1. **Write an SQL code block** these raise a user defined exception where business rule is voilated. BR for client\_ master table specifies when the value of bal\_due field is less than 0 handle the exception.

# Write an SQL code block

Borrow(Roll\_no, Name, DateofIssue, NameofBook, Status) 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 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.

a) SQL Code to Raise a User-Defined Exception for Client\_Master Table:

```sql

-- Assuming you have a table named client\_master with columns Roll\_no and bal\_due

-- Define a user-defined exception

DECLARE

neg\_balance\_exception EXCEPTION;

PRAGMA EXCEPTION\_INIT(neg\_balance\_exception, -20001);

BEGIN

-- Accept input from the user

DECLARE

v\_roll\_no NUMBER;

v\_bal\_due NUMBER;

BEGIN

-- Get the roll number and balance due from the user

v\_roll\_no := &roll\_no;

v\_bal\_due := &bal\_due;

-- Check if the balance due is less than 0

IF v\_bal\_due < 0 THEN

-- Raise the user-defined exception

RAISE neg\_balance\_exception;

ELSE

-- Perform other actions if the balance is valid

-- Your business logic here

END IF;

EXCEPTION

WHEN neg\_balance\_exception THEN

DBMS\_OUTPUT.PUT\_LINE('User-defined Exception: Balance due is less than 0.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

END;

/

```

In the above SQL code, we define a user-defined exception `neg\_balance\_exception` and use it to raise an exception when the `bal\_due` field is less than 0. We handle this exception and provide an appropriate error message.

b) SQL Code to Calculate Fine and Handle Exceptions for Borrow Table:

```sql

-- Assuming you have a table named Borrow with columns Roll\_no, DateofIssue, NameofBook, and Status

-- And a table named Fine with columns Roll\_no, Date, and Amt

-- Define a user-defined exception

DECLARE

fine\_exception EXCEPTION;

PRAGMA EXCEPTION\_INIT(fine\_exception, -20002);

-- Define variables

v\_roll\_no NUMBER;

v\_name\_of\_book VARCHAR2(100);

v\_date\_of\_issue DATE;

v\_status CHAR(1);

v\_fine\_amt NUMBER;

BEGIN

-- Accept input from the user

v\_roll\_no := &roll\_no;

v\_name\_of\_book := '&name\_of\_book';

v\_date\_of\_issue := SYSDATE; -- Assuming current date for date of issue

v\_status := 'I'; -- Initial status

-- Calculate the number of days since the date of issue

DECLARE

v\_days\_passed NUMBER;

BEGIN

SELECT TRUNC(SYSDATE - v\_date\_of\_issue) INTO v\_days\_passed FROM DUAL;

-- Check the number of days and calculate the fine amount

IF v\_days\_passed BETWEEN 15 AND 30 THEN

v\_fine\_amt := v\_days\_passed \* 5;

ELSIF v\_days\_passed > 30 THEN

v\_fine\_amt := v\_days\_passed \* 50;

ELSE

v\_fine\_amt := 0; -- No fine

END IF;

-- Change the status to 'R' after submitting the book

v\_status := 'R';

-- Insert fine details into the Fine table

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

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

-- Commit the transaction

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Book not found');

WHEN fine\_exception THEN

DBMS\_OUTPUT.PUT\_LINE('User-defined Exception: Fine calculation error.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

END;

/

```

In the above SQL code, we first calculate the number of days since the date of issue and then calculate the fine amount based on the number of days. We also handle exceptions and insert fine details into the Fine table.