15:

```
CREATE TABLE client_master (client_id INT PRIMARY KEY, client_name VARCHAR(100),
bal_due DECIMAL(10, 2));
INSERT INTO client_master (client_id, client_name, bal_due) VALUES (1, 'Alice',
200.50), (2, 'Bob', -50.00), (3, 'Charlie', 150.75), (4, 'David', 100.00), (5,
'Eva', -10.00);

DELIMITER $$
CREATE PROCEDURE CheckBalanceDue()
BEGIN
    DECLARE client_balance DECIMAL(10, 2);
    DECLARE client_id INT;
    DECLARE client_name VARCHAR(100);
    DECLARE message_text VARCHAR(255);
    DECLARE client_cursor CURSOR FOR
        SELECT client_id, client_name, bal_due FROM client_master;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET client_balance = NULL;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
        BEGIN
            -- When an exception is raised (negative balance), continue processing
            SELECT 'Balance due is negative for a client, skipping to next.' AS
message;
        END;
    OPEN client_cursor;
    FETCH client_cursor INTO client_id, client_name, client_balance;
    WHILE client_balance IS NOT NULL DO
        IF client_balance < 0 THEN
            -- Set the custom exception message including client name
            SET message_text = CONCAT('Balance due is negative for client: ',
client_name, ' (Client ID: ', client_id, ')');
            -- Output client details where balance due is negative
            SELECT client_id, client_name, client_balance
            FROM client_master
            WHERE client_id = client_id;
            -- Raise a custom exception with the message
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = message_text;
        END IF;
        FETCH client_cursor INTO client_id, client_name, client_balance;
    END WHILE;
    CLOSE client_cursor;
END $$
DELIMITER ;
CALL CheckBalanceDue();

CREATE TABLE Borrow (Roll_no INT PRIMARY KEY, Name VARCHAR(100), DateofIssue DATE,
NameofBook VARCHAR(100), Status VARCHAR(1));
CREATE TABLE Fine (Roll_no INT, Date DATE, Amt DECIMAL(10, 2), PRIMARY KEY
(Roll_no, Date));
INSERT INTO Borrow (Roll_no, Name, DateofIssue, NameofBook, Status) VALUES (1,
'Alice', '2024-10-01', 'Java Programming', 'I'), (2, 'Bob', '2024-09-15', 'Python
Basics', 'I'), (3, 'Charlie', '2024-08-10', 'Data Structures', 'I'), (4, 'David',
'2024-07-20', 'Web Development', 'I'), (5, 'Eva', '2024-10-10', 'Machine Learning',
'I');
INSERT INTO Fine (Roll_no, Date, Amt) VALUES (1, '2024-10-15', 25.00);

DELIMITER $$
CREATE PROCEDURE HandleFine(Roll_no INT, BookName VARCHAR(100))
BEGIN
    -- Declare variables first
```

```
    DECLARE fine_amount DECIMAL(10, 2);
    DECLARE fine_days INT;
    DECLARE due_date DATE;
    DECLARE current_date_var DATE;  -- Changed variable name
    DECLARE fine_rate DECIMAL(10, 2);

    -- Declare the exception handler after variables
    DECLARE CONTINUE HANDLER FOR NOT FOUND
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No record found for the
specified book and roll number';
    -- Get current date
    SET current_date_var = CURDATE();  -- Updated variable name
    -- Fetch the date of issue for the specified book and Roll_no
    SELECT DateofIssue INTO due_date
    FROM Borrow
    WHERE Roll_no = Roll_no AND NameofBook = BookName;
    -- Calculate the number of days the book is overdue
    SET fine_days = DATEDIFF(current_date_var, due_date);  -- Updated variable name
    -- Calculate fine based on the number of overdue days
    IF fine_days BETWEEN 15 AND 30 THEN
        SET fine_rate = 5.00;  -- Rs. 5 per day if between 15 and 30 days overdue
    ELSEIF fine_days > 30 THEN
        SET fine_rate = 50.00;  -- Rs. 50 per day if more than 30 days overdue
    ELSE
        SET fine_rate = 0.00;  -- No fine if less than 15 days
    END IF;
    -- If fine is greater than 0, store it in the Fine table
    IF fine_rate > 0 THEN
        SET fine_amount = fine_rate * fine_days;
        INSERT INTO Fine (Roll_no, Date, Amt)
        VALUES (Roll_no, current_date_var, fine_amount);  -- Updated variable name
    END IF;
    -- If book is returned, update status from 'I' to 'R'
    UPDATE Borrow
    SET Status = 'R'
    WHERE Roll_no = Roll_no AND NameofBook = BookName;
    -- Output the fine amount if any
    IF fine_amount > 0 THEN
        SELECT fine_amount AS Fine_Amount;
    ELSE
        SELECT 'No fine' AS Fine_Amount;
    END IF;
END $$
DELIMITER ;
CALL HandleFine(1, 'Java Programming');
```