
  end loop;
End;

PL/SQL procedure successfully completed.

SQL> SELECT * FROM stud19;

      ROLL       ATT     S
----------      -------     ——-
         1         89       -
         2         69       D
         3         76       -
         4         55       D
         5         99       -
```