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)

mysql> delimiter /

mysql> ;

-> /

ERROR 1065 (42000): Query was empty

mysql> CREATE PROCEDURE GETSTATUS(IN ORDER\_MONTH VARCHAR(10), IN ORDER\_YEAR INT)

-> BEGIN

-> SELECT ORDERNUMBER,ORDERDATE,STATUS FROM ORDERS WHERE YEAR(ORDERDATE) = ORDER\_YEAR AND LEFT(monthname(ORDERDATE),3)=ORDER\_MONTH;

-> END

-> ;

-> ;

-> /

Query OK, 0 rows affected (0.89 sec)

mysql> CALL GETSTATUS('FEB',2003);

-> /

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

| ORDERNUMBER | ORDERDATE | STATUS |

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

| 10105 | 2003-02-11 | Shipped |

| 10106 | 2003-02-17 | Shipped |

| 10107 | 2003-02-24 | Shipped |

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

3 rows in set (0.23 sec)

Query OK, 0 rows affected (0.27 sec)

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.

mysql> CREATE TABLE CANCELLATIONS(

-> INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,

-> CUSTOMERNUMBER INT,

-> ORDERNUMBER INT,

-> COMMENTS VARCHAR(100),

-> CONSTRAINT FK\_CUST FOREIGN KEY (CUSTOMERNUMBER) REFERENCES ORDERS(customerNumber),

-> CONSTRAINT FK\_ORDER FOREIGN KEY (ORDERNUMBER) REFERENCES ORDERS(ORDERNUMBER));

-> /

Query OK, 0 rows affected (2.65 sec)

mysql> select \* from cancellations;

-> /

Empty set (0.01 sec)

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

mysql> CREATE PROCEDURE CANCEL\_ORDER()

-> BEGIN

->

-> INSERT INTO CANCELLATIONS(CUSTOMERNUMBER,ORDERNUMBER)

-> SELECT CUSTOMERNUMBER,ORDERNUMBER FROM ORDERS

-> WHERE STATUS = 'CANCELLED';

->

-> END;

-> /

Query OK, 0 rows affected (0.25 sec)

mysql> CALL CANCEL\_ORDER();

-> /

Query OK, 6 rows affected (0.52 sec)

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

mysql> CREATE FUNCTION CUSTOMER\_LEVEL(CUST\_NUM INT)

-> RETURNS VARCHAR(20)

-> DETERMINISTIC

-> BEGIN

-> DECLARE CREDIT DECIMAL(10,2)DEFAULT 0;

-> DECLARE CUST\_LEVEL VARCHAR(20);

-> SELECT SUM(AMOUNT) INTO CREDIT

-> FROM PAYMENTS

-> WHERE CUSTOMERNUMBER = CUST\_NUM;

-> IF CREDIT>50000 THEN

-> SET CUST\_LEVEL = 'PLATINUM';

-> ELSEIF(CREDIT>=2500 AND CREDIT <=50000) THEN

-> SET CUST\_LEVEL = 'GOLD';

-> ELSEIF CREDIT<2500 THEN

-> SET CUST\_LEVEL = 'SILVER';

-> END IF;

-> RETURN(CUST\_LEVEL);

-> END;

-> /

Query OK, 0 rows affected (0.36 sec)

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

mysql> SELECT CUSTOMERNUMBER , AMOUNT,CUSTOMER\_LEVEL(CUSTOMERNUMBER) FROM PAYMENTS;

-> /

mysql> describe payments;

-> /

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

| Field | Type | Null | Key | Default | Extra |

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

| customerNumber | int | NO | PRI | NULL | |

| checkNumber | varchar(50) | NO | PRI | NULL | |

| paymentDate | date | NO | | NULL | |

| amount | decimal(10,2) | NO | | NULL | |

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

4 rows in set (0.31 sec)

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.

mysql> CREATE PROCEDURE CHECKSTATUS(IN CREDIT DECIMAL(10,2), IN STATUS VARCHAR(100))

-> BEGIN

-> DECLARE Message VARCHAR(1000);

->

-> BEGIN

->

-> IF CREDIT < 100000 AND STATUS ='PLATINUM' THEN

-> SELECT ('UPDATE THE CREDIT LIMIT TO 100000 !!!') AS message;

->

-> ELSEIF CREDIT > 60000 AND STATUS ='SILVER' THEN

-> SELECT ('UPDATE THE CREDIT LIMIT TO BELOW 60000 !!!') AS message;

-> END IF;

->

-> END;

->

-> END;

-> /

Query OK, 0 rows affected (0.22 sec)

mysql> CALL CHECKSTATUS(1000,'PLATINUM');

-> /

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

| message |

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

| UPDATE THE CREDIT LIMIT TO 100000 !!! |

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

1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> CALL CHECKSTATUS(100000,'SILVER');/

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

| message |

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

| UPDATE THE CREDIT LIMIT TO BELOW 60000 !!! |

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

1 row in set (0.00 sec)

Query OK, 0 rows affected (0.03 sec)

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.

Update -

mysql> CREATE DEFINER=`root`@`localhost` TRIGGER upd\_cascade

-> AFTER UPDATE ON `movies`

-> FOR EACH ROW

-> BEGIN

-> update rentals set movieid = new.id

-> where movieid = old.id;

-> END;

-> /

Query OK, 0 rows affected (0.25 sec)

Delete -

mysql> CREATE DEFINER=`root`@`localhost` TRIGGER del\_cascade

->

-> AFTER DELETE ON `movies`

-> FOR EACH ROW

-> BEGIN

-> delete from rentals

-> where movieid = old.id;

-> END;/

Query OK, 0 rows affected (0.26 sec)