Question 1

Soln:

select department\_id, min(salary) "Second Lowest salary"

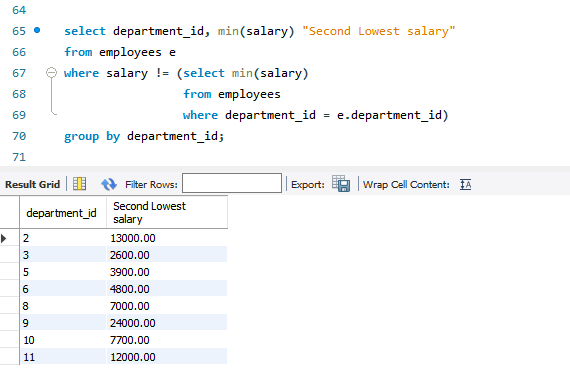
from employees e

where salary != (select min(salary)

from employees

where department\_id = e.department\_id)

group by department\_id;



Question 2.1

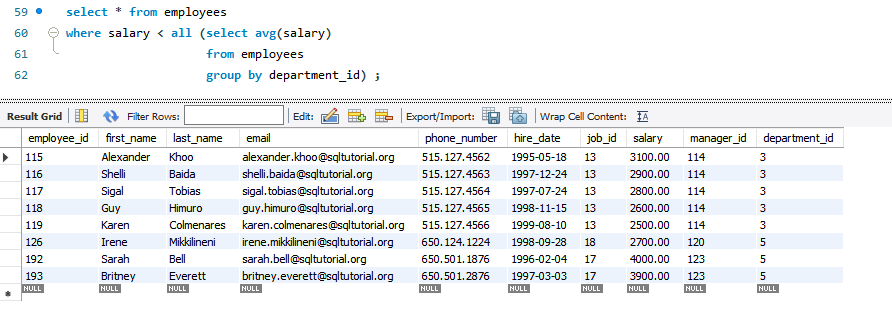
Soln:

select \* from employees

where salary < all (select avg(salary)

from employees

group by department\_id) ;



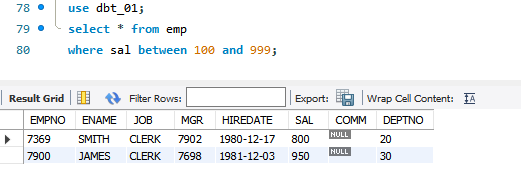
Question 2.2

Soln:

use dbt\_01;

select \* from emp

where sal between 100 and 999;



Question 3:

Soln:

select concat(e.first\_name,' ', e.last\_name) 'Full Name', d.department\_name, e.job\_id, concat(f.first\_name, ' ', f.last\_name) 'Manager'

from employees e

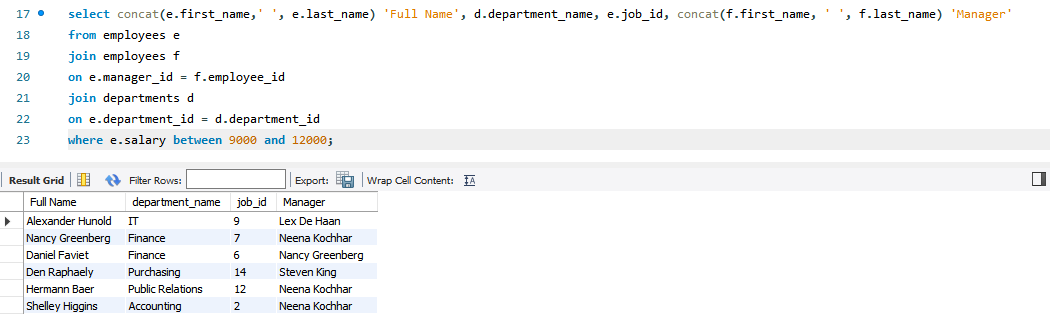
join employees f

on e.manager\_id = f.employee\_id

join departments d

on e.department\_id = d.department\_id

where e.salary between 9000 and 12000;



Qurstion 4

Soln:

select concat(first\_name, last\_name), department\_name, hire\_date, salary

from employees e

join departments d

on e.department\_id=d.department\_id

where datediff(sysdate(), hire\_date)/365>12;

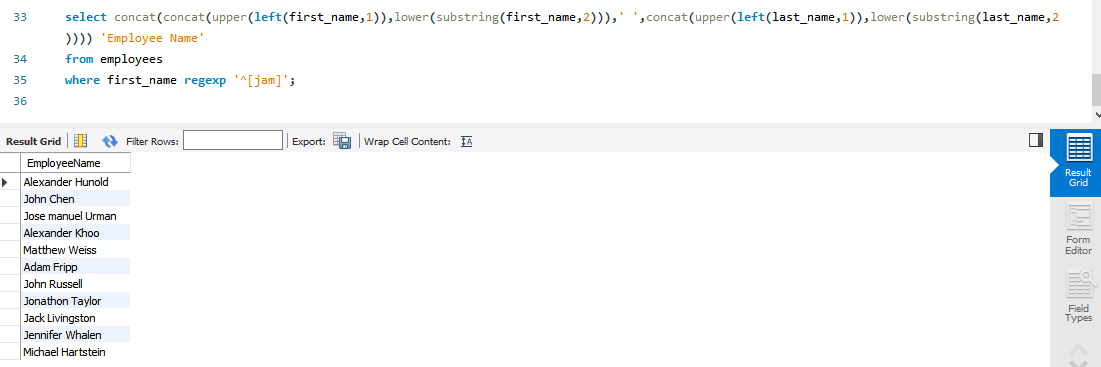
Question 5

Soln:

Select concat(concat(upper(left(first\_name,1)),lower(substring(first\_name,2))),' ',concat(upper(left(last\_name,1)),lower(substring(last\_name,2)))) 'Employee Name'

from employees

where first\_name regexp '^[jam]';



Question 6

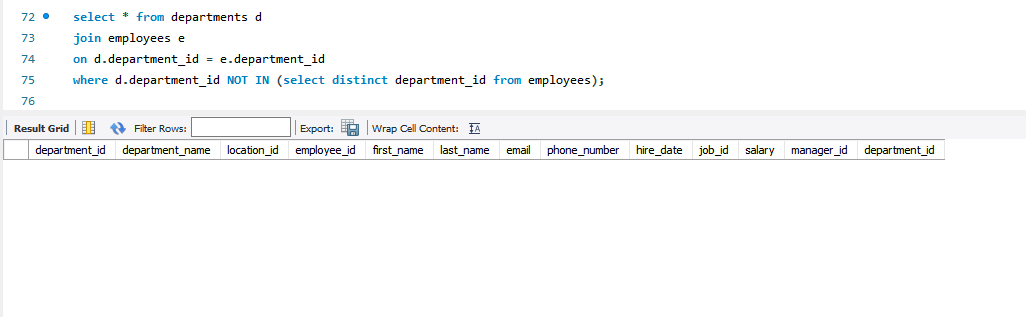
Soln:

select \* from departments d

join employees e

on d.department\_id = e.department\_id

where d.department\_id NOT IN (select distinct department\_id from employees);

 Empty SET

Question 7

Soln:

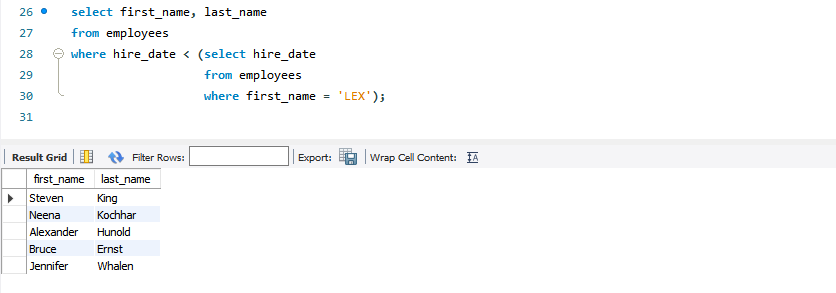
select first\_name, last\_name

from employees

where hire\_date < (select hire\_date

from employees

where first\_name = 'LEX');



Question 8

Soln:

delimiter #

drop procedure if exists proc\_const\_pk#

create procedure proc\_const\_pk(IN p\_table\_name varchar(100), IN p\_column\_name varchar(100))

begin

set @primkey = concat('alter table ',p\_table\_name,' add constraint pk\_on\_table primary key(',p\_column\_name,')');

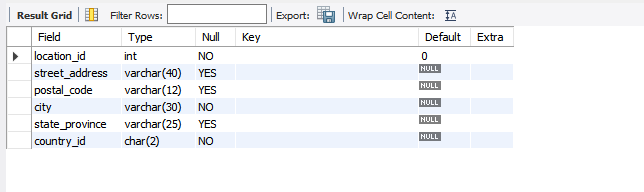
prepare stmt from @primkey;

execute stmt;

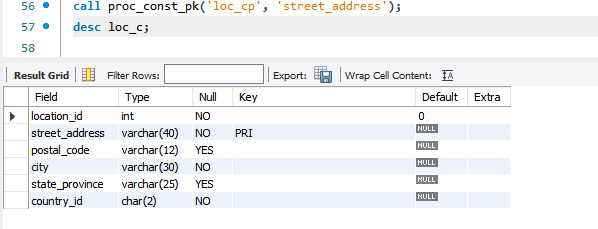
end#

delimiter ;

Before running the procedure:



After running the procedure:



Question 9

Soln:

DELIMITER #

drop function if exists card\_verification #

create function card\_verification(sal int)

returns varchar(50)

deterministic

begin

if sal >= 1000 and sal <= 10000 then return 'SILVER CARD WITH 50000 LIMIT';

elseif sal > 10000 and SAL <= 20000 then return 'GOLD CARD WITH 150000 LIMIT';

elseif sal > 20000 then return 'PLATINUM CARD WITH 300000 LIMIT';

else

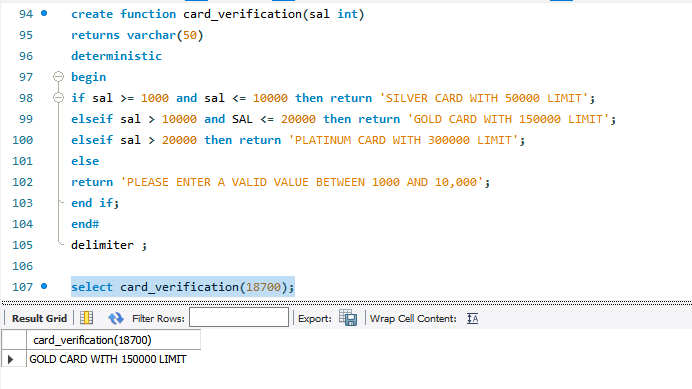
return 'PLEASE ENTER A VALID VALUE BETWEEN 1000 AND 10,000';

end if;

end#

delimiter ;

select card\_verification(18700);



Question 10

Soln:

drop view if exists view\_employee\_details;

create view view\_employee\_details as

select concat(e.first\_name, ' ',e.last\_name) 'Employee Name', d.department\_name 'Department', j.job\_title 'Job', j.min\_salary 'Min. Salary', j.max\_salary 'Max. Salary'

from departments d

join employees e

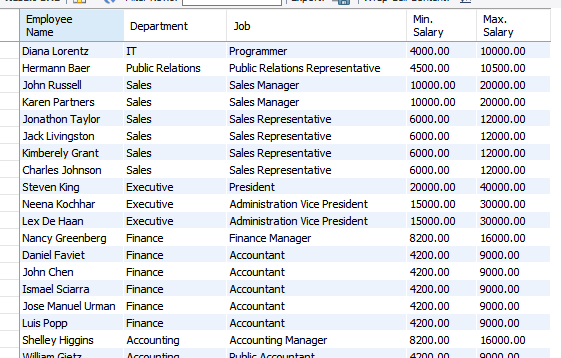
on e.department\_id = d.department\_id

join jobs j

on e.job\_id = j.job\_id;

select \* from view\_employee\_details;





Question 11

Soln:

drop table if exists sales;

create table sales (sales\_id INT PRIMARY KEY, product VARCHAR(50), QUANTITY INT, FYEAR DATE);

CREATE TABLE SALE\_CHANGES (ID\_OLD INT , Id\_new int, product\_old varchar(50), product\_new varchar(50), quantity\_old INT, quantity\_new INT, fyear\_old DATE, fyear\_new DATE);

Delimiter #

CREATE TRIGGER trig\_sales

After UPDATE

ON sales

FOR EACH ROW

BEGIN

insert into sale\_changes values(old.id, new.id, old.product, new.product, old.Quantity, new.quantity, old.fyear, new.fyear);

END#

delimiter ;

insert into sales values (1, 'cycle', 5, current\_date());

select \* from sales;

update sales set product = 'Bike' where id = 1;

select \* from sale\_changes;

