create table account(accno int unique auto\_increment , accname varchar(10));

Insert into account(accname) values('a'),('b'),('c');

Insert into account values(10,'d');

Insert into account(accname) values('a'),('b'),('c');

Start transaction;

Insert into account(accname) values('a'),('b'),('c');

Truncate table account;

alter table account auto\_increment=200000;

select \* from emp e join dept d on e.deptid=d.deptid where(select count(e2.salary) from emp e2 where e2.salary>e.salary and e2.deptid=e.deptid)<5 order by deptname, salary desc;

Select

Deptname,e1.ename,e1.salary ,count(\*)

From

Emp e1, emp e2, dept d

Where e1.deptid=e2.deptid

And e1.salary<=e2.salary

And e1.deptid=d.deptid

Group by Deptname,e1.ename,e1.salary

Having count(\*)<=5

Order by deptname,e1.salary desc;

Select

Deptname,e1.ename,e1.salary ,count(\*)

From

Emp e1 join emp e2

On e1.deptid=e2.deptid

join dept d

On e1.deptid=d.deptid

Where e1.salary<=e2.salary

Group by Deptname,e1.ename,e1.salary

Having count(\*)<=5

Order by deptname,e1.salary desc;

Select round(10.3232444);

Select round(10.3232444,2);

Select round(10.3232444,-2);

Select floor(10.9);

Select floor(-10.9);

Select ceiling(10.9);

Select ceiling(-10.9);

Select mod(10,3);

Select abs(-100);

Select round(48/5)\*5;

Select substring('abcdef',2,3);

Select substring('abcdef',2);

Select substring('abcdef',-2);

Select substring('abcdef',-2,-1);

Select instr('abcdef','ef');

Select instr('abcdef','efz');

Select instr('abcdefdsdefef','ef');

Select replace('asadvdfdcdc','c','z');

Select left('abcdef',2);

Select right('abcdef',2);

Select ltrim(' abcdef ');

Select rtrim(' abcdef ');

Select trim(' abcdef ');

Select concat('b','aas','ssas');

Select reverse('abcdd');

Select

length(substring('abcdefdsdefef',1,instr('abcdefdsdefef','ef')+length('ef')-1))+

instr(substring('abcdefdsdefef',instr('abcdefdsdefef','ef')+length('ef')),'ef');

CTE- common table expressions

With Queries

With t

as

(Select \* from emp)

Select \* from t;

With t

as

(Select 1 id )

Select \* from t;

Set @@cte\_max\_recursion\_depth=200000;

With recursive t

as

(Select 1 id

Union all

Select id+1 from t

Where id<2000)

Select \* from t;

– insert into 10000 to emp table as below

Eid- 1,2,3,4,5…

Ename- a1,b2,c3,...z26,a27,b28…

salary- 121212,43434,1212,435,776666,2132

Deptid-1,2,3,1,2,3,1,2,3,1,2,3,1,2,3

Select ascii('a');

Select mod(0,26)+97

Truncate table emp;

Insert into emp

With recursive t

as

(Select 1 eid ,

concat(convert(char(mod(0,26)+97), char(10)),1) ename,rand()\*100000 salary,mod(0,3)+1 deptid

Union all

Select eid+1,

concat(convert(char(mod(eid,26)+97), char(10)),eid+1) ,rand()\*100000,

mod(eid,3)+1 deptid

from t

Where eid<100000)

Select \* from t;

1. List all the columns of the Salespeople table.

Desc salespeople;

2. List all customers with a rating of 100.

Select \* from customers where rating=100;

3. Find all records in the Customer table with NULL values in the city column.

Select \* from customers where city is null;

4. Find the largest order taken by each salesperson on each date.

Select sname,odate,max(amt) max\_amt

From orders o , salespeople s

Where o.snum=s.snum

Group by sname,odate;

With t

As

(Select sname,odate,onum,amt,

rank() over(partition by sname,odate

Order by amt desc) rn

From orders o , salespeople s

Where o.snum=s.snum

)

Select \* from t where rn=1;

5. Arrange the Orders table by descending customer number.

Select \* from orders

Order by cnum desc;

6. Find which salespeople currently have orders in the Orders table.

Select \* from salespeople

Where snum in(select snum from orders);

Select distinct s.\*

From salespeople s, orders o

Where s.snum=o.snum;

7. List names of all customers matched with the salespeople serving them.

Select cname,sname

From customers c , salespeople s

Where c.snum=s.snum;

8. Find the names and numbers of all salespeople who had more than one customer.

Select sname,s.snum,count(distinct cnum)

From customers c , salespeople s

Where c.snum=s.snum

Group by sname,s.snum

Having count(distinct cnum)>1;

9. Count the orders of each of the salespeople and output the results in descending order.

Select sname,count(onum) ord\_cnt

From salespeople s, orders o

Where s.snum=o.snum

Group by sname

Order by count(onum) desc;

10. List the Customer table if and only if one or more of the customers in the Customer table are

located in San Jose.

Select \* from customers

Where exists

(select 1 from customers where city='San Jose');

Select \* from customers

Where 1 in

(select 1 from

Customers where city='San Jose');

11. Match salespeople to customers according to what city they lived in.

Select sname,cname,c.city

From customers c , salespeople s

Where c.snum=s.snum

And s.city=c.city;

12. Find the largest order taken by each salesperson.

With t

As

(Select sname,odate,onum,amt,

rank() over(partition by sname

Order by amt desc) rn

From orders o , salespeople s

Where o.snum=s.snum)

Select \* from t where rn=1;

13. Find customers in San Jose who have a rating above 200.

Select \* from customers where

city='San Jose' and rating>200;

14. List the names and commissions of all salespeople in London.

Select sname,comm

From salespeople where city='London';

15. List all the orders of salesperson Motika from the Orders table.

Select sname,o.\*

From orders o join salespeople s

On o.snum=s.snum

Where sname='Motika';

16. Find all customers with orders on October 3.

Select cname,onum,odate,amt

From orders o , customers c

Where o.cnum=c.cnum

And

odate='1996-10-03';

17. Give the sums of the amounts from the Orders table,

grouped by date, eliminating all those

dates where the SUM was not at least 2000.00 above the MAX amount.

Select odate,sum(amt) from orders group by odate

Having sum(amt)>2000+max(amt);

18. Select all orders that had amounts that were greater than at least one of the orders from

October 6.

Select \* from orders where amt>(

Select min(amt) from orders where

odate='1996-10-06');

Select \* from orders where amt>any(

Select amt from orders where

odate='1996-10-06');

19. Write a query that uses the EXISTS operator to extract all salespeople who have customers

with a rating of 300.

Select \* from salespeople s

Where exists

(select 1 from customers c

Where c.snum=s.snum

And rating=300);

20. Find all pairs of customers having the same rating.

Select c1.cname,c2.cname

From customers c1,customers c2

Where c1.cname<c2.cname

And c1.rating=c2.rating;

21. Find all customers whose CNUM is 1000 above the SNUM of Serres.

Select \* from customers

Where cnum>=1000+(select snum from salespeople where sname='Serres');

Select distinct c.\*

From

Customers c , salespeople s

Where c.cnum>=1000+s.snum

And s.sname='Serres';

22. Give the salespeople’s commissions as percentages instead of decimal numbers.

Select sname,concat(round((comm\*100),2),'%') as comm\_percent

From salespeople;

23. Find the largest order taken by each salesperson on each date, eliminating those MAX orders

which are less than $3000.00 in value.

Select sname,odate,max(amt) max\_amt

From orders o , salespeople s

Where o.snum=s.snum

Group by sname,odate

Having max(amt)>=3000;

24. List the largest orders for October 3, for each salesperson.

Select sname,odate,max(amt) max\_amt

From orders o , salespeople s

Where o.snum=s.snum

And odate='1996-10-03'

Group by sname,odate;

25. Find all customers located in cities where Serres (SNUM 1002) has customers.

Select \* from customers

Where city in (

Select c.city from customers c, salespeople s

Where s.snum=c.snum

And sname='Serres');

26. Select all customers with a rating above 200.00.

Select \* from customers where rating>200;

27. Count the number of salespeople currently listing orders in the Orders table.

Select count(distinct snum) from orders;

28. Write a query that produces all customers serviced by salespeople with a commission above

12%. Output the customer’s name and the salesperson’s rate of commission.

Select cname,sname,comm

From customers c, salespeople s

Where c.snum=s.snum

And comm>.12;

29. Find salespeople who have multiple customers.

Select sname,s.snum,count(distinct cnum)

From customers c , salespeople s

Where c.snum=s.snum

Group by sname,s.snum;

30. Find salespeople with customers located in their city.

Select sname,cname,c.city

From customers c , salespeople s

Where c.snum=s.snum

And s.city=c.city;

#### **Q-5. Write an SQL query to fetch the first 50% records from worker table.**

Select distinct \* from (Select \*,ntile(2) over(order by worker\_id) rn from worker) as t where rn=1;

Delimiter $$

Create function sf\_helloworld(name varchar(100))

Returns varchar(100)

DETERMINISTIC

Begin

return(concat('Hello ',name));

End;

$$

Delimiter ;

Select sf\_helloworld('abc');

Select ename,sf\_helloworld(ename)

From emp limit 10;

Delimiter $$

Create function sf\_closest\_multiple(num1 int,num2 int)

Returns int

DETERMINISTIC

Begin

return(round(num1/num2)\*num2);

End;

$$

Delimiter ;

Drop function sf\_checkprime;

Delimiter $$

Create function sf\_checkprime(num int)

Returns varchar(100)

DETERMINISTIC

Begin

Declare i int;

Set i=2;

while(i<=num/2)

do

If mod(num,i)=0 then

return(concat(num ,' is not a prime number'));

End if;

Set i=i+1;

End while;

return(concat(num ,' is a prime number'));

End;

$$

Delimiter ;

Delimiter $$

Create function sf\_fact(num int)

Returns bigint

DETERMINISTIC

Begin

Declare i int;

Declare fact bigint ;

Set fact=1;

Set i=2;

while(i<=num)

do

Set fact=fact\*i;

Set i=i+1;

End while;

return(fact);

End;

$$

Delimiter ;

Delimiter $$

Create function sf\_cocheckprime(num1 int, num2 int)

Returns varchar(100)

DETERMINISTIC

Begin

Declare i int;

Set i=2;

while(i<=num1 and i<=num2)

do

If mod(num1,i)=0 and mod(num2,i)=0 then

return(concat(num1 ,' and ' ,num2 ,' are not co-prime numbers'));

End if;

Set i=i+1;

End while;

return(concat(num1 ,' and ' ,num2 ,' are co-prime numbers'));

End;

$$

Delimiter ;

Drop function sf\_checkrange;

Delimiter $$

Create function sf\_checkrange(num int)

Returns varchar(100)

Deterministic

Begin

If num>50 then

If mod(num,2)=0 then

return(concat(num ,' is very high even number'));

Else

return(concat(num ,' is very high odd number'));

End if;

else

return(concat(num ,' is low number'));

End if;

end;

$$

Delimiter ;

Delimiter $$

Create function sf\_getdeptname(v\_eid int)

Returns varchar(100)

READS SQL DATA

Begin

Declare v\_dept varchar(100);

Select deptname into v\_dept from emp join dept

On emp.deptid=dept.deptid

Where eid=v\_eid;

return(v\_dept);

End;

$$

Delimiter ;

Delimiter $$

Create function sf\_getelist(v\_dept varchar(100))

Returns mediumtext

Reads sql data

Begin

Declare i int;

Declare cnt int;

Declare elist mediumtext;

Declare v\_ename varchar(100);

Set elist='';

Set i=0;

Select count(\*) into cnt from emp join dept on emp.deptid=dept.deptid

Where deptname=v\_dept;

while(i<cnt)

Do

Select ename into v\_ename from emp join dept on emp.deptid=dept.deptid

Where deptname=v\_dept

Limit 1 offset i;

Set elist=concat(elist,',',v\_ename);

Set i=i+1;

End while;

return(substring(elist,2));

End;

$$

Drop function sf\_getelist2;

Delimiter ;

Drop function sf\_getelist2;

Delimiter $$

Create function sf\_getelist2(v\_dept varchar(100))

Returns mediumtext

Reads sql data

Begin

Declare i int;

Declare elist mediumtext;

Declare v\_ename varchar(100);

Set i=0;

Set elist='';

Set v\_ename='';

while(v\_ename is not null)

Do

Set v\_ename=null;

Select ename into v\_ename from emp join dept on emp.deptid=dept.deptid

Where deptname=v\_dept

Limit 1 offset i;

If v\_ename is not null then

Set elist=concat(elist,',',v\_ename);

End if;

Set i=i+1;

End while;

return(substring(elist,2));

End;

$$

Delimiter ;

Assignment-

1. Redo all the queries and Functions
2. Complete 125 queries
3. Revise Normal forms
4. Revise Isolation Levels
5. Revise Data Types
6. Work in groups to create function to convert numbers to words
7. Write a recursive query to display position of all occurences of a string in a given string
8. Write a function to return position Nth occurence of a string in a given string