7. Stored Procedure : create stored procedure to calculate fine of students.

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

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

• Accept Roll\_no and 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. 5 per day.

• If no. of days > 30, per day fine will be Rs 50 per day and for days less than 30, Rs. 5per 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

CREATE TABLE borrower(roll\_no NUMBER , name VARCHAR2(25), dateofissue DATE,name\_of\_book

VARCHAR2(25), status VARCHAR2(20));

CREATE TABLE fine(roll\_no NUMBER,date\_of\_return DATE,amt NUMBER);

INSERT INTO borrower VALUES(45,'ASHUTOSH',TO\_DATE('01-08-2022','DD-MM-YYYY'),'HARRY

POTTER','PENDING');

INSERT INTO borrower VALUES(46,'ARYAN',TO\_DATE('15-08-2022','DD-MM-YYYY'),'DARK MATTER','PENDING');

INSERT INTO borrower VALUES(47,'ROHAN',TO\_DATE('24-08-2022','DD-MM-YYYY'),'SILENT HILL','PENDING');

INSERT INTO borrower VALUES(48,'SANKET',TO\_DATE('26-08-2022','DD-MM-YYYY'),'GOD OF WAR','PENDING');

INSERT INTO borrower VALUES(49,'SARTHAK',TO\_DATE('09-09-2022','DD-MM-YYYY'),'SPIDER-MAN','PENDING');

create or replace procedure proc\_fine

(i\_roll\_no in number,

name\_of\_book in borrower.name\_of\_book%type,

return\_date in DATE)

as

temp NUMBER;

doi DATE;

fine NUMBER;

no\_of\_days NUMBER;

begin

--dbms\_output.put\_line(return\_date);

SELECT to\_date(borrower.dateofissue,'DD-MM-YYYY') INTO doi FROM borrower WHERE

borrower.roll\_no = i\_roll\_no AND borrower.name\_of\_book = name\_of\_book;

no\_of\_days := return\_date-doi;

dbms\_output.put\_line(no\_of\_days);

IF (no\_of\_days >15 AND no\_of\_days <=30) THEN

fine := 5\*no\_of\_days;

ELSIF (no\_of\_days>30 ) THEN

temp := no\_of\_days-30;

fine := 150 + temp\*50;

END IF;

dbms\_output.put\_line(fine);

INSERT INTO fine VALUES(i\_roll\_no,return\_date,fine);

UPDATE borrower SET status = 'RETURNED' WHERE borrower.roll\_no = i\_roll\_no;

exception

when no\_data\_found then

dbms\_output.put\_line('Book ' || name\_of\_book ||' not found');

end;

/

DECLARE

i\_roll\_no NUMBER;

name\_of\_book VARCHAR2(25);

return\_date DATE := TO\_DATE(SYSDATE,'DD-MM-YYYY');

BEGIN

i\_roll\_no := &i\_roll\_no;

name\_of\_book := &name\_of\_book;

proc\_fine(i\_roll\_no,name\_of\_book,return\_date);

END;

/

Alternate code

-- Creating the Borrower table

CREATE TABLE Borrower (

Roll\_no INT PRIMARY KEY,

Name VARCHAR2(50),

DateofIssue DATE,

NameofBook VARCHAR2(50),

Status CHAR(1) -- 'I' for Issued, 'R' for Returned

);

-- Creating the Fine table

CREATE TABLE Fine (

Roll\_no INT,

FineDate DATE, -- Renamed from 'Date' to 'FineDate'

Amt NUMBER,

CONSTRAINT fk\_borrower FOREIGN KEY (Roll\_no) REFERENCES Borrower(Roll\_no)

);

-- Insert sample records into Borrower table

INSERT INTO Borrower (Roll\_no, Name, DateofIssue, NameofBook, Status) VALUES (1, 'Alice', SYSDATE - 20, 'Oracle SQL', 'I');

INSERT INTO Borrower (Roll\_no, Name, DateofIssue, NameofBook, Status) VALUES (2, 'Bob', SYSDATE - 35, 'Database Concepts', 'I');

INSERT INTO Borrower (Roll\_no, Name, DateofIssue, NameofBook, Status) VALUES (3, 'Charlie', SYSDATE - 10, 'PL/SQL Programming', 'I');

COMMIT;

drop table borrower cascade constraint;

-- Creating the stored procedure to calculate fine with additional conditions

CREATE OR REPLACE PROCEDURE calculate\_fine (

p\_roll\_no IN Borrower.Roll\_no%TYPE,

p\_nameofbook IN Borrower.NameofBook%TYPE

)

IS

v\_days\_due NUMBER;

v\_fine\_amt NUMBER := 0;

overdue\_exception EXCEPTION;

BEGIN

-- Find the number of days since the book was issued

SELECT TRUNC(SYSDATE - DateofIssue)

INTO v\_days\_due

FROM Borrower

WHERE Roll\_no = p\_roll\_no AND NameofBook = p\_nameofbook;

-- Calculate fine based on days overdue

IF v\_days\_due > 15 AND v\_days\_due <= 30 THEN

v\_fine\_amt := (v\_days\_due - 15) \* 5;

ELSIF v\_days\_due > 30 THEN

v\_fine\_amt := (30 - 15) \* 5 + (v\_days\_due - 30) \* 50;

END IF;

-- Update the book status to 'R' for returned

UPDATE Borrower

SET Status = 'R'

WHERE Roll\_no = p\_roll\_no AND NameofBook = p\_nameofbook;

-- Insert fine details into Fine table if a fine exists

IF v\_fine\_amt > 0 THEN

INSERT INTO Fine (Roll\_no, FineDate, Amt)

VALUES (p\_roll\_no, SYSDATE, v\_fine\_amt);

END IF;

-- Display the result

DBMS\_OUTPUT.PUT\_LINE('Fine calculated for Roll No: ' || p\_roll\_no || ' is Rs. ' || v\_fine\_amt);

EXCEPTION

-- Named exception handling for no data found

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No record found for Roll No: ' || p\_roll\_no || ' and Book: ' || p\_nameofbook);

-- Named exception for overdue fine calculation errors (optional, can be raised manually if needed)

WHEN overdue\_exception THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred while calculating the fine for Roll No: ' || p\_roll\_no);

-- General exception handler for any other errors

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An unexpected error occurred. Error code: ' || SQLCODE || ' Error message: ' || SQLERRM);

END calculate\_fine;

/

-- Enable DBMS\_OUTPUT to see messages

SET SERVEROUTPUT ON;

-- Test the procedure with user input

BEGIN

-- Accept user input for Roll\_no and NameofBook

calculate\_fine(&Roll\_no, '&NameofBook');

END;

/

select \* from fine;