**4.Unnamed PL/SQL code block: Use of Control structure and Exception handling is**

**mandatory.**

**Suggested Problem statement:**

**Consider Tables:**

**1. Borrower(Roll\_no, Name, DateofIssue, NameofBook, Status)**

**2. Fine(Roll\_no,Date,Amt)**

** Accept Roll\_no and NameofBook 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 and for days less than 30, Rs. 5 per**

**day.**

** After submitting the book, status will change from I to R.**

** If condition of fine is true, then details will be stored into fine table.**

** Also handles the exception by named exception handler or user define exception handler.**

CREATE TABLE Borrower(Roll INT PRIMARY KEY NOT NULL AUTO\_INCREMENT,Name VARCHAR(20),DateofIssue DATE,NameofBook VARCHAR(20),Status VARCHAR(2));

CREATE TABLE Fine(Roll INT,Date DATE,Amt INT,Foreign Key(Rol

l) REFERENCES Borrower(Roll));

INSERT INTO Borrower(Name,DateofIssue,NameofBook,Status) VALUES ('Sarthik','2023-10-15','CNS','I');

Query OK, 1 row affected (0.007 sec)

MariaDB [dbms]> INSERT INTO Borrower(Name,DateofIssue,NameofBook,Status) VALUES ('Vijay','2023-10-20','DBMS','I');

Query OK, 1 row affected (0.026 sec)

MariaDB [dbms]> INSERT INTO Borrower(Name,DateofIssue,NameofBook,Status) VALUES ('Ritika','2023-10-01','TOC','I');

Query OK, 1 row affected (0.006 sec)

MariaDB [dbms]> INSERT INTO Borrower(Name,DateofIssue,NameofBook,Status) VALUES ('Kannika','2023-11-01','SPOS','I');

Query OK, 1 row affected (0.007 sec)

MariaDB [dbms]> INSERT INTO Borrower(Name,DateofIssue,NameofBook,Status) VALUES ('Avish','2023-10-10','DS','I');

Query OK, 1 row affected (0.022 sec)

DELIMITER $$

CREATE PROCEDURE calc\_fine(IN rno INT,bname VARCHAR(20))

BEGIN

DECLARE i\_date DATE;

DECLARE fineAmt INT;

DECLARE diff INT;

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT='Error Occured';

END;

SELECT DateofIssue INTO i\_date FROM Borrower WHERE Roll=rno AND NameofBook=bname AND Status!='R';

SET fineAmt=0;

SET diff=DATEDIFF(CURDATE(),i\_date);

IF (diff<0) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT='Invalid Date Calculation';

ELSEIF (diff>15 AND diff<30) THEN

SET fineAmt=diff\*5;

ELSEIF (diff>30) THEN

SET fineAmt= (30-diff)\*50+15\*5;

ELSE

SET fineAmt=0;

END IF;

INSERT INTO Fine VALUES(rno,i\_date,fineAmt);

UPDATE Borrower SET Status='R' WHERE Roll=rno AND NameofBook=bname;

END;

$$

**( perform these quries to get output)**

call calc\_fine(1,'CNS');

select \* from fine;

select \* from Borrower;

**Theory: Calculating Fines for Borrowed Books**

In a library management system, it is essential to track borrowed books and manage fines associated with overdue returns. The SQL procedure calc\_fine serves this purpose by calculating fines based on the number of days a book has been overdue.

**Components of the Database**

**Borrower Table:**

This table stores details about individuals who borrow books, including a unique Roll number, the borrower's Name, the DateofIssue, the NameofBook borrowed, and the Status of the borrowing (e.g., Issued or Returned).

The Roll column is set as the primary key and is auto-incremented to ensure uniqueness for each borrower.

**Fine Table:**

This table records fines applied to borrowers. It contains the borrower's Roll, the Date the fine was calculated, and the Amt (amount of the fine).

It has a foreign key constraint that links the Roll column to the Roll column in the Borrower table, ensuring referential integrity.

Fine Calculation Procedure

The procedure calc\_fine accepts two parameters: the borrower's Roll number and the name of the book (bname). It performs the following steps:

**Date Retrieval:**

The procedure first retrieves the DateofIssue for the specified borrower and book from the Borrower table. If the book's status is not 'Returned', the procedure proceeds.

Date Difference Calculation:

It calculates the difference in days between the current date and the DateofIssue.

**Fine Logic:**

Based on the number of overdue days (diff), the procedure determines the fine amount:

If the difference is less than zero, an error is signaled for an invalid date calculation.

If the book is overdue for more than 15 days but less than 30 days, a fine of $5 per day is applied.

If the book is overdue for more than 30 days, a fine structure is applied that calculates a higher fee for days beyond the first 30 days.

If the book is not overdue (15 days or fewer), no fine is applied.

Data Insertion and Update:

The calculated fine is inserted into the Fine table.

The status of the borrowed book in the Borrower table is updated to 'Returned'.

**Error Handling:**

The procedure includes error handling to manage exceptions that may occur during execution, such as SQL errors or invalid date calculations, ensuring that the system remains robust.