ASSIGNMENT NO.6

1. Write a PL/SQL stored Procedure for following requirements and call the procedure in appropriate PL/SQL block.

1. Borrower(Rollin, Name, DateofIssue, NameofBook, Status)

2. Fine(Roll_no,Date,Amt)

- Accept roll_no &name of book from user.
- Check the number of days (from date of issue), if days are between 15 to 30 then fine amount will be Rs 5per day.
- If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day.
- After submiting the book, status will change from I to R.
- If condition of fine is true, then details will be stored into fine table.

```
SQL> CREATE TABLE Borrower (
  2     RollIn NUMBER PRIMARY KEY,
  3     Name VARCHAR2(20),
  4     DateOfIssue DATE,
  5     NameofBook VARCHAR2(20),
  6     Status CHAR(1)
  7  );
Table created.
SQL> CREATE TABLE FINE (
  2     RollNo NUMBER,
  3     DateOfFine DATE,
  4     Amt NUMBER(10,2),
  5     FOREIGN KEY (RollNo) REFERENCES Borrower(RollIn)
  6  );
Table created.
SQL> INSERT INTO Borrower  VALUES (1, 'Soham', TO_DATE('2024-03-9', 'YYYY-MM-DD'), 'Book1', 'I');
1 row created.
SQL> INSERT INTO Borrower  VALUES (2, 'Mrunalini', TO_DATE('2024-02-25', 'YYYY-MM-DD'), 'Book2', 'I');
1 row created.
SQL> INSERT INTO Borrower  VALUES (3, 'Hari', TO_DATE('2024-03-5', 'YYYY-MM-DD'), 'Book3', 'I');
1 row created.
SQL> INSERT INTO Borrower  VALUES (4, 'Atahrva', TO_DATE('2024-02-24', 'YYYY-MM-DD'), 'Book4', 'I');
1 row created.
SQL> INSERT INTO Borrower  VALUES (5, 'Khelesh', TO_DATE('2024-03-23', 'YYYY-MM-DD'), 'Book5', 'R');
1 row created.
SQL> CREATE OR REPLACE PROCEDURE CheckFine(
  2    Roll_no IN NUMBER
  3  )
  4  IS
  5    IssuedDays NUMBER;
  6    FineAmt DECIMAL(10,2);
  7    IssuedDate DATE;
  8   BookStatus varchar(1);
  9
 10  BEGIN
 11    SELECT DateOfIssue, Status
 12    INTO IssuedDate, BookStatus
```

```
 13   FROM Borrower
 14   WHERE Borrower.RollIn = Roll_no;
 15   IF BookStatus = 'I' THEN
 16     IssuedDays := SYSDATE - IssuedDate;
 17     IF IssuedDays > 15 AND IssuedDays < 30 THEN
 18       FineAmt := IssuedDays * 5;
 19     ELSIF IssuedDays >= 30 THEN
 20       FineAmt := (IssuedDays - 30) * 5 + 1500;
 21     ELSE
 22       FineAmt := 0; -- No fine within first 15 days
 23     END IF;
 24     IF FineAmt > 0 THEN
 25       INSERT INTO FINE (RollNo, Amt, DateOfFine) VALUES (Roll_no, FineAmt, SYSDATE);
 26     END IF;
 27   ELSE
 28     DBMS_OUTPUT.PUT_LINE('Book is not issued for the Roll Number ' || Roll_no);
 29   END IF;
 30  EXCEPTION
 31   WHEN NO_DATA_FOUND THEN
 32     DBMS_OUTPUT.PUT_LINE('No record found for Roll Number ' || Roll_no);
 33  END;
 34  /
Procedure created.
SQL>
SQL> DECLARE
  2   CURSOR BorrowerCursor IS
  3     SELECT RollIn FROM Borrower;
  4  BEGIN
  5   FOR BorrowerRec IN BorrowerCursor LOOP
  6     CheckFine(BorrowerRec.RollIn);
  7   END LOOP;
  8  END;
  9  /
PL/SQL procedure successfully completed.
SQL> SELECT * FROM FINE;

    ROLLNO DATEOFFIN        AMT
---------- --------- ----------
         1 27-MAR-24      94.57
         2 27-MAR-24    1509.57
         3 27-MAR-24     114.57
         4 27-MAR-24    1514.57
```

2. Write a stored function in PL/SQL for given requirement and use the same in PL/SQL block.

Account no. and branch name will be accepted from user. The same will be searched in table acct_details. If status of account is active then display appropriate message and also store the account details in active_acc_details table, otherwise display message on screen “account is inactive”.

```
SQL> CREATE TABLE Acc_details(
  2  AccountNo INT PRIMARY KEY,
  3  Branch varchar(10),
  4  AccountStatus varchar(1)
  5  );
Table created.
```

```
SQL> CREATE TABLE active_acc_details(
  2  AccountNo INT PRIMARY KEY,
  3  Branch varchar(10)
  4  );
Table created.
SQL> INSERT INTO Acc_details VALUES (1, 'Branch1', 'A');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (2, 'Branch2', 'A');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (3, 'Branch1', 'I');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (4, 'Branch3', 'A');
1row created.
SQL> INSERT INTO Acc_details  VALUES (5, 'Branch2', 'I');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (6, 'Branch1', 'A');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (7, 'Branch3', 'I');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (8, 'Branch2', 'A');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (9, 'Branch1', 'I');
1 row created.
SQL> INSERT INTO Acc_details  VALUES (10, 'Branch3', 'A');
1 row created.

SQL>
SQL> CREATE OR REPLACE PROCEDURE CheckStatus(
  2    Acc_number IN NUMBER,
  3    Acc_branch IN varchar2 -- Use varchar2 instead of varchar
  4  )
  5  IS
  6   Acc_status varchar2(1); -- Define size for varchar2
  7  BEGIN
  8   SELECT AccountStatus INTO Acc_status FROM Acc_details WHERE AccountNo = Acc_number AND Branch =
Acc_branch;
  9
 10   IF Acc_status = 'A' THEN
 11     INSERT INTO active_acc_details VALUES (Acc_number, Acc_branch);
 12     DBMS_OUTPUT.PUT_LINE('Account with Account No ' || Acc_number || ' is active.');
 13   ELSIF Acc_status = 'I' THEN
 14     DBMS_OUTPUT.PUT_LINE('Account with Account No ' || Acc_number || ' is inactive.');
 15   ELSE
 16     DBMS_OUTPUT.PUT_LINE('Account with Account No ' || Acc_number || ' does not exist.');
 17   END IF;
 18  EXCEPTION
 19   WHEN NO_DATA_FOUND THEN
 20     DBMS_OUTPUT.PUT_LINE('Account with Account No ' || Acc_number || ' does not exist.');
 21  END;
 22  /
Procedure created.
SQL> DECLARE
  2   v_account_number Acc_details.AccountNo%TYPE;
  3   v_branch Acc_details.Branch%TYPE;
  4  BEGIN
  5   FOR account_row IN (SELECT AccountNo, Branch FROM Acc_details) LOOP
```

```
  6     v_account_number := account_row.AccountNo;
  7     v_branch := account_row.Branch;
  8     CheckStatus(v_account_number, v_branch);
  9    END LOOP;
 10  END;
 11  /
```
Account with Account No 1 is active.
Account with Account No 2 is active.
Account with Account No 3 is inactive.
Account with Account No 4 is active.
Account with Account No 5 is inactive.
Account with Account No 6 is active.
Account with Account No 7 is inactive.
Account with Account No 8 is active.
Account with Account No 9 is inactive.
Account with Account No 10 is active.
PL/SQL procedure successfully completed.

```
SQL> SELECT * FROM active_acc_details;

 ACCOUNTNO BRANCH
---------- ----------
         1 Branch1
         2 Branch2
         4 Branch3
         6 Branch1
         8 Branch2
        10 Branch3
```

6 rows selected.

```
SQL> CREATE TABLE Stud_Marks (
  2     name VARCHAR2(100) PRIMARY KEY,
  3     total_marks NUMBER
  4  );
```

Table created.

```
SQL>
SQL> CREATE TABLE Result (
  2     Name VARCHAR2(100),
  3     Class VARCHAR2(100),
  4     FOREIGN KEY (Name) REFERENCES Stud_Marks(name)
  5  );
```

Table created.

```
SQL>
SQL> CREATE OR REPLACE PROCEDURE proc_Grade (
  2     marks IN NUMBER,
  3     grade OUT VARCHAR2
  4  )
  5  AS
  6  BEGIN
  7     IF marks <= 1500 AND marks >= 990 THEN
  8        grade := 'Distinction';
  9     ELSIF marks <=989 AND marks >= 900 THEN
 10         grade := 'First Class';
 11      ELSIF marks <= 899 AND marks >= 825 THEN
```

```
 12        grade := 'Higher Second Class';
 13     ELSE
 14        grade := 'No Grade';
 15     END IF;
 16  END;
 17  /
```

Procedure created.

3. Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and900 category is first class, if marks 899 and 825 category is Higher Second Class

Write a PL/SQL block for using procedure created with above requirement.

Stud_Marks(name, total_marks)

Result(Roll,Name, Class)

```
SQL> DECLARE
  2     v_name VARCHAR2(100);
  3     v_total_marks NUMBER;
  4     v_class VARCHAR2(100);
  5  BEGIN
  6     -- Inserting provided student data
  7     INSERT INTO Stud_Marks (name, total_marks) VALUES ('Soham', 1200);
  8     INSERT INTO Stud_Marks (name, total_marks) VALUES ('Hari', 1000);
  9     INSERT INTO Stud_Marks (name, total_marks) VALUES ('Mrunalini', 950);
 10      INSERT INTO Stud_Marks (name, total_marks) VALUES ('Atharva', 850);
 11      INSERT INTO Stud_Marks (name, total_marks) VALUES ('Vedant', 1100);
 12      INSERT INTO Stud_Marks (name, total_marks) VALUES ('Sanju', 1400);
 13      INSERT INTO Stud_Marks (name, total_marks) VALUES ('Ragini', 800);
 14      FOR rec IN (SELECT name, total_marks FROM Stud_Marks) LOOP
 15         v_name := rec.name;
 16         v_total_marks := rec.total_marks;
 17         proc_Grade(v_total_marks, v_class);
 18         INSERT INTO Result (Name, Class) VALUES (v_name, v_class);
 19         DBMS_OUTPUT.PUT_LINE('Student: ' || v_name || ', Class: ' || v_class);
 20     END LOOP;
 21  END;
 22  /
Student: Soham, Class: Distinction
Student: Hari, Class: Distinction
Student: Mrunalini, Class: First Class
Student: Atharva, Class: Higher Second Class
Student: Vedant, Class: Distinction
Student: Sanju, Class: Distinction
Student: Ragini, Class: No Grade

PL/SQL procedure successfully completed.

SQL> COLUMN Name FORMAT A15
SQL> COLUMN Class FORMAT A20
```

```
SQL> SELECT * FROM RESULT;

NAME            CLASS
--------------- --------------------
Soham           Distinction
Hari            Distinction
Mrunalini       First Class
Atharva         Higher Second Class
Vedant          Distinction
Sanju           Distinction
Ragini          No Grade

7 rows selected.
```