**ItJunction4all**

1)with cte

as

(

  select id,emp\_name,salary,dept\_id,

 (select max(salary) from employee e1 where e1.dept\_id=e2.dept\_id) as [max\_salary]

 from employee e2

)

select \*,round(cast(salary as float)/cast(max\_salary as float)\*100,2) as [ratio] from cte

2) with cte

As(

Select \*,row\_number() over (partition by [group] order by sequence asc) as [row\_number] from emp

  ),

 cte2

 as

  (

    select \*,sequence-[row\_number] as [val] from cte

   )

 select [group],max(sequence) as [max\_val],min(sequence) as [min\_val] from cte2 group by [group],[val] order by [group] asc

3) select student\_name,total\_marks,[year],prev\_marks

From

(

Select \*,case when total\_marks>=prev\_marks then 1 else 0 end as [flag] from

(

Select \*,isnull(lag(total\_marks) over (partition by student\_name order by year),100) as [prev\_marks] from student

) d

) k

Where [flag]=1

4) select dept\_id,string\_agg(emp\_name,'  |  ') from employee group by dept\_id

5a) with cte

As

(

Select product\_id,count(\*) as [pro\_count]

From

(

Select distinct order\_day,product\_id from order\_tbl

) s group by product\_id

Having count(\*)=(select count(distinct(order\_day)) from order\_tbl)

)

Select c.product\_id,count(\*) as [quantity] from cte c join order\_tbl o on c.product\_id=o.product\_id group by c.product\_id

5b) select distinct(product\_id) from order\_tbl where order\_day='2015-05-02' and product\_id not in (select distinct(product\_id) from order\_tbl where order\_day='2015-05-01')

6c) with cte

As

(

Select order\_day,product\_id,sales,rank() over (partition by order\_day order by sales desc) as [rank]

From

(

Select order\_day,product\_id,sum(quantity\*price) as sales from order\_tbl

Group by order\_day,product\_id

 ) d

 )

 select \* from cte where [rank]=1

6d) select product\_id,sum(case when order\_day='2015-05-01' then sales else 0 end) as [sales\_2015\_05\_01],

Sum(case when order\_day='2015-05-02' then sales else 0 end) as [sales\_2015\_05\_01]

From

(

Select order\_day,product\_id,sum(quantity\*price) as sales from order\_tbl

Group by order\_day,product\_id

 ) as d

 group by product\_id

6e) select order\_day,product\_id,count(\*) as [count] from order\_tbl

Group by order\_day,product\_id

Having count(\*)>1

7)select \* into temp from order\_tbl2 where 1=2

Declare @o\_id varchar(100),@p\_id varchar(100),@q int

While (select count(\*) from order\_tbl2)>0

Begin

    select top 1 @o\_id=order\_id,@p\_id=product\_id,@q=quantity from order\_tbl2

    while @q>0

    begin

        insert into temp values(@o\_id,@p\_id,1)

        set @q=@q-1

     end

     delete top (1) from order\_tbl2

 end

 select \* from temp

8a) with cte

As

(

Select emp\_name,dept\_id,salary,

(select avg(salary) from employee e1 where e1.dept\_id=e2.dept\_id) as [avg\_salary]

From employee e2

 )

 select \* from cte where salary>avg\_salary order by emp\_name

8b)select emp\_name,e1.dept\_id,salary,avg\_sal from employee e1 join (select dept\_id,avg(salary) as avg\_sal from employee group by dept\_id) e2

On e1.dept\_id=e2.dept\_id and salary>avg\_sal

Order by emp\_name

9) select t1.teamname as team1 ,t2.teamname as team2 from team t1

Cross join team t2 where t1.teamname<>t2.teamname

And t1.id>t2.id

10)select \* from one left join two on one.x=two.y

Select \* from one join two on one.x=two.y

Select \* from one right join two on one.x=two.y

Select \* from one full join two on one.x=two.y

11) with cte

As

(

Select j.team,matches\_played,wins,isnull(draw,0) as draw from

(

Select team,count(\*) as matches\_played  from

(

Select team\_1 as team from match\_result

Union all

Select team\_2 as team from match\_result

) as d

Group by team

) as j

Join

(select result,count(result) as wins from match\_result where result is not null group by result) k

On j.team=k.result

Left join

(

Select team\_1 as team,isnull(count(\*),0) as draw from match\_result where result is null group by team\_1

Union

Select team\_2 as team,isnull(count(\*),0) as draw from match\_result where result is null group by team\_2

 ) b

 on j.team=b.team

  )

  select \*,(matches\_played-wins-draw) as loss from cte

12)with cte

As

(

Select \*,row\_number() over(partition by accountnumber order by transactiontime desc) as r\_n

From transaction\_table

 )

 select \* from cte where r\_n=1

 order by transactionid

13) select 'total sales' as total\_sales,

Sum(case when salesyear=1998 then quantitysold else 0 end) as [1998],

Sum(case when salesyear=1999 then quantitysold else 0 end) as [1999],

Sum(case when salesyear=2000 then quantitysold else 0 end) as [2000]

From salestbl

14) select \*,sum(quantity) over (partition by prodname order by inventorydate) as running\_total from inventory

15) create table alphabets

(

  id int,

  val varchar(10)

 )

Declare @val as int=65

Declare @id as int=1

While (@val<=90)

Begin

    insert into alphabets values(@id,char(@val))

    set @id=@id+1

    set @val=@val+1

  end

  select \* from alphabets

18) with cte

As

(

Select \*,

Case when trantype='credit' then amount else -amount end as net

From account\_table

 )

 select \*,sum(net) over (order by trandate) as r\_n from cte

19) select studentname,subject,marks from studentinfo

Unpivot

(

    marks for subject in ([english],[maths],[science])

 )

 as un\_pivot

20) with cte

As

(

  select \*,row\_number() over (order by trade\_id) as r\_n from trade\_tbl

),

Cte2

As

(

  select t1.trade\_id as t1\_trade, t2.trade\_id as t2\_trade,

  abs(round(cast((t1.price-t2.price) as float)/cast(t2.price as float)\*100,1)) as percent\_diff

 ,abs(datediff(second,t1.trade\_timestamp,t2.trade\_timestamp))

 as date\_diff from cte t1 cross join cte t2 where t1.r\_n>t2.r\_n

)

Select \* from cte2 where date\_diff<10 and date\_diff<>0 and percent\_diff>10

24) select \* from sample\_table

Declare @min as int

Declare @max as int

Select @min=min(id),@max=max(id) from sample\_table

While(@min<=@max)

Begin

    if (select count(\*) from sample\_table where id=@min)=0

    begin

        insert into sample\_table values(@min)

    end

    set @min=@min+1

 end

 select \* from sample\_table order by id

25) with cte

As

(

    select  \*,rank() over (partition by dept\_id order by salary desc) as [rank] from employee

  )

  select id,emp\_name,gender,salary,dept\_id from cte

  where [rank]<=3

26) with cte

As

(

    select \*,row\_number() over (partition by source\_phone\_nbr order by call\_start\_datetime) as r\_n  from phone\_log

 )

 select distinct source\_phone\_nbr,case when [count]=2 then 'y' else 'n' end as [flag] from

 (

 select source\_phone\_nbr,destination\_phone\_nbr,count(\*) as [count] from

 (

 select p.source\_phone\_nbr,p.destination\_phone\_nbr from cte p join

 (select source\_phone\_nbr,min(r\_n) as min\_val from cte group by source\_phone\_nbr) as j

 on p.source\_phone\_nbr=j.source\_phone\_nbr and p.r\_n=j.min\_val

 union all

 select p.source\_phone\_nbr,p.destination\_phone\_nbr from cte p join

 (select source\_phone\_nbr,max(r\_n) as max\_val from cte group by source\_phone\_nbr) as j

 on p.source\_phone\_nbr=j.source\_phone\_nbr and p.r\_n=j.max\_val

  ) as u group by source\_phone\_nbr,destination\_phone\_nbr

 ) k

27) create table range

(

  id int

)

Select \* from sampletable

Declare cur cursor for select \* from sampletable

Declare @start as int, @end as int

Open cur

Fetch next from cur into @start,@end

While (@@fetch\_status=0)

Begin

    while (@start<=@end)

    begin

        insert into range values (@start)

        set @start=@start+1

    end

    fetch next from cur into @start,@end

End

Close cur

Select \* from range

28) select \* from sample\_1 where x in

(select x from sample\_1 where z>1)

And y in (select y from sample\_1 where z>1)

29a) select e1.employeename as employee, e2.employeename as manger

From employee\_1 e1 join employee\_1 e2

On e1.managerid=e2.employeeid where e1.managerid is not null

Order by e1.employeeid

29b) select e1.employeename as employee, isnull(e2.employeename,'boss') as manger

From employee\_1 e1 left join employee\_1 e2

On e1.managerid=e2.employeeid

Order by e1.employeeid

30a) select id,s1.product,min\_sales from sales1 s1 join

(select product,min(sales) as min\_sales from sales1 group by product) s2

On s1.product=s2.product

30b) with cte

As

(

  select \*,sum(sales) over (partition by product order by id) as [running\_total]

  from  sales1

  )

  select id,product,running\_total as sales from cte

31) select studentname,english,maths,science from studentinfo\_1

Pivot

(

    sum(marks) for subjects in ([english],[maths],[science])

 ) as [pivot\_table]

32) with cte

As

(

Select \*,rank() over (partition by dept\_id order by salary) as low\_salary,

Rank() over (partition by dept\_id order by salary desc) as high\_salary

From employee

 )

 select \* from cte where low\_salary=1 or high\_salary=1

 order by dept\_id

33) create table [pattren] (val varchar(100))

Declare @name as varchar(10)='babavali'

Declare @len as int

Select @len=len(@name)

While(@len>0)

Begin

    declare @val as varchar(100)=left(@name,@len)

    insert into [pattren] values (@val)

    set @len=@len-1

End

Select \* from [pattren]

34) with cte

As

(

Select quote\_id,count(\*) as total, count(case when order\_status='delivered' then 1 else null end ) as [delivered],

 count(case when order\_status='submitted' then 1 else null end ) as [submitted],

 count(case when order\_status='created' then 1 else null end ) as [created]

 from orderstatus group by quote\_id

  )

  select quote\_id,case when total=delivered then 'completed'

                        when total/2=delivered then 'in progress'

                        when total=submitted then 'yest to start'

                        when total=created+submitted then 'submission in progress'

                        when total=created then 'waiting for submission'

                        end as [status]

  from cte

35A) select \*, case when upper(First\_name)=first\_name COLLATE latin1\_general\_cs\_as then 1 else 0 end as [upper] from employees

35B) select Employee\_no,birth\_date,

substring(first\_name,1,iif(charindex(' ',first\_name)=0,len(first\_name),charindex(' ',first\_name))) as [first\_name],

iif(len(last\_name)=0,substring(first\_name,charindex(' ',first\_name),len(first\_name)),last\_name) as [last\_name],joining\_date

from employees

update employees

set last\_name=substring(first\_name,charindex(' ',first\_name),len(first\_name)),

first\_name=substring(first\_name,1,charindex(' ',first\_name))

where len(last\_name)=0

35C) with cte

AS

(

select \*,datediff(year,joining\_date,'2017-06-30') as exp\_years from employees

)

select exp,count(\*) as No\_of\_employees FROM

(

select \*,case when exp\_years<1 then 'less than 1 year'

              when exp\_years>=1 and exp\_years<=3 then 'between 1 and 3'

              when exp\_years>3 and exp\_years<=5 then  'between 3 and 5'

              else '5+ years' end exp

 from cte

  ) as d group by exp order by No\_of\_employees desc

35D) select \* from employees where month(birth\_date)=month(joining\_date)

and day(birth\_date)=day(joining\_date)

35E) with cte

AS

(

select \*,datediff(year,joining\_date,'2017-06-30') as exp,

datediff(year,birth\_date,'2017-06-30') as age from Employees

)

select \* from cte where exp>=5

and age=(select min(age) from cte where exp>=5)

36) with cte

AS

(

select \*,id%2 as val from SeatArrangement

)

select id,studentName,

isnull(case when val=1 then lead(studentName,1) over(order by id asc)

    else lag(studentName,1) over (order by id asc) end,studentName)

  as New\_Student\_Name from cte

37) with cte

AS

(

select \*,rank() over (partition by continents order by sales desc) as [rank] from salesinfo

)

SELECT continents, country, sales from cte where [rank]=1

39) create table theater\_seats

(

  id int,

  row\_name varchar(10),

  seats varchar(500)

  )

  create table ts (id int,row varchar(10),seats varchar(10))

  declare @val as int=65

  declare @count as int=1

  while (@val<=90)

  BEGIN

    declare @val2 as int=1

    while (@val2<=10)

    BEGIN

        insert into ts values(@count,char(@val),char(@val)+cast(@val2 as varchar(10)))

        set @val2=@val2+1

    END

    set @val=@val+1

    set @count=@count+1

 END

  insert into theater\_seats

  select id,row,string\_agg(seats,' | ') from ts group by id,row

  select \* from theater\_seats

  40) with cte

AS

(

  select distinct serialno,name from Emp\_Table

)

select k.serialno,k.name,k.month\_id,k.name,isnull(e.amount,0) as amount

from Emp\_Table e right join

(select \* from cte cross join Month\_Table) k

on e.month\_id=k.month\_id and k.name=e.name

order by k.serialno

41) with cte

AS

(

select club\_id,string\_agg(edu,':') as st\_ag from club group by club\_id

)

select club\_id,isnull(sum(points),0) as rewards FROM

(

select club\_id,value,case when value='mm' then 0.5

                    when value='ci' then 0.5

                    when value='co' then 0.5

                    when value in ('cd','cl','cm') then 1 end as points

 from cte c1 outer apply string\_split(c1.st\_ag,':')

 ) b group by club\_id

42) create table item\_dup( id int,name varchar(100),cust\_id int,quantity int)

declare cur cursor for select \* from Item

declare @name as varchar(10),@val as INT

open cur

fetch next from cur into @name,@val

declare @id1 as int=1

While(@@fetch\_status=0)

BEGIN

    declare @id as int=1

    While(@val>0)

    BEGIN

        insert into item\_dup values(@id1,@name,@id,@val)

        set @id=@id+1

        set @id1=@id1+1

        set @val=@val-1

     END

    fetch next from cur into @name,@val

END

close cur

select \* from item\_dup

43) with cte

AS

(

select app from Customer c join Transaction\_Tbls t on c.Customer\_id=t.Cus\_id

join

(select loc\_name,max(amount\_paid) as high\_amount from Transaction\_Tbls group by loc\_name) d

on d.loc\_name=t.Loc\_name and t.Amount\_paid=d.high\_amount

)

select upper(substring(app,1,1))+lower(substring(app,2,len(app))) as App, no\_of\_times

from

 (

   select isnull(app,'offline') as app,count(\*) as no\_of\_times from cte group by app

 ) as L

 order by no\_of\_times desc

44) with cte

As

(

Select userid,createdat from transactions\_amazon

),cte2 as

(

    select count(\*)/7 as [val] from transactions\_amazon

 )

Select distinct userid from

(

Select \*,abs(datediff(day,createdat,lag(createdat,1) over (partition by userid order by createdat)))

As diff from cte

) as f where diff<=7

45)With cte as

(

Select k.studentid,isnull([1],0) as subject1,

Isnull([2],0) as subject2

,isnull([3],0) as subject3

,isnull([4],0) as subject4,isnull(g.total,0) as total

From

(

Select studentid,[1],[2],[3],[4]

From exam\_score

Pivot

(

Sum(marks) for subjectid in

([1],[2],[3],[4])

) as pivot\_table

) as k

Join

(

Select studentid,sum(marks) as total

From exam\_score

Group by studentid

) as g

On g.studentid=k.studentid

)

Select \*,

Case when subject1&gt;=40 and subject2&gt;=40

And subject3&gt;=40 and subject4&gt;=40 and

Total&gt;=200 then ‘PASS’ else ‘FAIL’

End as result from cte

46)With cte as

(

Select employeeid,employeename,

Isnull(managerid,0) as managerid

,1 as level

From employee\_table

Where managerid is null

Union all

Select e.employeeid, e.employeename,

e.managerid,c.level+1 as level from

employee\_table e

join cte c

on e .managerid=

c.employeeid

)

Select \* from cte

Order by level

47)With cte as

(

Select \*,

Row\_Number() Over (Partition by id order by marks desc)

As rn from

(

Select id,subjects,marks from ssc\_exam

Unpivot

(

Marks for subjects in ([English],[Maths],

[Science],[Geography],

[History],[Sanskrit])

)

As unpivot\_table

) as G

)

Select s.\*,c.Total\_percent\_5 from

(

Select id,cast(sum(marks) as float)/cast(5 as float)

As [total\_percent\_5] from cte where rn&lt;6 group by id) as c

Join ssc\_exam s on s.id=c.id

49)With cte as

(

Select \*,

Case when from\_id>=to\_id then from\_id else to\_id end as p1,

Case when from\_id<to\_id then from\_id else to\_id end as p2

From calls

)

Select p1,p2, count(\*) as [count],

Sum(duration) as duration from cte

Group by p1,p2

Order by duration DESC

50)With cte as

(

Select s.id,s.student\_name,f.friend\_id,p.salary

From students\_tbl s join friends\_tbl f

On s.id=f.id join package\_tbl p on p.id=s.id

)

Select id,student\_name from

(

Select c.\*,p.salary as friend\_salary from cte c join package\_tbl p

On c.friend\_id=p.id

) as F where salary<=friend\_salary

52)with cte as

(

Select a1,b1, row\_number() over

(partition by a1,b1 order by a1) as rn

From

(select

Case when a>b then b else a end a1,

Case when a<b then b else a end b1

From reverse\_duplicates) as d

)

Delete from cte where rn>1

Select \* from reverse\_duplicates

53) with cte

AS

(

select from\_user,count(\*) as coun from google\_gmail\_emails

group by from\_user

)

select \*,row\_number() over (order by coun desc) as [rank]

from cte

54) with cte

AS

(

select t.\*,p.product\_name from Transactions\_Walmart t join Products\_Walmart p

on t.product\_id=p.product\_id

) ,cte2 AS

(

select c1.users\_id,c1.transaction\_id,c1.product\_name as p1, c2.product\_name as p2

from cte c1 join cte c2

on c1.users\_id=c2.users\_id and c1.transaction\_id=c2.transaction\_id

and c1.product\_id<>c2.product\_id

)

select top(3) \* from

(

select product\_1,product\_2,count(\*)/2 as [coun] from

(

select users\_id,transaction\_id,case when p1>p2 then p1 else p2 end as product\_1,

                               case when p1>p2 then p2 else p1 end as product\_2

from cte2

) as D

group by product\_1,product\_2

) as L

order by coun desc

55) with cte

AS

(

select users\_id,month(event\_date) as [month] from user\_actions

group by users\_id,month(event\_date)

)

select month,count(\*) as [active\_users] from

(

select \*,month-lag(month,1) over (partition by users\_id order by month) as diff from cte

) as d

where month=7 and diff=1

group by month

56) with cte

AS

(

select \* from

(

select \*,datediff(day,start\_dates,end\_dates)+1 as diff,year(end\_dates) as y1,year(start\_dates) as y2

from

(

select employee\_id,start\_dates,end\_dates from Staffing s join

Consulting\_engagements c on s.job\_id=c.job\_id where is\_consultant=1

) as d

) as e

where y1=2021 and y2=2021

)

select employee\_id,365-working\_days as bench\_days from

(

select employee\_id,sum(diff) as working\_days from cte

group by employee\_id

) as d

order by bench\_days

note: - datediff(day,2022-01-01’,2022-01-30) returns 29 not 30 so do +1

57) with cte AS

(

select \*,case when amount>lag then 1 else 0 end as inc

from

(

select \*,lag(amount,1) over (partition by brand order by years asc) as lag from Brands

) as e

 )

 --select \* from cte

 select brand from

 (

 select \*,(select count(\*) from brands b where b.brand=h.brand)-1 as val

 from

(

 select brand,count(\*)as incs from cte

where inc=1

group by brand

) as h

) as m

where incs=val

 Note:- we can achieve distinct by using group by with out aggregation.

58) with cte

AS

(

    select \*,dense\_rank() over (partition by users\_id order by transaction\_date desc) as rn

    from user\_transactions

  )

  select transaction\_date,count(\*)  as no\_of\_users,

  count(distinct(users\_id)) as number\_of\_product from cte where rn=1

  group by transaction\_date

59) create table temp( val int)

declare cur cursor for select \* from search\_frequency

declare @search as int,@users as INT

open cur

fetch next from cur into @users,@search

while (@@fetch\_status=0)

BEGIN

    while(@search>0)

    BEGIN

        insert into temp  values (@users)

        set @search=@search-1

    end

    fetch next from cur into @users,@search

 end

 close cur

 declare @g as INT,@k as int

 select @g=count(\*)/2 from temp

 select @k=count(\*)%2 from temp

 if @k=0

BEGIN

    select round(cast(sum(val) as float)/cast(2 as float),1) as median from

    (

    select \*,row\_number() over (order by val) as rn from temp

    ) as d where rn in (@g,@g+1)

 end

 else

 BEGIN

    select count(\*) from temp

    select sum(val) as median from

    (

    select \*,row\_number() over (order by val) as rn from temp

    ) as d where rn = @g

  end

60) with cte

AS

(

select \*,count(price) over (partition by stocksname order by datekey) as r

from stocks

)

select \*,first\_value(price) over ( partition by stocksname,r order by datekey asc) as [derrived\_price]

from cte

61) with cte

AS

(

select \* from (

select \*,rank() over (partition by device\_id order by max\_loc desc) as r from

(

select device\_id,locations,count(\*) as max\_loc from Device

group by device\_id,locations

) as e

) as d where r=1

)

select a.device\_id,max\_loc,no\_of\_loc,signals from

(select device\_id,count(\*) as signals from device group by device\_id) a

join (select device\_id,max\_loc from cte) b on a.device\_id=b.device\_id

join (select device\_id,count(locations) as no\_of\_loc from device group by device\_id)  C

on c.device\_id=a.device\_id

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Kudvenkat**

1) Nth highest salary

with cte

AS

(

select \*,dense\_rank() over (order by salary desc) as [rank] from EMPLOYEE

)

select \* from cte where rank=3

2)employee hierarchy by recursive cte

 with cte

AS

(

select employeeid,employeename,cast('boss' as varchar(100)) as manager\_name from Employees where managerid is null

union all

select  e.EmployeeID,e.EmployeeName,cast(c.employeename as varchar(100)) as manager\_name

from employees e join cte c on e.ManagerID=c.employeeid

)

select \* from cte

3)employee hierarchy by self join

select e1.employeeid,e1.employeename as emp\_name,isnull(e2.employeename,'boss') as manager\_name

from employees e1

left join employees e2

on e1.managerid=e2.EmployeeID

4) remove duplicates

 with cte AS

(

  select \*,row\_number() over (partition by id,firstname,lastname,gender,salary order by id)

  as rn from employees\_new

)

delete from cte where rn>1

select \* from employees\_new

5)employees hired in last 20 months

select e.\*,j.JOIN\_DATE from employee e join JOIN\_DATE j on e.id=j.id

where datediff(month,JOIN\_DATE,getdate())=20

6)Shaping table

select country,[city1],[city2],[city3] from

(

select \*,'city'+cast(row\_number() over (partition by country order by country) as varchar(10)) as cities from Countries

) as g

pivot

(

  max(city) for cities in ([city1],[city2],[city3])

) as pivot\_table

7)isnumeric

select \* from TestTable where isnumeric(value)=1

retuns 1 if value is numeric retuns 0 if value is not numeric

8)highest employees in department

with cte AS

(

select \*,rank() over (order by no\_of\_employees desc) as rank

from (

select dept\_id,count(\*) as [no\_of\_employees] from EMPLOYEE

group by dept\_id

    ) as g

)

select dept\_id,no\_of\_employees from cte where rank=1

9)Difference between inner join and left.

Inner join returns matching row between two tables like a intersection b.

Left join returns matching rows from right table and all rows from left table [matching and non matching].

10)joining multiple tables

Select e.\*,d.dept\_name,d.dept\_head\_id,l.location\_name,

L.location\_head\_id,j.join\_date from

Employee e join department d on e.dept\_id=d.dept\_id

Join location l on e.location\_id=l.location\_id

Join join\_date j on e.id=j.id

11)right join, cross join, and self-join

Right join returns matching rows from left table and all matching and non-matching rows from right table.

Cross join it used to join two tables in all possible ways. It does not have matching criteria [no on]. If table a has 10 rows and table b has 5 rows we get Cartesian product of both tables 10\*5 50.

Self-join when we join a table with it own table is nothing but a self join. Mostly used in employee hierarchy.

12)Join can possible without relation between them.

Yes we can perform joins between tables with out relationships. Columns involving in matching criteria in join must have same or convertible datatypes.

24) can we replace right join with left join.

Yes we can replace right join with left join by changing positions of table with join type.

23)most repeated value in a column

select top 1 location\_id, count(\*) as [no\_of\_times] from EMPLOYEE

group by location\_id

order by no\_of\_times

22)adding primary key on multiple columns.

Primary key is formed by more than one column it is nothing but composite key.

create table pk\_com

(

  subject\_id int,

  student\_id int ,

  constraint pk primary key(subject\_id,student\_id)

 )

 alter table pk\_com add constraint pk2 PRIMARY key(subject\_id,student\_id)

21)usingstring\_split()

String\_split() used to break the given string into values in a column [value] based on delimiter we specified.

String\_split(string,delimeiter)

declare @name as varchar(100)='john,mark,steve,tom'

select \* from employees where employeename in (select \* from string\_split(@name,','))

20)Extract numbers from text form name and id columns.

create table new( id int, name varchar(100))

declare @len as int

select @len=count(\*) from testtable

while(@len>0)

BEGIN

    declare @val as varchar(100)

    select top 1 @val=idname from TestTable

    declare @id as varchar(10)=''

    declare @name as varchar(100)=''

    declare @coun as int=1

    while len(@val)>=@coun

    BEGIN

        declare @char as char(1)=substring(@val,@coun,@coun)

        if @char like '[0-9]'

        BEGIN

            set @id=@id+@char

        end

        else

        BEGIN

            set @name=@name+@char

         end

         set @coun=@coun+1

      end

      insert into new values(cast(@id as int),@name)

      delete top (1) from TestTable

      set @len=@len-1

  end

  select \* from new

  drop table new

  select \* into temp from TestTable

  select \* from temp

19)RDBMS, SQL, T-SQL, PL SQL

RDBMS means relational database management system in which data is stored in tables and relation established between them using primary and foreign key relationships.

SQL is structured

SQL is structured query language which used to interface with relational databases. It is developed by ANSI.

T-SQL means transact SQL which is developed by Microsoft for database MS SQL Server.

PL SQL means procedural language SQL which is developed by oracle for oracle database.

Oracle is first company which developed SQL later ANSI make it standard.

PL SQL and T SQL has almost ANSI or Standard SQL with some more functionality developed by Microsoft for T-SQL and  Oracle for PL-SQL

17)Every join returns same number of rows

create table a( id int)

insert into a values(1),(1),(1)

create table b(id int)

insert into b values(1),(1)

select count(\*) as [count] from a join b on a.id=b.id

select count(\*) as [count] from a left join b on a.id=b.id

select count(\*) as [count] from a right join b on a.id=b.id

select count(\*) as [count] from a cross join b

16)date interview questions

a)join in particular date

select e.\*,join\_date from

EMPLOYEE e join JOIN\_DATE j on e.ID=j.ID

where join\_date='2016-08-19'

b)between two dates

select e.\*,join\_date from

EMPLOYEE e join JOIN\_DATE j on e.ID=j.ID

where join\_date between '2016-08-19'  and '2022-02-02'

c)join on same year

select e.\*,join\_date from

EMPLOYEE e join JOIN\_DATE j on e.ID=j.ID

where year(join\_date)=2019

d)join on same day and month

select e.\*,join\_date from

EMPLOYEE e join JOIN\_DATE j on e.ID=j.ID

where day(join\_date)=29 and month(join\_date)=3

14)find name starts with e with out like operator and end with 5.

select \* from EMPLOYEE

where left(emp\_name,1)='E' and right(emp\_name,1)='5'

            or

select \* from employee

where substring(emp\_name,1,1)='E' and substring(emp\_name,len(emp\_name),len(emp\_name))='5'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Learn SSIS**

1)what is SQL

SQL means structured query language which used to interact with relational database. By using SQL we can access, modify data in relational database.

2)What is SQL Server

SQL server is the relational database which is developed my Microsoft by using T-SQL [transact] we access the SQL Server database.

3)What is Database

Database it the organized collection of data which can easily accessed and manipulated. There are different databases like relational database, NO SQL [document] database etc.

4)what is Table

Table consists of rows and columns which are used to stored data. Relational database stores data in table only.

5)What is Views

Views are virtual tables which does not store data they only stores the query used to fetch the data. We can use it as abstraction to database object.

6)What is normalization

Normalization is a database design technique which used to eliminate redundance of data by splitting data in to different tables.

7)types of normalization

First normal form 1nf

Second normal form 2nf

third normal form 3nf

Boyce Codd normal form bcnf

fourth normal form 4nf

fifth normal form 5nf

8)what is denormalization

It merges data from multiple tables to single large table to speed up the query it is quite opposite to normalization.

9)Difference between DDL and DML

DDL used to define the structure of table it consists of create, drop, rename, truncate, alter. Where clause cannot be used.

DML used to manipulate the data in the table it consists of update, insert, delete, select. We can use where clause.

10)default constraint

Default constraint assign the default value mentioned it value is not specified to column or null assigned to column.

11)what is Joins and types.

Joins are used to join two tables based on matched columns between two tables.

Inner join gets matching rows from two tables.

Left join all rows from left and matching from right table.

Right join all rows from right table and matching from left table.

Full join all rows from both tables.

Self join table join with same table.

Cross join cartesian product of two tables.

12)what is primary key.

Primary key is  a column or combination of columns which is used to uniquely identify the row in the table.

It  must be unique and not null.

13)what is unique key or constraint

Unique key or constraint ensures that the values in the particular column on which unique key is created must be unique no duplicates are allowed only one null value is accepted even two nulls are different.

14)difference between primary key and unique key

Primary key does not allow nulls where unique key allow on null. One primary key is allowed per table but multiple unique key can have in a table.

15)what is foreign key

Foreign key is a column or combination columns which is primary key in other tables. By suing this primary key and foreign key we establish relation between table.

16)ACID properties

ACID properties are the properties which are possessed by transaction

A-atomicity-all commands in a transaction must executes successfully if any one command fails remanning commands also need to be rollback.

C-consistency-data w=touched by transaction must be consistent before and after the transaction.

I-isolated-each transaction must isolated from other.

D-durability- one the change has made by transaction that should be permanent.

17)difference between delete and truncate.

Delete used to delete the records in the table we can specify where clause. It is DML command. Identity value cannot reset to starting value. DML can be rolled back.

Truncate used to remove records from table we cannot use where clause it removes all records in the table. It is DDL command. It resets the identity value to starting value. Truncate cannot be rolled back.

18)difference between drop and truncate

Both are DDL commands but drop used to remove table where truncate removes data in table and make it empty.

19)what is triggers

Triggers are special type of stored procedures which are pre compiled and stored in database and they trigger which correspond event in database like, insert, update, delete, create, rename etc.

20)error handling

We can handle error in SQL Server by using try and catch block we write our code in try block if any error occurs it moves to catch block and we can store that error. And we can use @@error and riseerror for handling error but it not must efficient and try catch block.

21)what is functions

Functions are block of code and used to do a specific task. There are two types of functions user defined and system functions. Which reduces line of code increases reusability.

22)types of user defined functions

 User defined functions are functions which are defined by user they are further divided in to 3 types

scalar user defined functions which return as single value.

Inline table valued functions which return a table and

multi statement table valued function which return as table which is defined by us.

23)what is stored procedure.

Stored procedure is a block of code which is pre compiled and stored in database and we can use it n number of times based on our need it reduces line of code and reduces repeated writing of code.

24)difference between stored procedure and function

Function need parameters must return a value, we can use it in select statement, we cannot write a procedure in function.

Stored procedure we can create as SP without parameters, not mandatory to return a value, we cannot use it in select statement. we can write a function in stored procedure.

25)what is coalesce function.

Coalesce function used to replace nulls.

It return first not null value specified in the list of values in it. We can specify n number of values in it.

If all specified values are null it returns error.

27)how sort in SQL Server.

By using order clause, we can sort data in SQL Server. By default it sorts ascending order we can specify as ASC to sort in descending order use DESC. We can use multiple columns in order by clause or its column positions 1,2 etc. in order by clause.

28)How to filter data in SQL Server

By using where and having clause.

Where clause used to filter data before data is aggregated where having used to filter data after aggregated.

29)wild card characters

We wild card characters in like.

-   Represent single character.

%  Represents any number of characters.

Range of characters [a-z0-9]

Multiple characters [a,c,v,f]

^ as not

30)Temp tables.

Temp table as temporary tables which stored in tempdb which is system database for certain period of time.

There are two types of temp tables

Local temp tables and global temp tables.

31)difference between local and global temporary table.

Local temporary table starts with single # at prefix global temporary table has double # at prefix.

Local temporary tables are accessible with in the session it created. Where global temporary tables are accessible across the session until the session in which it created is not closed.

32)what is cursor

Cursors are special datatypes which used to iterate the table row by row.

But cursors are very slow.

When we need to update value row by row we uses cursor. Example extract id, name from single column.

33)what is window functions

Window functions aggregate, ranking, analytical functions over a set or rows or at specified window.

To specify the set of row we use rows and range.

Cum\_dist, percent\_rank, ntile etc.

34)what is transaction

Transaction is a block or single unit which groups multiple SQL commands, either all commands executes, or no command will executes.

35)what are indexes in transaction.

Index are used to fast up the query. Index can create on single or multiple columns.

Primary key = unique clustered index

There are two types clustered index and non-clustered index. We can use unique property for both clusters.

Clustered index defines the physical order of table and it stored with in the table. Only one per table.

Non clustered index used to speed up the query and does not define physical order of table and can have multiple in the table. And are stored in separate memory so create non clustered index based on requirement.

36)what is clustered index.

Clustered index defines the physical order of table and it stored with in the table.

Primary key = unique clustered index. One clustered index allowed per table.

Leaf of pages of clustered index consist of data.

Table does not have clustered index called heap table. Table have clustered index called clustered table.

37)what is Non clustered index.

Non clustered index used to speed up the query and does not define physical order of table and can have multiple in the table. And are stored in separate memory so create non clustered index based on requirement.

Leaf page of non clustered index consist of pointers which refers the data in the table.

When we are loading data into table first drop all non-clustered index which speed ups query after create index.

Can have 999 non clustered index per table.

38)difference between clustered and non-clustered index.

Clustered index Non clustered index

One per table   999 per table [multiple]

Defines physical order of table It does not define physical order of table

Stored in the table and fast    Slow stored separately.

39)difference between table and view.

Table consist of data, views are virtual tables does consist of data and views can be created on multiple tables.

Tables are independent of views where views are dependent on table.

40)what is self join.

Self join means joining a table with it. Mainly used to find employee hierarchy.

41)Difference between OLTP and OLAP

OLTP    OLAP

Online transaction processing   Online analytic processing

It used to store day to day data    It used to historical data.

DML happens on OLTP. Small size tables  Select used on OLAP. Large size tables

42)difference between Union and Union all

Union and Union all returns union of records from specified tables where union does not returns duplicates means unique records where union all returns duplicate records.

43)Copy data from one table to other.

If table is already created we use insert into [new\_table] select column1,column2,… from [old\_table]

If table is not created we use select column1,column2,.. into [new\_table] from [old\_table].

44)remove duplicate records.

Using cte, row\_number, where clause row\_number>1

with cte AS

(

  select \*,row\_number() over (partition by id,firstname,lastname,gender,salary order by id)

  as rn from employees\_new

)

delete from cte where rn>1

select \* from employees\_new

45)Nth highest salary

Using cte, dense\_rank(), where dense\_rank=Nth

with cte

AS

(

select \*,dense\_rank() over (order by salary desc) as [rank] from EMPLOYEE

)

select \* from cte where rank=3

46)How to create new table with same structure of old table.

Select \* into [new\_table] from [old\_table] where 1=2

Here where is always false so no data load into new table but both contains same structure.

47)what is DMVS

DVMS means Dynamic Management view/Functions used to know the information of server state which helps to monitor the health of server instance and diagnosis problems and tune performance.

48)list out DVMS

Select \* from sys.system\_objects where name like ‘dv%’

If type is view it is dynamic management view if type is sql inline table valued function then it is dynamic management function.

50)Count duplicate rows

with cte as

(

  select \*,row\_number() over (partition by id order by id) as rn from emp\_dup

)

 select count(\*) as [duplicate\_rows] from cte where rn>1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LeetCode Top 50 for interview Prep**

1)select product\_id

from products

where low\_fats='y' and recyclable='y'

2)select name

from customer

where referee\_id<>2 or referee\_id is null

3)select name,population,area

from world

where area>=3000000 or population>=25000000

4)select distinct(author\_id) as id

from views

where author\_id=viewer\_id

5)with cte

as

(

select tweet\_id,content,len(content) as [len\_tweet]

from tweets

)

select tweet\_id from cte where [len\_tweet]>15

6)select EU.unique\_id,E.name

from Employees E full join EmployeeUNI EU

on E.id=EU.id where E.name is not null

7)select  product\_name,year,price

from sales s join product p

on s.product\_id=p.product\_id

8a)select customer\_id,count(\*) as [count\_no\_trans]

from visits v full join transactions t

on v.visit\_id=t.visit\_id where t.visit\_id is null

group by customer\_id order by [count\_no\_trans] desc

8b)select customer\_id,count(\*) as [count\_no\_trans]

from visits

where visit\_id not in (select visit\_id from transactions)

group by customer\_id

order by [count\_no\_trans]

9)with cte

as

(

select id,recordDate,temperature,

lag(temperature,1,temperature) over (order by recordDate asc) as [prev],

lag(recordDate,1,recordDate) over (order by recordDate asc) as diff

from weather

)

select id from cte where prev<temperature and

datediff(day,diff,recordDate) in (1,-1)

10)with cte

as

(

select machine\_id,[start],[end] from activity

pivot

(

    sum([timestamp]) for activity\_type in ([start],[end])

) as pivot\_table

)

select machine\_id,

round(sum(round([end]-[start],3))/

(select count(\*)/2 as no\_of\_activities

from activity a

where a.machine\_id=c.machine\_id

group by a.machine\_id),3)  as [processing\_time]

from cte c

group by machine\_id

11)select name,bonus from

employee e left join bonus b

on b.empid=e.empid

where bonus<1000 or bonus is null

12)select a.student\_id,a.student\_name,a.subject\_name,count(b.subject\_name) as [attended\_exams] from

(select student\_id,student\_name,subject\_name

from students s

cross join subjects su) a

left join

(

select s.student\_id,s.student\_name,e.subject\_name

from students s join examinations e on s.student\_id=e.student\_id

)b

on a.student\_id=b.student\_id and a.subject\_name=b.subject\_name

group by a.student\_id,a.student\_name,a.subject\_name

order by a.student\_id,a.subject\_name

13)with cte as

(

Select managerId,count(managerId) as [no\_of\_reporties]

from Employee

where managerId is not null

group by managerId

)

select name from employee where Id in

( select managerId from cte where [no\_of\_reporties]>=5)

14)with cte

as

(

select s.user\_id,isnull([action] ,'timeout') as [action]

from signups s left join confirmations c

on s.user\_id=c.user\_id

)

select user\_id,

round(cast([confirmed] as float)/cast([timeout]+confirmed as float),2) as

confirmation\_rate from

(

select user\_id,[timeout],[confirmed] from cte

pivot

(

    count([action]) for [action] in([timeout],[confirmed])

) as pivot\_table

) as d

15)select id,movie,description,rating

from cinema where id%2=1 and description<>'boring'

order by rating desc

16) with cte as

(

select p.product\_id,units,price from

prices p left join unitssold u

on p.product\_id=u.product\_id

where purchase\_date between [start\_date] and end\_date or purchase\_date is null

)

select product\_id,round(cast(sum(isnull(units,0)\*price) as float)/cast(isnull(sum(units),1) as float),2) as average\_price from cte

group by product\_id

Note: - athematic operation of a value and null returns null

17)select project\_id,

round(cast(sum(experience\_years) as float)/cast(count(experience\_years\*1.0) as float),2)

 as average\_years

from project p join employee e

on p.employee\_id=e.employee\_id

group by project\_id

order by project\_id

18)select contest\_id,round(((cast(count(\*) as float)/

(select count(\*) as total from users))\*100),2) as [percentage]

from users u join register r

on u.user\_id=r.user\_id

group by contest\_id

order by [percentage] desc, contest\_id asc

note:- in athematic operation if one value is float result will be float no need both to be float

19)select query\_name,

round((sum(cast(rating as float)/position))/count(\*),2) as [quality],

round(cast(count(case when rating<3 then 1 else null end) as float)/count(\*),4)\*100 as poor\_query\_percentage

from queries

group by query\_name

order by quality

20)with cte as

(

select \*,substring(cast(trans\_date as varchar(100)),1,7) as [year\_month]

from transactions

)

select [year\_month] as [month],country,

count(\*) as [trans\_count],

count(case when [state]='approved' then amount else null end) as approved\_count,sum(amount) as [trans\_total\_amount],

sum(case when [state]='approved' then amount else 0 end) as approved\_total\_amount

from cte

group  by [year\_month],[country]

order by [month], [country] desc

21) with cte

as

(

select delivery\_id,customer\_id,order\_date,customer\_pref\_delivery\_date,

row\_number() over (partition by customer\_id order by order\_date asc) as rn

from delivery

)

select round(cast(count(case when order\_date=customer\_pref\_delivery\_date then 1 else null end) as float)/count(\*),2) \*100 as [immediate\_percentage] from cte where rn=1

22)with cte

as

(

select \* from

(

select \*,lag(event\_date,1,event\_date) over (partition by player\_id order by event\_date asc) as prev\_date,row\_number() over (partition by player\_id order by event\_date asc ) as rn from activity

) as d where rn in (1,2)

)

select round(cast(count(distinct(player\_id)) as float)/

cast((select count(distinct(player\_id)) from activity) as float),2) as

fraction from cte where datediff(day,prev\_date,event\_date)=1

23)select teacher\_id,count(distinct(subject\_id)) as cnt

from teacher

group by teacher\_id

24)select [activity\_date] as [day],count(distinct(user\_id)) as

active\_users from activity

where [activity\_date] between dateadd(day,-29,'2019-07-27')

and '2019-07-27'

group by [activity\_date]

25)with cte as

(

select p.product\_id,[year],

row\_number() over (partition by s.product\_id order by [year]) rn,

quantity,price from sales s join product p

on s.product\_id=p.product\_id

)

select product\_id,[year] as first\_year,quantity,price from cte where rn=1

26)select class from

(

select class,count(distinct(student)) as [no\_of\_students]

from courses

group by class

having count(distinct(student))>=5

) as d

27)select [user\_id],count(follower\_id) as followers\_count

from followers

group by [user\_id]

28)with cte as

(

select num,count(\*) as [repeated]

from mynumbers

group by num

having count(\*)=1

)

select max(num) as num from cte

29)with cte as

(

select customer\_id,count(distinct(product\_key)) as [products\_bought]

from customer

group by customer\_id

having count(distinct(product\_key))=(select count(product\_key) from product)

)

select customer\_id from cte

30)select e2.employee\_id,e2.name,count(e1.employee\_id) as reports\_count,

round(avg(cast(e1.age as float)),0) as average\_age

from employees e1 join employees e2

on e1.reports\_to=e2.employee\_id

group by e2.employee\_id,e2.name

order by e2.employee\_id

31)select employee\_id,department\_id

from employee

where primary\_flag='Y'

union

select e1.employee\_id,department\_id from

employee e1 join

(select employee\_id

from employee

group by employee\_id

having count(\*)=1

) e2

on e1.employee\_id=e2.employee\_id

order by employee\_id

32)select x,y,z,

case when x+y>z and y+z>x and x+z>y then 'Yes'

else 'No' end as triangle

from triangle

Note:- sum of any two sides must greater then 3rd side

33)with cte

as

(

select id,num,

lag(num,1) over(order by id) as [lag],

lead(num,1) over(order by id) as [lead]

from logs

)

select distinct(num)  as consecutiveNums from cte where num=[lag] and [lag]=[lead]

34)with cte as

(

select \*,row\_number() over (partition by product\_id order by diff\_date ) as r\_n

from

(

select product\_id,new\_price as price,change\_date,datediff(day,change\_date,'2019-08-16')

as diff\_date from products

) d

where diff\_date>=0

)

select distinct p.product\_id,isnull(price,10) as price from

(select product\_id,price from cte where r\_n=1) as w

right join products p on w.product\_id=p.product\_id

35)select top 1 person\_name from

(

select \*,sum([weight]) over (order by turn) as [running\_total]

from [Queue]

) s

where running\_total<=1000

order by running\_total desc

36)select category,accounts\_count

from

(

select count(case when income<20000 then 1 else null end) as'Low Salary',count(case when income between 20000 and 50000 then 1 else null end) as 'Average Salary',

count(case when income>50000 then 1 else null end) as 'High Salary'

from accounts

) as e

unpivot

(

accounts\_count for category in ([Low Salary],[Average Salary],[High Salary])

) as unp\_tbl

37)select employee\_id from employees

where manager\_id not in

(select employee\_id from employees) and manager\_id is not null

and salary<30000

order by employee\_id

38)select id,

case when id%2=1 then lead(student,1,student) over (order by id)

else lag(student,1,student) over (order by id) end as [student] from seat

39)select [name] as results from

(

select [name],row\_number() over (order by visits desc, name asc) as rn1

from

(select [user\_id],count(\*) as [visits]

from movieRating

group by [user\_id] ) m join Users u

on m.user\_id=u.user\_id

) as e where rn1=1

union all

select title as results

from

(

select title,row\_number() over (order by rat desc,title asc) as rn2 from

(

select movie\_id,avg(cast(rating as float)) as rat

from movierating

where created\_at<='2020-02-29'

group by movie\_id

) f join movies m

on f.movie\_id=m.movie\_id

) as q where rn2=1

40)with cte as

(

select \*,round(cast(amount as float)/7,2) as average\_amount

from

(

select visited\_on,sum(amount) over(order by visited\_on rows between 6 preceding and current row) as [amount] from

(select visited\_on,sum(amount) as amount from customer group by visited\_on) as a

) as b

)

select \* from cte where visited\_on>=

dateadd(day,6,(select min(visited\_on) from customer))

41)select top 1 id,count(id) as num from

(

select requester\_id as id from requestaccepted

union all

select accepter\_id as id from requestaccepted

) as a

group by id

order by num desc

43)with cte as

(

select departmentid,name,salary,dense\_rank() over (partition by departmentid order by salary desc) as rk from Employee

)

select d.name as Department, c.name as Employee,Salary

from cte c join department d on c.departmentid=d.id

where rk<=3

44)select [user\_id],upper(left(name,1))+lower(substring(name,2,len(name)))

as [name] from users

order by [user\_id] asc

45)select patient\_id,patient\_name,conditions

from patients

where conditions like 'DIAB1%' or conditions like '% DIAB1%'

46)with cte as

(

    select id,email,row\_number() over (partition by email order by id) as rn

    from person

)

delete from cte where rn>1

47)with cte as

(

    select id,salary,dense\_rank() over (order by salary desc) as dr

    from employee

)

select max(salary) as SecondHighestSalary from cte where dr=2

Note: - to get null in case of blank output use min, max functions as above

--listing duplicate rows

with cte AS

(

select \*,row\_number() over (partition by id,emp\_name,gender,salary,dept\_id,location\_id order by id ) as rn from EMP

 )

select count(\*) from cte where rn>1

select\*,count(\*) from EMP

group by id,emp\_name,gender,salary,dept\_id,location\_id

having count(\*)>1

--removing rows which are repeated

delete from emp

where id in (

select id

from

(

select\*,count(\*) as d from EMP

group by id,emp\_name,gender,salary,dept\_id,location\_id

having count(\*)>1

) as d

)

select \* from emp

**ProRank**

1)find nth highest salary

with cte AS

(

select \*,dense\_rank() over (order by salary desc) as DR from EMPLOYEE

)

select \* from cte where DR=2

2)select Id,emp\_name,gender,salary,dept\_id,location\_id from

(Select \*,dense\_rank() over (order by salary desc) as rn

from EMPLOYEE) h

where rn=3

3)with cte

AS

(

 select \*,row\_number() over(order by salary desc) as rn from

(select distinct(salary) as salary from employee) e

)

select e.\* from employee e join cte c

on e.salary=c.salary

where c.rn=3

select \*,sum(salary) over(order by id) as running\_total

from employee

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LearnatKnowStar**

1)remove duplicate records with unique id

with cte AS

(

select \*, row\_number() over (partition by firstname,lastname,phone,email order by employeeid) as rn

from Employee\_new

)

delete from cte where rn>1

select \* from Employee\_new

2)Nth highest rank

select id,emp\_name,gender,salary,dept\_id,salary,dept\_id,location\_id from

(

select \*,dense\_rank() over (order by salary desc) as [rank] from EMPLOYEE

) as e where rank=1

3)employee hierarchy by self join

select e1.employeeid,e1.EmployeeName as emp\_name,isnull(e2.EmployeeName,'Boss') as Manager from

employees e1 left join employees e2

on e2.EmployeeID=e1.ManagerID

4)shaping data using case

select id,

max(case when name='name' then value end) as name,

max(case when name='gender' then value end) as gender,

max(case when name='salary' then value end) as salary

from dc

group by id

5)shaping data using pivot

select id,[name],[gender],[salary] from

(

select id,name as name1,value from dc

) as e

pivot

(

  max(value) for name1 in ([name],[gender],[salary])

) pv\_table

6)Order month name by month number

select month\_name,salary from

(

select datename(month,join\_date) as month\_name,sum(salary) as salary,datepart(month,join\_date) as month

from employee e join JOIN\_DATE j

on e.id=j.id

group by datename(month,join\_date),datepart(month,join\_date)

 ) as d

 order by month

7)compare previous quarter

with cte AS

(

select year(join\_date) as year,'Q '+cast(datepart(quarter,join\_date) as varchar(10)) as quarter,sum(salary) as salary

from employee e join JOIN\_DATE j on e.id=j.id

group by year(join\_date),datepart(quarter,join\_date)

)

select \*,(salary-lag(salary,1,0) over (partition by year order by year,quarter))

as prev\_quarter from cte

order by year,quarter

8)extract first name and last name

create table split

(

  id int identity,

  name varchar(100)

 )

 insert into split values('pavan kumar'),('sasi kumar'),('babavali noorbasha')

 insert into split values('pavan')

select \*,

iif(charindex(' ',name)=0,name,substring(name,1,charindex(' ',name)-1)) as first\_name,

iif(charindex(' ',name)=0,' ',substring(name,charindex(' ',name)+1,len(name))) as last\_name

from split

9)extract first\_name and last\_name by string\_split

with cte AS

(

select \*,row\_number() over (partition by id order by id) as val from

(

select \* from split cross apply string\_split(name,' ')

) as e

)

select id,name,[1] as [first\_name],isnull([2],' ') as last\_name

from cte

pivot

(

  max(value) for val in ([1],[2])

) pv\_table

11)declare @start\_date as date='2022-01-01'

declare @end\_date as date='2022-12-31';

with cte as

(

  select @start\_date as [date]

  union all

  select dateadd(day,1,[date]) as [date] from cte

  where dateadd(day,1,[date])<=@end\_date

)

select count(\*) as no\_of\_weekdays from cte

where datepart(weekday,[date]) not in (1,7)

option(maxrecursion 500)

12)create function dob(@date as date)

returns int AS

BEGIN

    declare @age as int

    select @age=datediff(year,@date,getdate())

    return @age

End

select [dbo].dob('2021-08-02')

13)select \* from employee

where emp\_name like '%[0-9]%' and emp\_name like '%[a-z]%'

14)remove trailing zeroes

declare @val as varchar(10)='20.02000000'

select cast(@val as float) as val

15)extract id and name from single column

create table after\_split

(

  id int,

  name varchar(100)

 )

 while(select count(\*) from temp2)>0

 BEGIN

    declare @val as varchar(100)

    declare @id as varchar(10)=''

    declare @name as varchar(100)=''

    select top 1 @val=emp\_name from temp2

    declare @count as int=1

    while(len(@val)>=@count)

    BEGIN

    declare @char as char(1)=substring(@val,@count,@count)

    if @char like '[0-9]'

    BEGIN

        set @id=@id+@char

    end

    else if @char like '[a-z]'

    BEGIN

        set @name=@name+@char

    end

    set @count=@count+1

    end

    insert into after\_split values(cast(@id as int),@name)

    delete top (1) from temp2

 end

 select \* from after\_split

16)calculating running and cumulative total

select \*,sum(salary) over (order by id) as [running\_total]

from employee

17)declare @start\_date as date

declare @end\_date as date

select @start\_date=min(sale\_date),@end\_date=max(sale\_date) from sales;

with cte as

(

select @start\_date as [date]

 union all

select dateadd(day,1,[date]) as [date] from cte where dateadd(day,1,[date])<=@end\_date

),

cte2 as

(

  select id,[date],isnull(amount,0) as amount

  from cte c left join sales s on c.[date]=s.sale\_date

),cte3 as

(

select \*,year(date) as [year],datepart(quarter,[date]) as [quarter],

month(date) as [month] from cte2

)

select \*,sum(amount) over (partition by [year],[quarter],[month] order by [date]) as MTD,

sum(amount) over (partition by [year],[quarter] order by [date]) as QTD,

sum(amount) over (partition by [year] order by [date]) as YTD

from cte3

option(maxrecursion 2000)

18)with cte as

(

  select \*,year(sale\_date) as [year],datepart(quarter,sale\_date) as [quarter],month(sale\_date) as [month]

  from sales

)

select \*,first\_value(id) over (partition by [year],[quarter],[month] order by [sale\_date]) as first\_order,

Last\_value(id) over (partition by [year],[quarter],[month] order by [sale\_date]

                    rows between current row and unbounded following) as last\_order

from cte

--here we using frame clause for Last\_value function because it return same row value as last value for each row so we need to define the frame from which it need to take last value. Each row consider it as last row by default to avoid it we define frame clause.

--we can use rows between unbounded preceding and current row in First\_value or direct First\_value no difference.

19)with cte as

(

  select emp\_name,salary,dept\_id from EMPLOYEE e1

  where salary<(select avg(salary) from employee e2 where e1.dept\_id=e2.dept\_id)

)

,cte2 as

(

select c.\*,isnull(g.avg\_salary,0) as avg\_salary from cte c

cross join

(select dept\_id,avg(salary) as avg\_salary from employee group by dept\_id) g

where c.dept\_id<>g.dept\_id

),cte3 as

(

select emp\_name,count(case when salary>avg\_salary then 1 else null end) as [flag] from cte2

group by emp\_name

)

select emp\_name from cte3

where [flag]=(select count(distinct(dept\_id))-1 from EMPLOYEE)

20)with cte as

(

select emp\_name,salary,dept\_id,dense\_rank() over (partition by dept\_id order by salary desc) as rk

from employee

)

select emp\_name,salary,dept\_id from cte

where rk=1

21)select Emp\_name

from EMPLOYEE E2

where salary>(select avg(salary) as avg\_sal from EMPLOYEE E1 where E1.dept\_id=E2.dept\_id group by E1.dept\_id)

and salary<(select avg(salary) from EMPLOYEE)

22)with cte as

(

  select id,Emp\_name,Salary,dept\_Head\_id from

  employee e join DEPARTMENT d on e.DEPT\_ID=d.DEPT\_ID

)

select E1.Emp\_name,E1.salary,E2.salary as manager\_salary

from cte E1 join cte E2

on E1.dept\_head\_id=E2.id

where E1.salary>E2.salary

23)with cte as

(

  select E.\*,join\_date from

  EMPLOYEE e join JOIN\_DATE j on e.id=j.id

)

select \*,

datediff(year,join\_date,getdate())\*2500+salary as [new\_salary] from cte

where datediff(year,join\_date,getdate())>=2

25)two alphabet sequence

create table sequences

(

  id int identity(1,1),

  val varchar(100)

 )

 declare @val as int=65

 while(@val<=90)

 BEGIN

    declare @v1 as varchar(10)=char(@val)

    declare @v2 as int=65

    while(@v2<=90)

    BEGIN

        declare @v3 as varchar(10)=char(@v2)

        insert into sequences VALUES(@v1+@v3)

        set @v2=@v2+1

     end

     set @val=@val+1

 end

 select \* from sequences

26)shaping data

select location\_id,year,salary from

(

select location\_id,[2012],[2013],[2014],[2015],[2016],[2017],[2018],[2019],

[2020],[2021],[2022],[2023],[2024],[2025] from

(

select location\_id,year(join\_date) as year,sum(salary) as salary from EMPLOYEE e join JOIN\_DATE j

on e.ID=j.ID

group by location\_id,year(join\_date)

) as d

pivot

(

  sum(salary) for year in ([2012],[2013],[2014],[2015],[2016],[2017],[2018],[2019],

[2020],[2021],[2022],[2023],[2024],[2025])

  ) as pv\_table

  ) as f

  unpivot

  (

    salary for year in ([2012],[2013],[2014],[2015],[2016],[2017],[2018],[2019],

[2020],[2021],[2022],[2023],[2024],[2025])

  ) as unpivot\_table

28)with cte AS

(

  select E.\*,j.join\_date

  from employee e join JOIN\_DATE j

  on e.ID=j.ID

 )

 select \* from cte

 where join\_date between dateadd(month,-40,join\_date) and join\_date

29)Declare @name as varchar(100)='Thota PaVan KUmar'

select @name as Namel,

upper(substring(@name,1,1))+lower(substring(@name,2,len(@name))) as Name2

30)with cte AS

(

  select E.\*,j.join\_date

  from employee e join JOIN\_DATE j

  on e.ID=j.ID

 )

 select \* from cte

 where eomonth(getdate())=join\_date

32)Declare @name as varchar(100)='Thota Pavan Kumar'

select len(@name)-len(replace(@name,'a','')) as [Occurance\_of\_a]

33)Select \* from employees

where Employeename like 'a%'

Select \* from employees

where Employeename like '%a'

Select \* from employees

where Employeename like '%a%'

Select \* from employees

where Employeename like '\_a%'

Select \* from employees

where Employeename like '[mld]%'

Select \* from employees

where Employeename like '[a-z]%'

Select \* from employees

where Employeename like '[^a-z]%'

Select \* from employees

where Employeename like '[^a-z0-9]%'

Select \* from employees

where Employeename like '[0-9]%'

34)with cte AS

(

select dept\_id,[male],[female] from

(

select dept\_id,gender,count(\*) as gender\_count

from EMPLOYEE

group by dept\_id,gender

 ) as G

 pivot

 (

   sum(gender\_count) for gender in ([male],[female])

  ) as pivot\_table

 )

 select dept\_id,case when male is null then 'no\_male'

 else 'no female' end as types

 from cte

 where male is null or female is null

35)declare @name as varchar(100)='Thota Pavan Kumar'

select @name where @name='thota pavan kumar'

declare @name as varchar(100)='Thota Pavan Kumar'

select @name where @name collate latin1\_general\_cs\_as ='thota pavan kumar'

36) diff between count(\*), count(1), count(column)

There is no difference between count(\*) and count(1) both are same.

Difference between count(\*) and count(column) is count(\*) counts nulls also where count(column) does not count null.

37)create table cat( [name] varchar(100),[2015] int,[2016] int,[2017] int,[2018] int,[2019] int)

insert into cat values ('bike',20000,3000,23000,120000,3000),('cars',null,20000,23000,12000,234000)

a)select [name],

(select max(sales) from (values([2015]),([2016]),([2017]),([2018]),([2019])) sale\_tbl(sales)) as max\_val

from cat

b)with cte as

(

select name,[year],sales from cat

unpivot

(

  sales for [year] in ([2015],[2016],[2017],[2018],[2019])

) as un\_pivot

)

select name,max(sales) as max\_val from cte

group by [name]

38)with cte as

(

select \*,dateadd(day,-row\_number() over (order by sale\_date),sale\_date) as [n] from sales

where sale\_date between '2023-01-01' and '2023-01-31'

)

select min(sale\_date) as start\_date,max(sale\_date) as end\_date,

count(\*) as [no\_of\_days] from cte

group by n

39)with cte as

(

select \*,datediff(day,sale\_date,lead(sale\_date) over (order by sale\_date))-1 as [n] from sales

where sale\_date between '2023-01-01' and '2023-01-31'

)

select dateadd(day,1,sale\_date) as [start\_date],

dateadd(day,n,sale\_date) as end\_date,n as no\_sales\_days from cte

where n>=1

40)Select \*,

dateadd(day,-datepart(weekday,join\_date)+2,join\_date) as week\_start,

dateadd(day,8-datepart(weekday,join\_date),join\_date) as week\_end

from JOIN\_DATE

41)with cte as

(

select \*,dense\_rank() over (partition by dept\_id order by salary desc) as [rank] from employee

 ), cte2

 AS

 (

 select emp\_name,salary,dept\_id from cte where [rank]<=3

 ), cte3 as

 (

   select dept\_id,sum(case when [rank] not in (1,2,3) then salary else 0 end) as salary from cte

   group by dept\_id

 )

 select \* from cte2

 union

 select 'others' as emp\_name,salary,dept\_id from cte3

 order by dept\_id

42)

a)previous month

with cte as

(

select year(sale\_date) as [year],month(sale\_date) as [month],

sum(amount) as [sales] from sales

group by year(sale\_date),month(sale\_date)

)

SELECT \*,lag(sales,1,sales) over (order by year,month) as [prev\_month\_sales]

from cte

b)previous year

with cte as

(

select year(sale\_date) as [year],month(sale\_date) as [month],

sum(amount) as [sales] from sales

group by year(sale\_date),month(sale\_date)

)

SELECT \*,lag(sales,12,sales) over (order by year,month) as [prev\_month\_sales]

from cte

43)declare @domin as varchar(100)='Pavankumar@cognizant.com'

select substring(@domin,charindex('@',@domin)+1,patindex('%.com%',@domin)-1-charindex('@',@domin)) as domin\_name

44)create table dyn\_sql( id int,[column] varchar(100),[values] varchar(100))

insert into dyn\_sql values (1,'name','pavan'),(1,'age','24'),(1,'gender','male')

,(2,'name','sasi'),(2,'age','23'),(2,'gender','male'),(3,'name','babavali'),(3,'age','25'),

(3,'gender','female')

declare @col as varchar(max)

select @col=string\_agg([column],',') from (select distinct([column]) from dyn\_sql) as d

declare @sql varchar(100)

set @sql='select id,'+@col+' from dyn\_sql pivot (min([values]) for [column] in ('+@col+') ) as pivot\_tab'

select @sql

execute (@sql)

46)declare @year as int=year(getdate())

if @year%400=0 or (@year%4=0 and @year%100<>0)

BEGIN

    select cast(@year as varchar(100))+' is a leap year'

end

else

BEGIN

    select cast(@year as varchar(100))+' is not a leap year'

end

47)declare @start\_date as date='2022-01-01'

declare @end\_date as date='2022-12-31';

with cte AS

(

  select @start\_date as [date]

  union all

  select dateadd(day,1,[date]) as [date] from cte where [date]<=@end\_date

 )

 select \* from cte

 option(maxrecursion 500)

48)Difference between replace and translate

Declare @name as varchar(100)='Thota Pavan Kumar'

select replace(@name,'kumar','naidu') as [replace]

select Translate(@name,'kumar','naidu') as [Translate]

replace changes the old string with new string entire pattern match.

Translate changes character to character.

49)Difference between rank(),Dense\_rank(),Row\_Number()

Rank() gives rank to the values in the table if multiple values have same rank next ranks will be skipped.

Dense\_rank() is same as rank() but when multiple values have same value the next ranks will not skipped.

Row\_number() it used to give incremental number to the rows. Based on column in order by clause.

Rank-1,2,2,4,5

Dense\_rank()-1,2,2,3,4

Row\_number()-1,2,3,4,5

50)with cte AS

(

select dept\_id,avg(salary) as avg\_sal

from EMPLOYEE

group by dept\_id

),cte2 as

(

select \*,dense\_rank() over (partition by dept\_id order by diff ) as [rank]

from

(

select E.\*,c.avg\_sal,abs(salary-C.avg\_sal) as diff from

employee e join cte c ON

e.dept\_id=c.dept\_id

) as d

)

select \* from cte2 where [rank]=1

51)declare @string as varchar(100)='yes,no,yes,no,yes,yes,yes,yes,no,yes'

declare @substring as varchar(100)='no'

select (len(@string)-len(replace(@string,@substring,'')))/len(@substring) as [substring\_occurance]

52)create table sales( id int identity(1,1),[sale\_date] date,[amount]  int)

declare @start\_date as date='2021-01-01'

declare @end\_date as date='2024-12-31';

with cte as

(

  select @start\_date as [date]

  union all

  select dateadd(day,1,[date]) as [date] from cte where dateadd(day,1,[date])<=@end\_date

)

insert into sales

select [date],round(rand()\*100,0) as [amount] from cte

option(maxrecursion 2000)

--use delete to miss some records

declare @start\_date as date

declare @end\_date as date

select @start\_date=min(sale\_date) from sales

select @end\_date=max(sale\_date) from sales;

with cte as

(

  select @start\_date as [date]

  union all

  select dateadd(day,1,[date]) as [date] from cte where dateadd(day,1,[date])<=@end\_date

)

select c.[date],isnull(amount,0) as amount from

sales s right join cte c on s.sale\_date=c.[date]

order by [date]

option(maxrecursion 2000)

53)Declare @val as int=1000

select format(@val,'000000000000#')-#represents value of variable.

54)declare @name as varchar(100)='pavan,,,,,kumar,,,,naidu,,,,,thota'

select replace(replace(replace(@name,',',',\*'),'\*,',''),'\*','') as [new\_string]

55)select \*,

case when emp\_name='Employee 233' then 1 else 0  end as cust\_sort

from employee

order by cust\_sort desc, salary desc

57)declare @depts as varchar(100)='1,2,3,4,5'

select \* from EMPLOYEE

where dept\_id in (select value from string\_split(@depts,','))

58)declare @nameid as varchar(100)='Pavan@213'

declare @count as int

select @count=len(@nameid)

declare @name as varchar(100)=''

declare @id as varchar(100)=''

declare @val as int=1

while (@count>0)

BEGIN

    declare @char as char(1)=substring(@nameid,@val,1)

    if (@char like '[a-z]')

    begin

        set @name=@name+@char

    end

    else if (@char like '[0-9]')

    begin

        set @id=@id+@char

    end

    set @count=@count-1

    set @val=@val+1

end

select @name,@id

59)create table alphanumeric( id varchar(100))

insert into alphanumeric values ('pavan13'),('212'),('210'),('sasi189'),('babavali200'),('199'),('purna123')

select id,left(id,patindex('%[0-9]%',id)-1) as [alpha],

cast(right(id,len(id)-patindex('%[0-9]%',id)+1) as int) as [numeric]

from alphanumeric

order by [numeric] asc

60)declare @name as varchar(100)='pavankumar157999@gmail.com'

select replace(@name,'@gmail','@hotmail') as new\_name

62)declare @password varchar(100)=''

declare @len as int=8

while(@len>0)

BEGIN

    declare @val as int

    select @val=abs(checksum(newid())%26)

    if (@len>=4)

    begin

        set @password=@password+char(65+@val)

    end

    else if (@len=3)

    BEGIN

    declare @val2 as int

    declare @char as varchar(100)

    select @val2=abs(checksum(newid()))%5+1

    select @char=choose(@val2,'@','#','!','\*','%')

    set @password=@password+@char

   end

   else

   begin

        set @password=@password+cast(@val as varchar(100))

   end

   set @len=@len-1

 end

select @password

63)declare @password varchar(100)=''

declare @len as int=8

while(@len>0)

BEGIN

    declare @val as int

    select @val=round(rand()\*2,0)

    if (@len>=4)

    begin

        set @password=@password+char(65+@val)

    end

    else if (@len=3)

    BEGIN

    declare @val2 as int

    declare @char as varchar(100)

    select @val2=abs(checksum(newid()))%5+1

    select @char=choose(@val2,'@','#','!','\*','%')

    set @password=@password+@char

   end

   else

   begin

        set @password=@password+cast(@val as varchar(100))

   end

   set @len=@len-1

 end

select @password

65)select dept\_id, string\_agg(Emp\_Name,' | ') as Emp\_names

from Employee

Group by Dept\_id

--if we have null it does not ci=use any problem if no emp for the department it reytun null

69)declare @num as varchar(100)='0000000233'

select iif(patindex('%[^0]%',@num)=0,0,substring(@num,patindex('%[^0]%',@num),len(@num)-patindex('%[^0]%',@num)+1)) as [num]

70)update emp\_dup set emp\_name=gender,gender=emp\_name

select \* from emp\_dup

select \* from employee

update emp\_dup set emp\_name='F' where emp\_name='Male'

update emp\_dup set emp\_name='M' where emp\_name='Female'

update emp\_dup set emp\_name='Male' where emp\_name='M'

update emp\_dup set emp\_name='Female' where emp\_name='F'

71)create table reverse\_pair( id1 int,id2 int)

insert into reverse\_pair values(1,2),(3,4),(2,3),(2,2),(3,2),(2,1),(8,1)

select distinct r1.\* from reverse\_pair r1

join reverse\_pair r2

on r1.id1=r2.id2 and r2.id1=r1.id2 and r1.id1<>r1.id2

72)select isnull(cast(year(sale\_date) as varchar(100)),'Grouped by all Years') as [year],

isnull(cast(datepart(quarter,sale\_date) as varchar(100)),'grouped by al months of '+cast(year(sale\_date) as varchar(100))) as [quarter],

isnull(cast(month(sale\_date) as varchar(100)),'grouped by all months of Q'+cast(datepart(quarter,sale\_date) as varchar(100))+' of '+

       cast(year(sale\_date) as varchar(100))) as [month]

,sum(amount) as [sales]

from sales

group by rollup(year(sale\_date),datepart(quarter,sale\_date),month(sale\_date))

74)select \*,

sum(salary) over (order by id)/ count(id) over (order by id) as [moving\_average]

from employee

75)with cte as

(

select \*,

lag(num,1) over(order by id) as [lag],

lead(num,1) over (order by id) as [lead]

from logs

)

select num from cte where num=[lag] and [lag]=[lead]

77)Mock interview questions

a)what is primary key and why it needed.

Primary key is a column or combination of columns which are used to uniquely identity the records in the table.

Primary key must be not null and unique. It used to uniquely identity records in a table. Using this primary key with foreign key establishes relations between table.

b)What is best way to auto increment values.

By using identity, we can auto increment value. Identity(start, step).

Identity(1,1) same as identity.

Mostly used identity column as primary column.

c)Convert left join to inner join with where condition

select count(\*) from employee e

left join DEPARTMENT d

on e.DEPT\_ID=e.DEPT\_ID

select count(\*) from employee e

left join DEPARTMENT d

on e.DEPT\_ID=e.DEPT\_ID

where d.dept\_id is not null

d)output

declare @val as int=99

select case when @val<100 then 1

            when @val>90 then 0

            end as [val]

--it return 1 first matching condition.

e)How improve select query performance.

use column names instead of \*

use two-part name for table name.

Avoid using distinct.

f)difference between union and union all

union returns result sets with out any duplicates where union all returns result set with duplicates.

Performance wise use union all use union only if it required.

80)create table fill\_up( id int identity(1,1), val  int)

insert into fill\_up values (1),(2),(null),(null),(null),(4),(4),(1),(null),

(null),(5),(null),(null),(null),(null),(3),(4),(null)

with cte AS

(

select \*,

count(val) over (order by id) as coun from fill\_up

)

select id,val,first\_value(val) over (partition by coun order by id) as [fill\_down]

from cte

83)DECLARE @name as varchar(100)='Thota Pavan Kumar kumar'

select charindex('kum',@name) as charindex

select patindex('%kum%',@name) as patindex

select charindex('k',@name) as charindex

select patindex('%kun%',@name) as patindex

charindex used to return the index of character we specified at first occurrence if character not found return 0.

Patindex used to return the index of pattern we specified at first occurrence if character not found returns 0.

Patindex used to find pattern where charindex used to find character in giveb string if they not found both return 0.

84)select e.empid,isnull(d.deptid,de.deptid) as dept\_id,isnull(d.deptname,de.deptname) as dept\_name,e.IsExternal

from emp e

left join dept d on e.DeptID=d.DeptID and isexternal<>'x'

left join DeptExt de on e.DeptID=de.DeptID and isexternal='x'

85)declare @filename as varchar(100)='C:\Users\2125208\OneDrive - Cognizant.com\Pictures\Screenshots\pavan.txt'

select right(@filename,charindex('.',reverse(@filename))-1) as file\_extension

select right(@filename,charindex('\',reverse(@filename))-1) as file\_name

86)select Dept\_id,

count(case when salary>70000 then 1 else null end) as high\_salary\_count,

sum(case when salary>70000 then salary else 0 end) as high\_salary\_sum,

sum(case when salary>70000 then salary else 0 end)

/iif(count(case when salary>70000 then 1 else null end)=0,1,count(case when salary>70000 then 1 else null end))

as high\_salary\_avg

from employee

group by dept\_id

87)WITH CTE AS

(

select case when c1.Productid>c2.Productid then c1.Productid else c2.Productid end AS P1,

CASE WHEN C1.Productid>C2.Productid THEN C2.Productid ELSE C1.Productid END AS P2

from CustOrders c1

join CustOrders c2

on c1.Orderid=c2.Orderid and c1.Productid<>c2.Productid

),

cte2 AS

(

Select p1,p2,count(\*) as [no\_of\_occurance] from cte

group by p1,p2

)

select top 2  p1,p2 from cte2

order by no\_of\_occurance desc

89)declare @sql as varchar(max)

declare @col as varchar(100)

select @col=string\_agg(name,',') from sys.columns

where [object\_id]=object\_id('tbl1')

set @sql='select \* from tbl1 where ''Cancelled'' in ( '+@col+' )'

select @sql

execute (@sql)

90)with cte as

(

select cast(replicate(' ',9)+'\*' as nvarchar(100)) as [level],1 as n

union all

select cast(replicate(' ',9-n)+replicate('\*',(2\*n)+1) as nvarchar(100)) as [level],n+1 as n from cte

where n<10

)

select \* from cte

91)create table ipl\_fixer (id int identity(1,1), team varchar(100))

insert into ipl\_fixer values ('CSK'),('SRH'),('MI'),('KKR'),('LSG'),('GT'),('RCB')

,('PK'),('DC'),('RR')

select t1.team as team1, t2.team as team2

from ipl\_fixer t1 cross join ipl\_fixer t2

where t1.id<t2.id

93)Declare @column as varchar(100)

set @column='Age'

if not exists(select \* from sys.columns where [object\_id]=object\_id('tbl1') and name=@column)

begin

    alter table tbl1 add age int

    select 'Column created'

End

else

Begin

    alter table tbl1 drop column age

    select 'Column dropped'

End

94)declare @start\_date as date='2022-01-01'

declare @end\_date as date='2022-12-31';

with cte as

(

  select @start\_date as [date]

  union all

  select dateadd(day,1,[date]) as [date] from cte where dateadd(day,1,[date])<=@end\_date

)

select count(\*) as [count\_of\_sunday] from

(

select date,datepart(weekday,date) as weekday from cte

) as d where weekday=1

option(maxrecursion 500)

95)declare @date as date='2024-04-09'

select dateadd(day,-datepart(weekday,@date),@date) as start\_of\_week,

dateadd(day,7-datepart(weekday,@date),@date)  as end\_of\_week

98)difference between delete vs truncate

|  |  |
| --- | --- |
| Delete | Truncate |
| Delete used to remove specific records from table | Truncate used to remove all records from table. |
| It is DML command | It is DDL command |
| We can use where clause | We cannot use where clause |
| It can be rolled back. | It cannot be rolled back. |

99)declare @start\_date as date='2022-01-01'

declare @end\_date as date='2022-01-31';

with cte as

(

  select @start\_date as [date]

  union all

  select dateadd(day,1,[date]) as [date] from cte where dateadd(day,1,[date])<=@end\_date

),cte2 as

(

select b,month(date) as [month],count(distinct(date)) as [no\_days]

from

(

select \*

from cte

cross join

(select \* from (values(1),(2),(3),(4),(5)) a(b)) n

) g

group by b,month(date)

)

select b from cte2 c2 join

(select distinct month(date) as [month],day(eomonth(date)) as [no\_of\_days]

from cte) c1 on

c1.month=c2.month

where no\_days=no\_of\_days

100)select isnull(cast(dept\_id as varchar(10)),'total') as dept\_id,isnull(gender,'dept\_wise')as gender,sum(salary) as salary

from EMPLOYEE

group by dept\_id,gender with rollup

104)Common mistakes using date time in where clause

Use correct date format yyyy-mm-dd

For time hh:mm:ss.msmsms

2023-02-02 23:59:59:999 –it rounds to 2023-02-03 be careful while using it.

If we ant records between 2023-02-02 00:00:00.000 to 2024-02-05 23:59:59.999 then use greater than 2023-02-02 00:00:00.000 and less than 2023-02-05 00:00:00.000

105)Merge [target] as T

using [source] as S

on s.id=t.id

when matched then

update set t.id=s.id,t.emp\_name=s.emp\_name,t.gender=s.gender,t.salary=s.salary,t.dept\_id=s.dept\_id,t.location\_id=s.location\_id

when not matched by [t] then

insert (id,emp\_name,gender,salary,dept\_id,location\_id) values(s.id,s.emp\_name,s.gender,s.salary,s.dept\_id,s.location\_id)

when not matched by [s] then

delete;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Interview Happy**

1)Difference between DBMS and RDBMS

|  |  |
| --- | --- |
| DBMS | RDBMS |
| Database Management System | Relational Database Management System |
| Data stored in files | Data stored in tables |
| Example XML,JSON etc. | MS SQL Server, Oracle, MYSQL etc. |
| Normalization not possible | Normalization is possible |
| Handles less data | Handles large data |
| One user at a time | It provides multiple users at a time |

2)Constraints in SQL Server

Constraints are used to specify the rules on data we entering into table. We specify constraints on column.  
there are following constraints in SQL Server

Null it allows nulls in to column.

Not null it does not allows nulls in to column.

Default it gives default value if no value is specified.

Check it allows value in a specified limit.

Unique key it does not allow duplicates. It allows only one null.

Primary key it used to unique identification of records in the table it does not allow nulls and duplicates.

Foreign key refers primary key in other table. It used to maintain data integrity [referential integrity]

3)Difference between unique and primary key.

Unique key allows on one null value where primary key does not allow nulls.

We can have multiple unique keys in a table, but primary key must one.

Both does not allow duplicates.

4)What are triggers and it types.

Triggers are special types of stored procedures which gets fired when ever an event occurs in a database or at server level.

Triggers are classified as follows

Data Manipulation Triggers

Instead of Triggers- Instead of insert, Instead of update, Instead of delete.

After triggers\_ Insert, Delete, Update

Data Define Triggers-Create\_table, Drop\_Table, Alter\_table, Rename etc. we can define them on server level or database level.

Logon Triggers- triggers based on login used to limit no of sessions by user, to limit user’s login etc.

5)Difference Between Where and Having Clause,

|  |  |
| --- | --- |
| Having Clause | Where Clause |
| Used to filter data after getting aggregated | Used to filter data before data get aggregated |
| Group by clause is mandatory. | Group by is not mandatory |
| We use column with aggregate function in Having clause | In where clause we cannot use aggregate function. |
| Used after group by clause | Used before group by clause. |

6)What is subquery or nested query.

Query in a query is called sub query.

It enclosed in parentheses ().

We can use sub query in select, from, where, having clauses.

Sub query is two type’s

Correlated subquery inner query executes for each row of outer query. Inner query depends upon the outer query.

Non correlated subquery here sub query does not depends upon the main query. It can execute individually.

7)Identity

Identity is used to auto increment the value in the column. It generates numeric value.

We specify start and step value. Identity(start,stop)

Identity without start and stop means identity(1,1).

Most of times identity column used as primary key.

8)What are joins.

Joins are used to join the two tables based on the columns.

9)Types of Joins.

Inner Join- it returns matching records from both the tables.

Left outer Join- it returns all records from the left table and matching records from the right table.

Right outer join – it returns all records from the right table and matching record from the left table.

Full join – it returns matching and non-matching records from both tables.

Cross join – it returns cartesian product of two tables ( all possible combinations ). Table a has 2 rows table b has 3 rows we get 2\*3=6 rows.

10)Self join

Joining a table with the same table is called self-join which used to find employee manager hierarchy i.e. same table acts as both employee and manger tables.

11)Fetch employees who are manager also.

select \* from employees

where employeeID in (Select managerID from employees)

12)What is index

Index used to speed up the SQL query. They are same as index in book. They ordered data in a specific order so we can fetch them quickly.

13)Clustered Index.

Clustered index defines physical order of the table. Clustered indexes are stored in the table. One clustered index is allowed per table. When we create as primary key a unique clustered index is created. At leaf node we have data in clustered index.

14)Non-Clustered index.

Non clustered index does not define physical order of the table. Non-Clustered index are stored separately. We can create multiple non clustered index on a table up to 999. Leaf node of the non clustered index points to actual data.

15Difference between clustered and non-clustered

|  |  |
| --- | --- |
| Clustered Index | Non-Clustered Index |
| Defines physical order of the table | It does not defines physical order of table. |
| We can have only one clustered index. | We can have multiple non clustered index up to 999 |
| They stored with in the table and are fast comparatively. | They stored separately and are slow comparatively. |
| Leaf node has actual data | Leaf node points to actual data. |

16)How to create index on table.

Create Unique Clustered Index UNCI on Employee (Emp\_Id desc)

Create Unique NonClustered Index UNCI on Employee (Emp\_Id desc)

Create Clustered Index UNCI on Employee (Emp\_Id asc)

Create NONClustered Index UNCI on Employee (Emp\_Id desc)

17)Which column we need to use index.

Create index on columns which are mostly used in where clause. To improve performance.

18)Difference between stored procedure and function

|  |  |
| --- | --- |
| Stored Procedure | Function |
| No need of input parameters | Need input parameters |
| Not mandatory to return a value. | Mandatory to return value. |
| We cannot use stored procedure in select clause | We can use function in select clause. |
| We can call a function in stored procedure. | We cannot call a stored procedure from function |
| Stored procedures support try catch and transaction | Functions does not supports support try catch and transaction. |

19)How to increase performance of stored procedure.

Set nocount on.

Use column names instead on \*

Avoid using dynamic SQL they are vulnerable to SQL Injections.

Use two part name for objects like dbo.table\_name

Use transactions only if they needed.

20)what is cursor

Cursors are special data types used to iterate the data row by row.

It takes very long time to iterate row by row so try to avoid it as much as possible.

Steps to work with cursor

Create cursor

Open

Fetch next

Close

Deallocate – to delete

21)what is cte.

Cte means common table expression.

It uses to hold the temporary result set we need to use cte right below it created. To create cte we use with keyword. With single with we can create multiple cte. We have recursive cte to which refers itself until condition is true.

We can DML on table using cte.

22)Difference between delete, drop, truncate

Delete is DML, used to remove specific records in a table, it can be rolled back.

Drop used to delete the table.

Truncate removes all rows in table it is DDL it cannot be rolled back. We cannot use where clause here.

23)Nth highest Salary.

with cte AS

(

select \*,dense\_rank() Over (order by salary Desc) as [rank]

from Employee

)

Select Id ,Emp\_Name ,Gender, Salary, Dept\_Id, Location\_Id

from Cte

Where [Rank]=11

24)ACID properties

Acid properties are the properties which are possessed by transactions.

1. Atomicity means all commands in a transaction must execute if any one command does not executes then all commands must be rolled back.

C-Consistency data which is touched by transaction must be consistent before and after the transaction.

I-Isolation Each transaction must be isolate each other.

D-Durability any change done by transaction must reflects in database and it must be permanent.

25Magic tables

Magic tables are the table which are created by SQL Server when ever and DML occurs [delete, update, insert].

Inserted and deleted are two magic tables available in SQL Server. We can access this magical tables with trigger.

When ever a new row inserted it stored in inserted .When ever a row is delete it stores in deleted

When an update occurs then record with old values stored in deleted and record with new values stored in inserted. Eg 1,’pavan’,22 changed using update to 1,’pavan’,24 then 1,’pavan’,22 stored in deleted and 1,’pavan’,24 stored in inserted.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**SS UNITECH**

2)declare @c as int

select @c=count(\*) from Comma\_Separated\_Tbl;

with cte as

(

 select \*,

  case when wk='M' then 1

  when wk='T' then 2

  when Wk='W' then 3

  When Wk='TH' then 4

  when Wk='F' then 5

  when Wk='Sa' then 6

  Else 7 end as [flag]

  from Comma\_Separated\_Tbl

  order by training,flag desc

  offset 0 rows

  fetch next @c rows only

)

select Training, Classroom, StartTime,

duration,string\_agg(wk,',') as wK

from cte

Group by Training, Classroom, StartTime, duration

3)What happen we update same table in a update trigger on same table.

Trigger does not goes recursive i.e. if we update table in a update trigger of same table it just updates does not trigger again the update trigger when update happen in update trigger.

4)select s1.Name as son,s1.Fname as Father,S2.Fname as Grand\_ather from

Son\_Father\_Tbl s1 left join Son\_Father\_Tbl s2

on s1.Fname=s2.Name

6)

a)If multiple statements satisfy condition in case statement it return result for first condition match.

b) create table shaping

(

  id int identity(1,1),

  english int,

  maths int,

  science INT

)

insert into shaping values

(80,70,60),(60,50,60),(50,90,90),(30,30,40),(23,30,23),(23,23,23),(34,89,90);

with cte as

(

select id,subject,marks from shaping

unpivot

(

  marks for subject in ([english],[maths],[science])

) as unpivot\_table

), cte2 AS

(

  select \*,row\_number() over (order by id) as r1,

  row\_number() over (partition by id order by id) as r2 from cte

)

select \* from cte2

7)declare @val as int

select @val=count(\*)-1 from sys.columns where [object\_id]=object\_id('shaping');

with cte as

(

select id,subject,marks from shaping

unpivot

(

  marks for subject in ([english],[maths],[science])

) as unpivot\_table

),cte2 as

(

select \*,

case when marks>=70 then 'merit'

     when marks>=50 then 'pass'

     else 'fail' end as [result]

from cte

),cte3 as

(

select id,

count(case when result='merit' then 1 else null end) as merit,

count(case when result='pass' then 1 else null end) as pass,

count(case when result='fail' then 1 else null end) as fail

from cte2

group by id

)

select id,case when fail=0 and (pass+merit)=@val then 'good'

            when fail=1 and (pass+merit)=@val-1 then 'average'

            when fail=@val then 'bad'end  as [grade]

from cte3

8)select \* from Employee\_new

where firstname like 'a%'

select \* from Employee\_new

where firstname like '%m'

select \* from Employee\_new

where firstname like 'a%m'

select \* from Employee\_new

where firstname like '\_a%'

select \* from Employee\_new

where firstname like '%ar%'

select \* from Employee\_new

where firstname like 'a\_\_%'

select \* from Employee\_new

where firstname like 'a%' and len(firstname)>=3

select \* from Employee\_new

where firstname like '[a,m,e]%'

select \* from Employee\_new

where firstname like '[^a,m,e]%'

select \* from Employee\_new

where firstname like '[a-m]%'

select \* from Employee

where Emp\_name like '%[\_]%'

select \* from Employee\_new

where firstname like '%[%]%'

10)CREATE TABLE TEAM

(

TEAMID     INT,

TEAMNAME   VARCHAR(20)

)

INSERT TEAM VALUES (1,'IND'),(2,'AUS'),(3,'SL'),(4,'PAK')

select T1.teamname+' vs '+T2.teamname as [fixture]

from TEAM T1 cross join Team T2

where t1.TEAMID>t2.TeamID

select T1.teamname+' vs '+T2.teamname as [fixture]

from TEAM T1 join TEAM T2 on T1.TEAMID>T2.TEAMID

11)Create Table Item\_Master

(

Id   Int ,

Did  Varchar(20)

)

Create Table Variant\_Master

(

Id     Int,

Name   Varchar(10)

)

Insert Item\_Master Values (1,'1,2'),(2,'3,4'),(3,'5,4'),(4,'3,4'),(5,'11,2')

Insert Variant\_Master Values (1,'A'),(2,'B'),(3,'C'),(4,'D'),(5,'E'),(11,'K');

with cte as

(

select id,Value from Item\_Master cross apply string\_split(did,',')

),cte2 as

(

select c.id,v.name from Variant\_Master v join cte c on v.id=c.value

)

select id,string\_agg(name,',') as vdesc from cte2

group by id

select \* from Variant\_Master

select \* from Item\_Master

12)declare @st as varchar(100)='pavan231kumar5739goat736'

declare @len as int

select @len=len(@st)

declare @sum as int=0

declare @val as int=1

while(@len>=1)

begin

    declare @char as char(1)=substring(@st,@val,@val)

    if @char like '[0-9]'

    Begin

        set @sum=@sum+cast(@char as int)

    End

    set @len=@len-1

    set @val=@val+1

End

Select @sum

14)with cte as

(

select id,Subject,Marks

from shaping

unpivot

(

  marks for subject in ([english],[maths],[science])

) as up\_table

)

select id,

max(case when [subject]='english' then Marks end) as English,

max(case when [subject]='maths' then Marks end) as Maths,

max(case when [subject]='science' then Marks end) as Science

From cte

group by id

15)with cte as

(

SELECT \*,

DENSE\_RANK() OVER (partition by dept\_id order by salary desc) as rank

FROM EMPLOYEE

)

select \* from cte where rank=1

16)with cte as

(

SELECT \*,

DENSE\_RANK() OVER (partition by dept\_id order by salary desc) as rank

FROM EMPLOYEE

)

select \* from cte where rank<=2

order by dept\_id

17)with cte as

(

select \*,Teamid-1 as lag,Teamid+1 as lead from Team

)

select T.TeamID,T.TeamName,T1.TeamName as Lead,T2.TeamName as Lag

from Cte T left join Team T1 on T.lead=T1.TEAMID

left join Team T2 on T.lag=T2.TEAMID

18)create table dis (id int, [start] varchar(100), [end] varchar(100), dis int)

insert into dis values (1,'A','B',2000),(2,'D','C',2300),(3,'E','F',2100),(5,'B','A',2000),(6,'C','D',2300)

select distinct

case when [start]>[end] then start else [end] end as [start],

case when [start]<[end] then [start] else [end] end as [end],

dis from dis

with cte as

(

select id,

case when [start]>[end] then start else [end] end as [start],

case when [start]<[end] then [start] else [end] end as [end],

dis from dis

 ),cte2 as

 (

   select \*,row\_number() over (partition by [start],[end] order by dis) as rn from cte

 )

 select \* from cte2 where rn=1

19)create table status\_rep

(

TranId int,

Status varchar(10)

)

insert into status\_rep values(1,'P'),(2,'F'),(3,'F'),(4,'P'),(5,'P'),(10,'F'),(22,'F')

with cte as

(

select \*,lag(Status) over (order by TranId) as [lag] from status\_rep

)

select \* From cte

where lag='P' and status<>'P'

20)with cte as

(

select \*,row\_number() over (order by id desc) as r from part21

)

select r as id,name from

cte order by r

select c2.r as id,c2.name

from cte c1 join cte c2

on c1.id=c2.r

22)select id,val from Nullvalues\_Tbl

where val is not null

union

select DISTINCT id,val from Nullvalues\_Tbl

where val is null and id not in (select id from Nullvalues\_Tbl where val is not null)

order by id

23)declare @sql as varchar(max)

declare @col as varchar(100)

select @col=string\_agg(subject,',') from

(

select distinct(subject) as [subject] from pivot\_sample

) as d

set @sql = 'Select id,'+@col +' from pivot\_sample

pivot

(

  sum(marks) for subject in ('+@col+')

) as pv\_t'

 select @sql

execute (@sql)

declare @sql as varchar(max)

declare @col as varchar(100)

select @col=string\_agg(name,',')  from sys.columns where [object\_id]=object\_id('shaping') and name<>'id'

select @col

set @sql='select id,subjects,marks from shaping

unpivot

(

  marks for subjects in ('+@col+')

 ) as un\_pvt'

  select @sql

  execute (@sql)

25)Declare @st as varchar(100)='Thota Pavan Kumar'

Declare @len as int

Declare @st2 as varchar(100)=''

select @len=len(@st)

while(@len>0)

Begin

    declare @char as varchar(1)

    set @char=substring(@st,@len,1)

    set @st2=@st2+@char

    Set @len=@len-1

End

select @st2

26)create table emp\_up( id int, name varchar(100), adress varchar(100))

insert into emp\_up (id,name) values (1,'pavan'),(2,'sasi'),(3,'babavali'),(4,'murali')

create table address (id int, adress varchar(100), [date] date)

insert into address values

(1,'chirala','2022-02-03'),(1,'hyd','2023-03-01'),(1,'ght','2022-02-03'),

(2,'Gudivada','2024-02-03'),(2,'DLF','2022-03-03'),(3,'guntur','2023-02-02'),

(3,'gachibowli','2023-04-09')

with cte as

(

select id,max([date]) as [date] from address

group by id

),cte2 as

(

select c.id,adress from address a

join cte c on c.id=a.id and c.[date]=a.[date]

)

update emp\_up

set emp\_up.adress=c.adress

from cte2 c where emp\_up.id=c.id

select \* from emp\_up

28)with cte as

(

select S\_id,sum(charges) as charges from Aggregate\_Charges group by S\_id

),cte2 as

(

select S\_id,sum(fee) as fee

from Aggregate\_Fees

group by S\_ID

)

select isnull(c1.S\_ID,C2.S\_ID) as ID,isnull(charges,0) as charges,isnull(fee,0) as fee from cte c1 full join cte2 c2

on C1.s\_id=C2.S\_id

30)select \* from employee

where month([dob])=month([doj]) and day([dob])=day([doj])

with cte as

(

  select \* from employee

  where datediff(year,[doj],getdate())>=3

),cte2 as

(

select \*,datediff(year,[dob],getdate()) as [age]

from cte

)

select id,Name,Salary,Doj,Dob from

(

select \*,rank() over (order by age asc) as rk from cte2

) as d

where rk=1

31)create table con\_self( id int, product varchar(100))

insert into con\_self values (1,'A'),(1,'B'),(1,'D'),(1,'E'),(2,'A'),(2,'B'),(2,'C'),(3,'A'),(3,'B'),(3,'X'),(3,'Y')

with cte as

(

select a.id from

(

select id,count(\*) as no\_of\_products from con\_self group by id

) a

join

(

  select id,count(\*) as no\_of\_products\_witout\_c from con\_self where product not in ('C')

  group by id

)b

on a.id=b.id

where no\_of\_products=no\_of\_products\_witout\_c

)

select \* from con\_self

where id in (select id from cte)

32)create table com (id int identity(1,1), name varchar(100))

insert into com values ('pavan'),('kumar'),('thota'),

('pavan'),('kumar'),('thota'),('pavan'),('kumar'),('thota'),('pavan'),('kumar'),('thota'),

('pavan'),('kumar'),('thota')

declare @count as int

select @count=count(\*)/(select count(distinct(name)) from com) from com

select @count

select id,string\_agg(name,' ') as [Name] from

(

select id as id2,name,ntile(@count) over (order by id asc) as id from com

) as d

group by id

33)select id+lag(r,1,0) over (order by id) as id,name,gender from

(

select \*,row\_number() over (order by id) as r from part3 where gender='M'

) as d

union

select id+r-5 as id,name,gender from

(

select \*,row\_number() over (order by id) as r from part3 where gender='F'

) as m

order by id

select Id,Name,Gender from

(

select \*,row\_number() over (partition by gender order by gender) as pat

from part3

) as e order by pat,gender desc

34)with cte as

(

select E.\*,D.Dept\_head\_ID from Employee E

join Department D on E.dept\_ID=D.dept\_id

)

select C1.\*,c2.Emp\_name as manager\_name,c2.Salary as manager\_salary from cte c1

join cte c2

on c1.dept\_head\_id=c2.Id

where c1.Salary>=c2.Salary

35)Declare @val as decimal(18,9)=1.2511304;

with cte as

(

select cardnum,[value],abs(cast([value]-@val as decimal (18,9))) as val

from PART35\_CARDDETAILS

)

select cardnum,[Value] from

(

select \*,dense\_rank() over (order by val asc) as r

from cte

) as e

where r<=3

36)select \* from PART36

where isnumeric([value])=1

37)create table Missing\_Rows\_New( id int)

declare @start as int,@end as int

select @start=min(id),@end=max(id) from Missing\_Rows

while(@start<=@end)

begin

    if @start not in (select id from Missing\_Rows)

    BEGin

        insert into Missing\_Rows\_new values (@start)

    End

    set @start=@start+1

End

select \* from Missing\_Rows

SELECT \* from Missing\_Rows\_new

38)with cte AS

(

select \* , count([Value]) over (partition by [CARDNUM] order by [Date]) as cn from UPDATEMISSINGNUMBER\_P38

)

select CARDNUM,first\_value([Value]) over (partition by cn order by [date]) as [Value],[Date]

from cte

order by cardnum,[date]

39)select id,sum(Salary) as [Total]

from

(

select \* from PART39\_A

union all

select \* from PART39\_B

Union all

select \* from PART39\_C

) as D

Group by id

40)select sum(A+B) from OPTUM\_P1

select null+3 --return Null

update OPTUM\_P1 set a=10,b=10

where a is null or b is null

select \* from OPTUM\_P1

@@RowCount return number of row effected in this case 3 rows.

41)Declare @total as float

select @total=sum(Sales) from OPTUM\_P2;

with cte as

(

select Cities,Sales from OPTUM\_P2

Where Cities in ('Mumbai','Bangalore','Hyderabad')

union all

Select 'Other' as Cities,Sum(Sales) as sales from OPTUM\_P2

where Cities not in ('Mumbai','Bangalore','Hyderabad')

)

Select Cities,Round(cast(Sales as float)/@total,4)\*100 as [sales\_percent]

from cte

order by [sales\_percent] desc

42)

a)select E.\*,D.DEPT\_NAME from employee E

left join DEPARTMENT D

on D.DEPT\_ID=E.DEPT\_ID

where d.dept\_id is null

b)create table student (id int identity, name varchar(100))

insert into student values('pavan'),('Babavali'),('Sasi'),('murali')

create table phone( id int, number int, [type] varchar(100))

insert into phone values (1,90183837,'M'),(1,22342,'L'),(2,919393,'L'),(3,920203,'M'),(4,92292,'L')

with cte as

(

select s.\*,p.[type],p.number from student s

join phone p on s.id=p.id

)

select id,name,number,type from

(

select \*,row\_number() over (partition by id order by [type] desc ) as r

from cte

) as d where r=1

43)with cte as

(

  select col1,row\_number() over (order by (select 1)) as r from COLVALUES

)

select C2.\* from cte c1

join

(

select col1,col2 from COLVALUES

union all

SELECT col1,col3 as col2 from COLVALUES

) c2

on c1.col1=c2.col1

44)SELECT CHARINDEX('S','RAMSINGH')

--inex tarts from 1 so we get 4 because s at 4th position

SELECT CASE WHEN NULL=NULL THEN 1 ELSE 0 END

-- both nulls are not same 0

SELECT NULL UNION ALL SELECT 'NULL' UNION ALL SELECT NULL

--it consider all nulls are different retuwn 3 nulls in second column it is string firat and last are null

SELECT SUBSTRING('123456789',CHARINDEX('XYZABC','Z'),4)

-- in charindex we woth find character because we need to find character to match first and next string here it is reversea

--so we get 0 now 0 to 4 means 0,1,2,3 no 0 in sql so we get values at 1,2,3 indesx which are 123

SELECT LEFT('1234321',3)  AS LEFT3,RIGHT('1234321',3) AS RIGHT3

--123 for left and 321 for right

SELECT STUFF('FIS Global Pvt Ltd.',12,3,'INDIA')

--'FIS Global INDIA Pvt Ltd'

SELECT REPLICATE('XYZ',3)

--'XYZXYZXYZ'

SELECT REVERSE(1234321)

--OUTPUT--'1234321'

--we use it for integer also it reverse the integer and string

SELECT REPLACE('ABCXYZ','ABC','XYZ')

--'XYZXYZ'

SELECT CHARINDEX('Ind','FIS Global INDIA Pvt Ltd. India',15)

--27 because we are provide start at 15 ohter wise first matching it return 12

SELECT PATINDEX('%Ind%','FIS Global INDIA Pvt Ltd. India')

--12 first matching pattern in the given string.

45)create table splits( id int identity,Chars varchar(1))

Declare @name as varchar(100)='Thota Pavan Kumar'

declare @count as int

select @count=len(@name)

declare @val int=1

while(@count>0)

Begin

    declare @char as char(1)=substring(@name,@val,1)

    insert into splits values (@char)

    set @val=@val+1

    set @count=@count-1

End

select \* from splits

46)

a)declare @a as float,@b as float,@c as float,@d as float

select @a=min(lat\_n),@b=min(long\_w),@c=max(lat\_n),@d=max(long\_w)

from STATION

select round((@c-@a)+(@d-@b),4) as [distance]

b)declare @a as float,@b as float,@c as float,@d as float

select @a=min(lat\_n),@b=min(long\_w),@c=max(lat\_n),@d=max(long\_w)

from STATION

select round(sqrt(square(@c-@a)+square(@d-@b)),4) as [distance]

47)with cte as

(

select \*,

row\_number() over (order by lat\_n desc) as r1,

row\_number() over (order by lat\_n asc) as r2

from STATION

)

select round(avg(lat\_n),2) as [median]

from cte

where r2 in (r1,r1+1,r1-1)

48)with cte1 as

(

select top 1 city,len(city) as [length]

from STATION order by length asc, [city]

),cte2 as

(

select top 1 city,len(city) as [length]

from STATION order by length desc, [city]

)

select \* from cte1

union

select \* from cte2

49)

a)declare @input varchar(100)='abc,,,dddd,,,,,,dde,,2lmdl3emf,cewec,,,,,'

select replace(replace(replace(@input,',','\*,'),',\*',''),'\*','')

b)create table s( id int, name varchar(100))

insert into s values (1,'pavan'),(2,'sasi'),(3,'babavali')

create table d( id int, name varchar(100))

merge d as d

using s as s

on s.id=d.id

when matched then

update set d.id=s.id,d.name=s.name

when not matched by target then

insert values(s.id,s.name);

c)declare @val as varchar(100)='Thota Pavan Kumar'

select len(@val)-len(replace(@val,'a','')) as [occurance\_of\_a]

d)DECLARE @ATT TABLE (EMP INT,DATEPRESENT VARCHAR(50))

INSERT @ATT VALUES (1,'1,2,3'),(2,'2,4,5,6'),(3,'1,2,5,4,7,9')

select EMP,count(\*) as [present] from

(

select EMP,Value  from @att cross apply string\_split(DatePresent,',')

) as f

group by EMP

e)update d set id=name,name=id

alter table d alter column id varchar(100)

select \* from d

f)with cte as

(

  select 1 as n

  union all

  select n+1 as n from cte where n+1<=100

)

select n from cte

50)with cte as

(

select \*,cast(overs as int)\*6+right(overs,1) as [ov] from CRICKET\_BOWLING\_STATS

)

,cte2 as

(

select playername,count(\*) as Matches\_Played, sum([Ov]) as [Overs\_Bowled],

sum(run) as runs\_scored,sum(Wickets) as Wickets

from cte

group by playername

)

select playername,Matches\_played,cast(overs\_bowled/6 as varchar(100))+'.'+

cast(Overs\_bowled%6 as varchar(100)) as [overs\_bowled]

,runs\_scored,wickets  from cte2

51)with cte as

(

select \*,count(id) over (order by seq) as r from prev\_null\_val

)

select Seq,first\_value(ID) over (partition by r order by seq) as ID,

Name from cte

53)with cte as

(

  select id,name, 1 as n from PART53\_REPEATE\_VAL

  union all

  select c.id,c.Name,n+1 as n from cte c

  join PART53\_REPEATE\_VAL r

  on r.id=c.id where r.OCCURRENCE>=n

 )

 select id,name from cte order by name

54)declare @val as int

select @val=max(len(ip)) from PART54\_FIXEDFORMAT

select right(replicate(0,@val)+ip,@val)

from PART54\_FIXEDFORMAT

55)create table prime\_number( num int)

Declare @value int=3

while @value<=50

begin

    declare @count as int=0

    declare @val as int=2

    while @val<@value

    begin

        if @value%@val=0

        begin

            set @count=@count+1

        End

        set @val=@val+1

    End

    if @count=0

    begin

        insert into prime\_number values (@value)

    End

    set @value=@value+1

 End

select 2 as num

union all

select \* from prime\_number

56)

a)with cte as

(

  select cast(replicate('\*',1)as nvarchar(100)) as r,1 as n

  union all

  select cast(replicate('\*',n+1) as nvarchar(100)) as r,n+1 as n from cte where n+1<=10

 )

 select r as pattren from cte

b)with cte as

(

  select cast(replicate('\*',10)as nvarchar(100)) as r,10 as n

  union all

  select cast(replicate('\*',n-1) as nvarchar(100)) as r,n-1 as n from cte where n-1>=1

 )

 select r as pattren from cte

c)with cte as

(

  select cast(replicate(' ',10)+replicate('\*',1)as nvarchar(100)) as r,1 as n

  union all

  select cast(replicate(' ',10-n)+replicate('\*',n+1) as nvarchar(100)) as r,n+1 as n from cte where n+1<=10

 )

 select r as pattren from cte

d)with cte as

(

  select cast(replicate(' ',0)+replicate('\*',10)as nvarchar(100)) as r,1 as n

  union all

  select cast(replicate(' ',n)+replicate('\*',10-n) as nvarchar(100)) as r,n+1 as n from cte where n+1<=10

 )

 select r as pattren from cte

57)Declare @val as int=153

declare @len as int=1

declare @b as int

select @b=len(@val)

declare @total as int=0

while @b>0

begin

    declare @a as int=substring(cast(@val as varchar(100)),@len,1)

    set @len=@len+1

    set @total=@total+(@a\*@a\*@a)

    set @b=@b-1

End

select iif(@val=@total,'Armstrong','Not a Armstrong')

58)create table val( characters char(1))

Declare @name as varchar(100)

select @name=string\_agg(name,'') from STRING\_IQ\_TBL

declare @len as int

select @len=len(@name)

declare @val as int=1

while(@len>0)

begin

    declare @char as char(1)=substring(@name,@val,1)

    insert into val values (@char)

    set @val=@val+1

    set @len=@len-1

 End

 select characters,count(\*) as [occurance] from val

 group by characters

59)

a)with cte as

(

  select 1 as id

  union all

  select id+2 as id from cte where id+2<=99

)

select \* from cte

60)create table val(id int)

insert into val values (1),(2),(3),(4),(5),(6),(7),(8),(9),(10)

select id,id+lag(id,1,0) over (order by id)+lead(id,1,0) over (order by id) as [new\_val]

from val

--WE CANNOT USED A WINDOW FUNCTION IN OTHER WINDOW FUNCTION

63)alter PROC [Book\_Price] (@start\_date date, @End\_date date,@Price float)

AS

Begin

    declare @days as int

    declare @cost as float

    select @days=datediff(day,@start\_date,@end\_date)+1

    if @days>=60

    begin

        set @cost=(30\*0.5)+(30\*0.75)+(@days-60)\*1.5

    End

    else if @days>=30

    BEGIN

        set @cost=(30\*0.5)+(@days-30)\*0.75

    End

    Else

    Begin

        set @cost=@days\*0.5

     End

     Select iif(@cost>@price,@price,@cost) as [total\_price]

End

Exec [Book\_price] '2022-03-02','2022-05-02',50

65)

a)DECLARE @T1 TABLE(ID INT)

INSERT @T1 VALUES (1),(2),(3)

DECLARE @T2 TABLE(ID VARCHAR(1))

INSERT @T2 VALUES ('A'),('B'),('C');

with cte as

(

select \*,row\_number() over (order by id asc) as r

from @t2

),cte2 as

(

select c.\* from cte c join @t1 t

on c.r=t.id

)

select cast(r as varchar(100))+replicate(id,r) from cte2

b)Declare @table as table(EmpId int,Branch Varchar(20))

Insert @table

Select 1,'INDIA'

Union

Select 2,'US'

Union

Select 3,'US'

Union

Select 4,'INDIA'

Union

Select 5,'INDIA'

Union

Select 6,'SINGAPORE'

Union

Select 7,'SINGAPORE'

declare @input as int=1;

with cte as

(

select branch from @table where Empid=@input

)

select \* from @table where branch=(select branch from cte) and empid<>@input

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Tech TFQ Top 10**

1)with cte as

(

    select \*,row\_number() over (partition by model\_name,color,brand order by model\_id) as r

    from cars

)

select \* from cte where r<=1

order by model\_id

2)WITH CTE AS

(

select DEPT\_ID,max(salary) as [max\_salary],

min(salary) as [min\_SALARY] from EMPLOYEE

GROUP BY DEPT\_ID

)

SELECT e.\* FROM EMPLOYEE e

JOIN CTE C ON c.dept\_id=e.dept\_id AND

e.salary=c.[max\_salary] or e.salary=c.[min\_salary]

order by e.dept\_id,e.salary

3)select \*,abs(lag(cumulative\_distance,1,0) over (partition by cars order by days)-cumulative\_distance)

as [distance]

from car\_travels

4)select distinct \* from

(

select

case when source>destination then source else destination end as source\_new,

case when source<destination then source else destination end as destination\_new,

 distance

from src\_dest\_distance

) as e

5)with cte as

(

  select id,item\_name,total\_count from travel\_items

  union all

  select c.id,c.item\_name,c.total\_count-1 from cte c

  join travel\_items t on c.id=t.id

  where c.total\_count>1

)

select id,item\_name from cte

order by id

6)

--twice

select t1.team\_code,t1.team\_name as team1,t2.team\_code,

t2.team\_name as team2 from

teams t1 cross join teams t2

where t1.team\_name<>t2.team\_name

--single

with cte as

(

  select \*,row\_number() over (order by team\_code) as r

  from teams

)

select \* from cte

select c1.team\_code,c1.team\_name as team1,

c2.team\_code,c2.team\_name as team2

from

cte c1 join cte c2 on

c1.r<c2.r

7)with cte as

(

select \*,cast(replace(amount,'$','') as int) as [amount1],

left(datename(month,sales\_date),3)+'-'+right(cast(year(sales\_date) as varchar(100)),2) as [year\_month]

from sales\_data

),cte2 as

(

select customer\_id,isnull([jan-21],0) as [jan-21],isnull([feb-21],0) as [feb-21],isnull([mar-21],0) as [mar-21],

isnull([apr-21],0) as [apr-21],isnull([may-21],0) as [may-21],isnull([jun-21],0) as [jun-21],isnull([jul-21],0) as [jul-21],

isnull([aug-21],0) as [aug-21],isnull([sep-21],0) as [sep-21],isnull([oct-21],0) as [oct-21],

isnull([nov-21],0) as [nov-21],isnull([dec-21],0) as [dec-21] from

(

select year\_month,customer\_id,sum(amount1) as [amount]

from cte

group by year\_month,customer\_id

) as d

pivot

(

  sum(amount)  for year\_month in ([jan-21],[feb-21],[mar-21],[apr-21],[may-21],[jun-21],[jul-21],

[aug-21],[sep-21],[oct-21],[nov-21],[dec-21])

 ) as pivot\_table

)

select \* from cte2

union all

select 'total' as customer\_id,

sum([jan-21]) as [jan-21], sum([feb-21]) as [feb-21],

sum([mar-21]) as [mar-21],sum([apr-21]) as [apr-21],

sum([may-21]) as [may-21], sum([jun-21]) as [jun-21],

sum([jul-21]) as [jul-21],sum([aug-21]) as [aug-21],

sum([sep-21]) as [sep-21], sum([oct-21]) as [oct-21],

sum([nov-21]) as [nov-21], sum([dec-21]) as [dec-21]

from cte2

8)declare @input as int=7;

with cte as

(

select \* from emp\_details where id=@input

union all

select e.\* from cte c join emp\_details e

 on c.id=e.manager\_id

)

select \* from cte

select \* from emp\_details

9)with cte as

(

  select join\_date as [date],salary as sales   from EMPLOYEE E

  join JOIN\_DATE J on E.id=J.ID

),cte2 as

(

select year([date]) as [year], datename(month,[date]) as [month\_name],sales

from cte

) ,cte3 as

(

select [year],[month\_name],sum(sales) as [sales] from

cte2

group by [year],[month\_name]

)

select c1.[year],c1.[month\_name],c1.sales,c2.Sales,abs(c1.sales-c2.sales) as diff\_sales

from cte3 c1

left join cte3 c2

on c1.year-1=c2.year and c1.month\_name=c2.month\_name

order by [year]

10)with cte as

(

select cust\_name,

count(\*) as total\_orders,

count(case when status='DELIVERED' then 1 else null end) as Delivered,

count(case when status='SUBMITTED' then 1 else null end) as Submitted,

count(case when status='CREATED' then 1 else null end) as Created

from cust\_orders

group by cust\_name

)

select cust\_name,

case when total\_orders=Delivered then 'COMPLETED'

     when delivered>=1 then 'In Progress'

     when total\_orders=Submitted then 'Awating Progress'

     else 'Awating Submission' end as Final\_status

 from cte

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Random Questions**

1)Declare @name as varchar(100)='Pavan Kumar Thota'

select iif(charindex(' ',trim(@name))=0,@name,substring(trim(@name),1,charindex(' ',trim(@name))-1)) as First\_name,

iif(charindex(' ',@name)=0,null,reverse(substring(reverse(@name),1,charindex(' ',@name)-1))) as Last\_name

,iif(charindex(' ',trim(@name))=charindex(' ',reverse(trim(@name))),null,

     substring(trim(@name),charindex(' ',trim(@name))+1,charindex(' ',reverse(trim(@name)))-1)) as Middle\_Name

2)Performance tunning

used columns instead of \* by column name it process quickly.

avoid the use of distinct after getting result it need to filter the data again to get distinct records.

avoid correlated sub queries sub query table must execute for each row of main query table.

filter data in where clause instead of having because filter data in early stages improves performance.

avoid cursor as much as possible because they are very slow.

avoid using while loop takes more time

avoid use of more joins it slows the query

do inner joins rather than where it creates Cartesian product of tables.

Avoid using dynamic SQL because it causes SQL injections.

3)

a)select \* from EMPLOYEE

where salary=(select min(salary) as salary

from EMPLOYEE)

b)with cte as

(

  select \*,dense\_rank() over (order by salary desc) as [rank]

  from EMPLOYEE

)

select \* from cte where rank=2

c)select \* from employee

where emp\_name like 'a%'

d)select \* from employee

where salary in (11000,21000,31000)

select \* into emp\_dup from employee --structare and data

select \* into emp\_dup2 from EMPLOYEE where 1=2 --structure

select \* from emp\_dup2

e)select \* from employee

where salary between 25000 and 38000

f)delete from employee

where id in (1,2,3,4,5)

4)create table table1( id int)

create table table2(id int)

insert into table1 values(1),(2),(3),(3),(4),(5)

insert into table2 values(2),(3),(5),(8),(null)

select \* from table1 t1 join table2 t2 on t1.id=t2.id

select \* from table1 t1 left join table2 t2 on t1.id=t2.id

select \* from table1 t1 right join table2 t2 on t1.id=t2.id

select \* from table1 t1 full join table2 t2 on t1.id=t2.id

--inner join (2,2),(3,3),(3,3),(5,5)

--left join (1,null),(2,2),(3,3),(3,3),(4,null),(5,5)

--right join (2,2),(3,3),(3,3),(5,5),(null,8),(null,null)

--full join (1,null),(2,2),(3,3),(3,3),(4,null),(5,5),(null,8),(null,null)

5)create table add\_new( id int, name varchar(100), address varchar(100), [date] date)

insert into add\_new values

(1,'aa','addr1','2022-03-04'),(1,'aa','addr1','2022-02-03'),

(2,'bb','addr2','2023-02-03'),(2,'bb','addr2','2022-01-01'),

(3,'cc','addr3','2022-02-03'),(3,'cc','addr3','2023-02-03'),

(3,'cc','addr3','2022-07-03');

with cte as

(

  select \*,row\_number() over (partition by id,name,address order by [date] asc) as rn

  from add\_new

)

select \* from cte where rn=1

6)create table gender (area varchar(100),gender varchar(100))

insert into gender values('HYD','M'),('HYD','M'),('HYD','F'),('HYD','M'),('HYD','F'),('HYD','M'),

('HYD','M'),('HYD','F'),('CHN','M'),('CHN','M'),('CHN','M'),('CHN','M'),('CHN','M'),('CHN','M'),

('CHN','M'),('CHN','M'),('DHL','F'),('DHL','M'),('DHL','F'),('DHL','F'),('DHL','F'),('DHL','M'),('DHL','F') ,

('DHL','F'),('DHL','M'),('DHL','F'),('DHL','M')

select area,count(\*) as total\_count,

isnull(count(case when gender='M' then 1 else null end),0) as Male\_count,

isnull(count(case when gender='F' then 1 else null end),0) as Female\_count

from gender

group by area

with cte as

(

select area,[M],[F] from gender

pivot

(

  count(gender) for gender in ([M],[F])

) as PV\_Table

)

select area,[M]+[F] as [total\_Count],[M] as Male\_count,[F] as Female\_count

From cte

**Gate Smashers Top 15**

1)What is DBMS

DBMS means Database Management system

DBMS is a software which is used to create, define, and maintain the database and provide the controlled access to the data.

Types of DBMS

Relational Database

Non-Relational Database

Hierarchy Database

Object oriented Database etc.

2)What is Primary key

Primary key is a column or group of columns which are used to uniquely identify records in the table.

Primary key must be one in table and does not allow nulls and duplicates.

3)what is foreign key

Foreign key is a column or a group of columns which is primary key in other table. By using this primary and foreign key we establish the relation between the table.

Foreign key may have duplicates and null.

By using referential integrity constraint, we can maintain integrity of data.

4)Constraints and types of constraint

Constraints are rules specified on the data we entering table.

Null it allows nulls,

Not null it does not allows nulls.

Check it allows values in a specific or specified limit or range.

Default it assign default value if no value specified for column

Unique it does not allow duplicates and allows one null.

Primary key it does not allow duplicates and nulls.

foreign key it maintain the integrity of data by avoiding the delete of rows if it referred to other table as primary key.

6)Difference between drop delete truncate.

Drop removes table structure from the database.

Delete used to remove the records from the table by using where clause we can remove the specific records. Delete is a DML command and it can be rolled back and identity does not reset to 0.

Truncate used to remove all records from the table we cannot use where clause it is DDL command it does not be rolled back. It resets identity to 0.

7)Group by and Order by

Group by used to group the data at specific column level. Group by used to aggregate the data.

Order by clause is last clause in SQL statement it used to order the data in asc or desc by a one or more columns which are separated comma.

8)Types of joins

Joins are used to join two tables based on a single or multiple columns (use logical operators).

Types of joins

Inner join return matching records from both the tables.

Left join return matching records from the right table and all records from the left table.

Right join return matching records from left table and all records from right table.

Full join return both matching and non-matching records from both tables.

Cross join it return the Cartesian product to tables i.e. all possible combinations.

9)Sub Query

Sub query means query in a query which enclosed in parentheses ().

There are two types of sub queries

Correlated and non-correlated sub query

Correlated sub query means inner queries and outer queries both depends upon each other. They won’t execute individually. Correlated sub query mostly written in from clause.

select \* from EMPLOYEE

where salary=(select max(salary) From Employee)

Non correlated means both queries can execute individually. Mostly we use non correlated sub query in from, where and having clause.

select \*,(select dept\_name from DEPARTMENT D where D.dept\_id=E.dept\_id)

from EMPLOYEE E

10)Pattern matching

To do pattern matching we use like in where clause

\_ for single character

%- for any number of characters

[a,d,c]-multiple characters

[a-z]- range of characters

^ - not operator.

11)Find second highest salary

select \* from Employee

where salary=

(

select max(salary) from Employee

where salary not in (select max(salary) from EMPLOYEE)

)

12)What is SQL and who are RBDMS providers.

SQL means structured query language it used to interact with the relational database.

SQL Server, Oracle, MYSQL, MarinaDB, PostgreSQL are some RDBMS providers.

13)varchar vs varchar2

Varchar is ANSI standard where varchar2 is oracle standard.

14)Triggers

Triggers are special type of stored procedures which automatically executes whenever and event occurs in the database.

There are 3 types of triggers

DML in DML we have instead of and after triggers in each we have insert, update, delete. Instead of means instead of action it executes triggers in after triggers after performing action it executes trigger.

DDL triggers it trigger for Create\_table, Drop\_Table, rename, Alter\_table in database level and server level based on scope we specified.

15)ACID properties

ACID properties

Atomicity every command in the transaction must execute successfully otherwise rollback all.

Consistency data touched by transaction must be consistent before and after transaction.

Isolation each transaction must be isolated with each other.

Durability mean any changes performed by the transaction must be permanent even at power failure.

**KSRDataVision**

1)what is DDL and DML and difference between them.

DDL means Data defining language and DML means Data Manipulation Language.

DDL consists of Create, Drop, Truncate, Alter.

DML consists of Delete, update, Insert.

Main difference between DDL and DML is DDL cannot be rolled back and used to work on structure of tables. Where DML can be rolled back and perform operation on data of table.

2)Difference between drop, delete, truncate.

Drop used to drop the table structure.

Delete used to delete records from the table we can use where clause to delete specific records. Delete can be rolled back and it does not reset the identity.

Truncate used to remove all records from the table we cannot use where clause here. It is DDL we cannot rolled back. It resets identity to 0.

3)Group by

Group by used when we need to aggregate the data based on columns we specified in group by.

4)Difference between where and having

Where clause used before group by and we cannot use aggregated data in where and group by is not mandatory. Where used to filter the data before data getting aggregated.

Having clause used after group by and we can use columns with aggregate functions. Group by clause is mandatory and it used to filter data after data getting aggregated.

5)difference between scalar and aggregate functions

Scalar function return output for each row eg len, upper, lower etc.

Aggregate function are the function which return single numeric value by grouping the several or multiple values like sum, min, max etc.

6)coalesce function

Coalesce function used to replace null value.

It return first not null value in the specified list of values.

7)What are the constraints and it types.

Constraints are used to specify the rules to the data.

There are different types of constraints

Null it allows nulls.

Not null it does not allow nulls.

Check it used to check the value is it in specified range.

Unique it does not allow duplicates but allow one null.

Default it assign default value to the column for it value is not specified.

Primary key it does not allow null and duplicates.

Foreign key it maintain integrity of data by avoiding delete of records if it refered to other table.

8)How many primary and unique keys per table.

We can have only one primary key per table.

We can have one or more unique keys per table based on requirement.

9)what is join and types of joins.

Joins are used to join the two tables based on columns.

Types of joins

Inner join matching records from both tables.

Left join it return matching record from right table and all records from the left table.

Right join it return matching record from left table and all records from right table.

Full join it return both matching and non-matching records from both tables.

Cross join it return cartesian product of two tables i.e. all possible combination.

10)difference between union and union all.

Union return the distinct records in output i.e. avoids duplicates.

Union all return duplicates in output.

11)Order of execution

From

Where

Group By

Having

Select

Order by

Offset/Fetch Next

12)How many joins used to join 8 tables.

No\_Of\_Joins=No\_Of\_Tables-1

So for joining 8 tables we need 7 joins.

13)Windows Function

Window functions are the functions which perform operation based on set of rows of result set this set of rows nothing but a window. We can define set or rows or window in over clause by partition by, order by, row and range.

Eg:- lag, lead, rank, dense rank etc.

14)What is difference between rank and Dense\_Rank.

Rank is a window function which used to return a column with ranks based on a column. If two records get same rank next rank will be skkiped.

Dense\_Rank is Window function which used to return a column with ranks based on a column. It two records get same rank next ranks wont skipped.

15)what is view how it difference from table.

View are virtual table it does not consist of actual data it just consist of query to retrieve the data. By using views, we can update underlying table. We can create view on multiple tables.

Tables stored actual data.

16)what is index.

Index used to speed up the query.

Indexes are two types clustered and non-clustered index.

Cluster index are defines the physical order of the table. They stored with in the table and are fast compared to non-clustered index. Leaf node of clusterd index consists of actual data.

Primary key means unique clustered index. Only one clustered index allowed for table.

Non-Clustered index does not defines physical order of the table. They stored separately and are slow compared to clustered index. Leaf node points to actual data. We can have mutiple non clusterd index up to 999 per table.

We can use unique or both clustered and non-clustered index.

17)what is cte. How it used.

CTE means comman table expression.

It used to store the temporary result set.

We can use it right behind it created. We cannot access it anywhere.

18)What is SQL vs PLSQL vs T-SQL

SQL means structured query language it used to interact with relational databases.

SQL is ANSI standard.

PLSQL is developed by oracle it consist of SQL and some more additional functionalities created by oracle used in oracle database.

PL SQL means Procedural SQL

T-SQL is developed by Microsoft it consist of SQL and some more additional functionalities created by Microsoft used in SQL Server.

T SQL -Transact SQL.

19)Cross join

One o the use case of cross join is we can create IPL match fixtures using Cross join.

20)Top 3 Salary

with cte as

(

  select \*,dense\_rank() over (order by salary desc) as r

  from EMPLOYEE

)

select id,emp\_name,Gender,salary,dept\_id,location\_id from cte where r=3

21)remove duplicates

with cte as

(

    select \*,row\_number() over (partition by id order by id) as r from EMPLOYEE

)

delete from cte where r=2

22)top 5 salary departments

select dept\_id,sum(salary) as sum\_sal

from EMPLOYEE

group by dept\_id

order by sum\_sal desc

offset 0 rows

fetch next 5 rows only

23)Display alternative rows

with cte as

(

  select \*,row\_number() over (order by id) as r

  from EMPLOYEE

)

select id,emp\_name,Gender,salary,dept\_id,location\_id

from cte where r%2=1

24)Employees joined in this month

  with cte as

(

    select e.\*,j.join\_date from employee e

    join join\_date j

    on e.id=j.id

)

select \* from cte

where year(getdate())=year(join\_date)

and month(getdate())=month(join\_date)

25)Employee and Manager

with cte as

(

  select e.\*,d.Dept\_Head\_Id from EMPLOYEE e

  join DEPARTMENT d on e.dept\_id=d.dept\_id

)

select c1.emp\_name as Employee\_name,isnull(c2.emp\_name,'Boss') as Manager\_Name

from cte c1 left join cte c2 on c1.dept\_head\_id=c2.id

**Rishab Mishra**

Order of execution

[From] [Where] [Having] [Group by] [Select] [Order by] [Offset/Fetch Next]

2)Select Year(Sale\_Date) as [Year],

Month(Sale\_Date) as [Month],

Sum(Sales) as [Sum\_Sales]

From Products

Group by Year(Sale\_Date),

Month(Sale\_Date)

3)Select Candidate\_Id,Count(\*) as [Count]

From Aplications

Where skills in ('Power BI','SQL Server','SSIS')

Group by Candidate\_Id

Having Count(\*)=3