**Assignment\_part\_3\_Rountines**

1. Write a stored procedure that accepts the month and year as inputs and prints the ordernumber, orderdate and status of the orders placed in that month. The month should be abbreviated to three characters.

Example:

Input: month -> 'Feb'

year -> 2003

Output:

+------------+---------+

| orderdate | status |

+------------+---------+

| 2003-02-11 | Shipped |

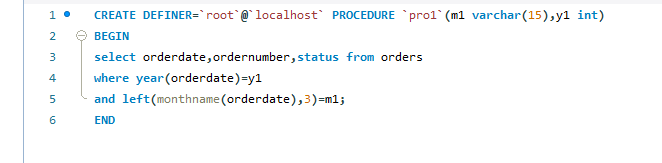
| 2003-02-17 | Shipped |

| 2003-02-24 | Shipped |

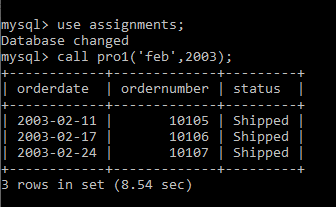
+------------+---------+

3 rows in set (0.001 sec)

**Query:**

****

**Output:**

****

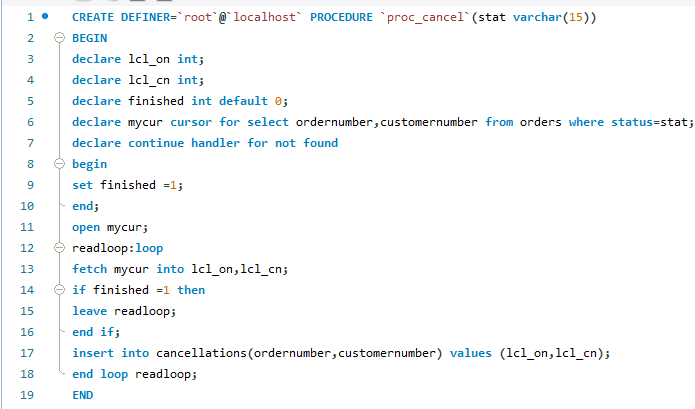
2. Write a stored procedure to insert a record into the cancellations table for all cancelled orders.

STEPS: a. Create a table called cancellations with the following fields

id (primary key), custumernumber (foreign key), ordernumber (foreign key), comments

All values except id should be taken from the order table.

b. Read through the orders table . If an order is cancelled, then put an entry in the cancellations table.

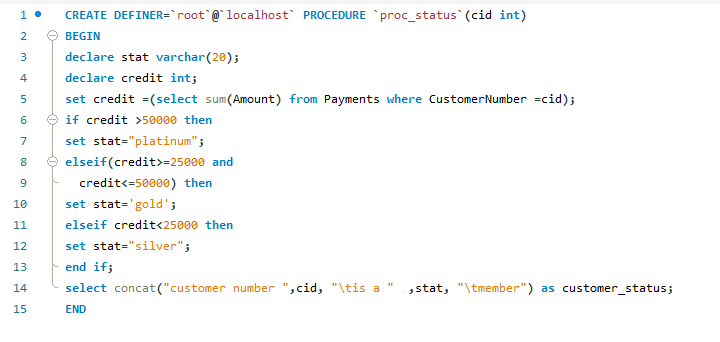
**Query:** ****

3. a. Write function that takes the customernumber as input and returns the purchase\_status based on the following criteria . [table:Payments]

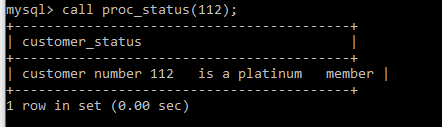
if the total purchase amount for the customer is < 25000 status = Silver, amount between 25000 and 50000, status = Gold

if amount > 50000 Platinum

b. Write a query that displays customerid, customername and purchase\_status

**Query:** ****

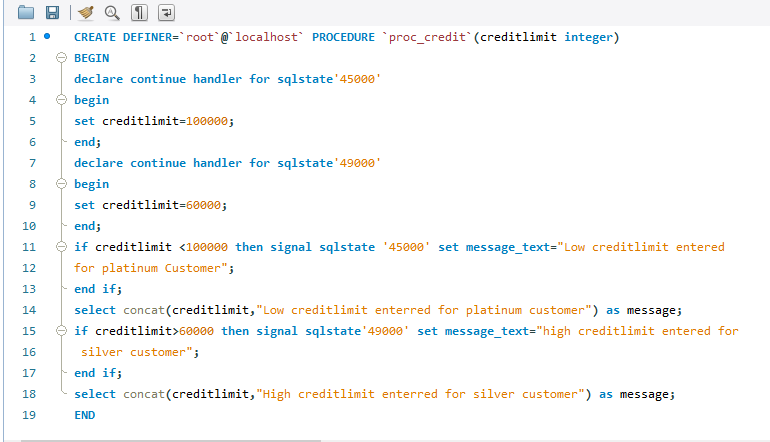
**Output:**

****

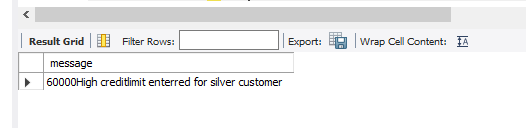
4. Write a stored procedure that checks the creditlimit and the purchase status of the customers.

If a platinum customer has crediltlimit less than 100,000 raise an exception. In the exception handler update the crediltlimit to 100000.

If a silver customer has creditlimit greater than 60,000 raise an exception. In the exception handler update the crediltlimit to 60000.

**Query:** ****

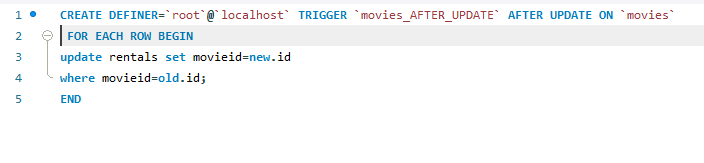
**Output:**

****

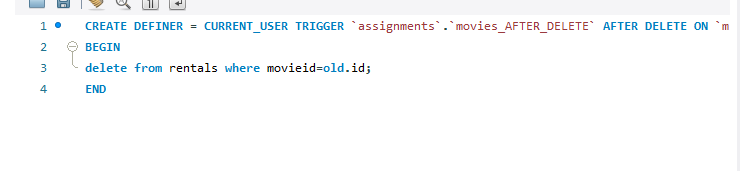
5. Replicate the functionality of 'on delete cascade' and 'on update cascade' using triggers on movies and rentals tables. Note: Both tables - movies and rentals - don't have primary or foreign keys. Use only triggers to implement the above.

**Query:**

**On Update cascade:**

****

**On Delete cascade:**

****