SET 3

-----

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.

***Example***: call order\_status(2005, 11);

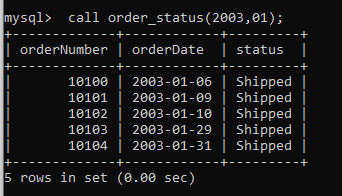
Procedure:

CREATE DEFINER=`root`@`localhost` PROCEDURE `order\_status`(yea integer, mon integer)

BEGIN

select orderNumber, orderDate, status from orders where year(orderDate) = yea and month(orderDate) = mon;

END



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

STEPS:

1. Create a table called cancellations with the following fields

id (primary key),

customernumber (foreign key - Table customers),

ordernumber (foreign key - Table Orders),

comments

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

create table cancellations (id integer primary key auto\_increment,

customernumber integer,

ordernumber integer,

comments varchar(255),

foreign key(customernumber) references customers(customerNumber),

foreign key(ordernumber) references orders(orderNumber)

on delete cascade

on update cascade);

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

CREATE DEFINER=`root`@`localhost` PROCEDURE `cancelation`()

BEGIN

declare lcl\_custno integer;

declare lcl\_ordno integer;

declare lcl\_comm varchar(15);

declare not\_found integer default 0;

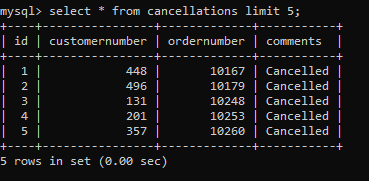
declare mycur cursor for select c.customerNumber,o.orderNumber,o.status from customers as c inner join orders as o on o.customerNumber = c.customerNumber where o.status = "Cancelled";

declare continue handler for not found

begin

set not\_found =1;

end;



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

CREATE DEFINER=`root`@`localhost` FUNCTION `pur\_status`(i integer, amount integer) RETURNS varchar(255) CHARSET utf8mb4

BEGIN

DECLARE customerLevel VARCHAR(20);

if amount < 25000 then

SET customerLevel = 'Silver';

elseif amount > 50000 then

SET customerLevel = "Platinum";

else

SET customerLevel = "Gold";

end if;

RETURN (customerLevel);

END

1. Write a query that displays customerNumber, customername and purchase\_status from customers table.

select c.customerNumber,c.customerName,p.amount,pur\_status(c.customerNumber, p.amount) as Status from customers as c inner join payments as p on c.customerNumber = p.customerNumber;

4. 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.

For on Update cascade.

CREATE TRIGGER trg\_movies\_update

AFTER update ON movies

FOR EACH ROW

BEGIN

UPDATE rentals

SET movieid = id

WHERE movieid = OLD.id ;

END;

For on Delete cascade.

CREATE TRIGGER trg\_movies\_delete

AFTER DELETE ON movies

FOR EACH ROW

BEGIN

DELETE FROM rentals

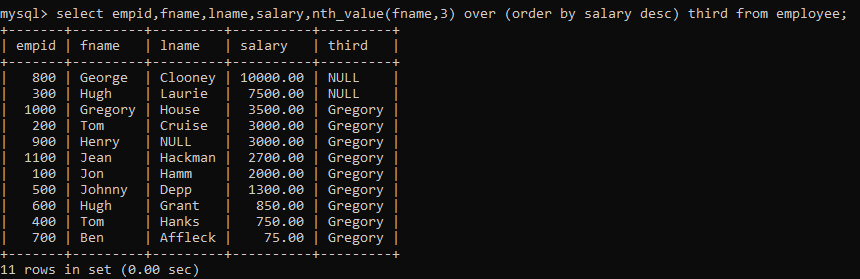
WHERE movieid

NOT IN (SELECT DISTINCT id FROM movies);

END;

5. Select the first name of the employee who gets the third highest salary. [table: employee]

select empid,fname,lname,salary,nth\_value(fname,3) over (order by salary desc) third from employee;



6. Assign a rank to each employee based on their salary. The person having the highest salary has rank 1. [table: employee]

select empid,fname,lname,salary,rank() over (order by salary desc) as rnk from employee;

