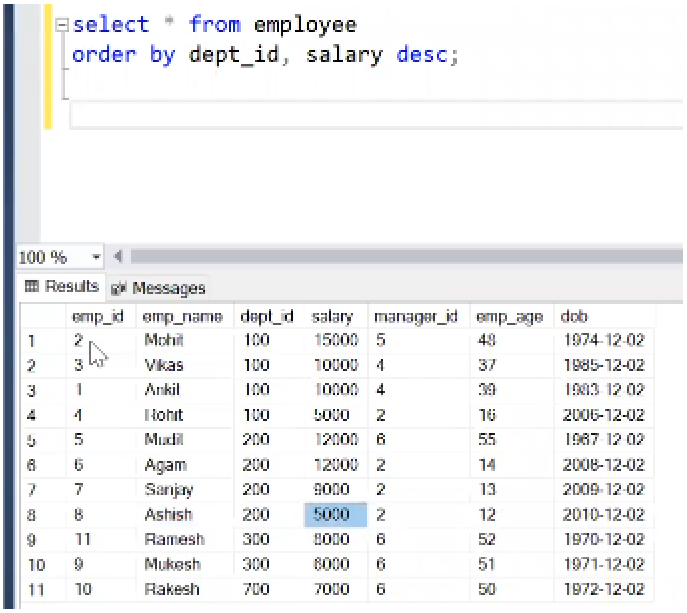
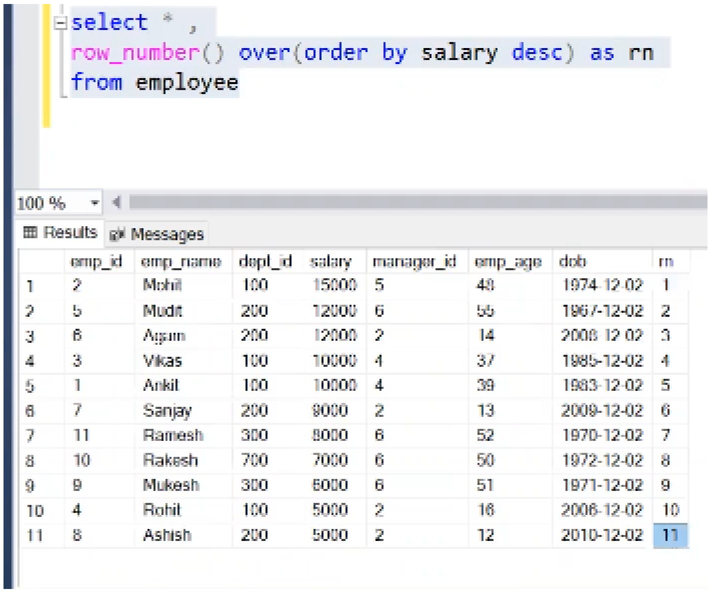
WAF

1. 

```sql
select * from employee
order by dept_id, salary desc;
```

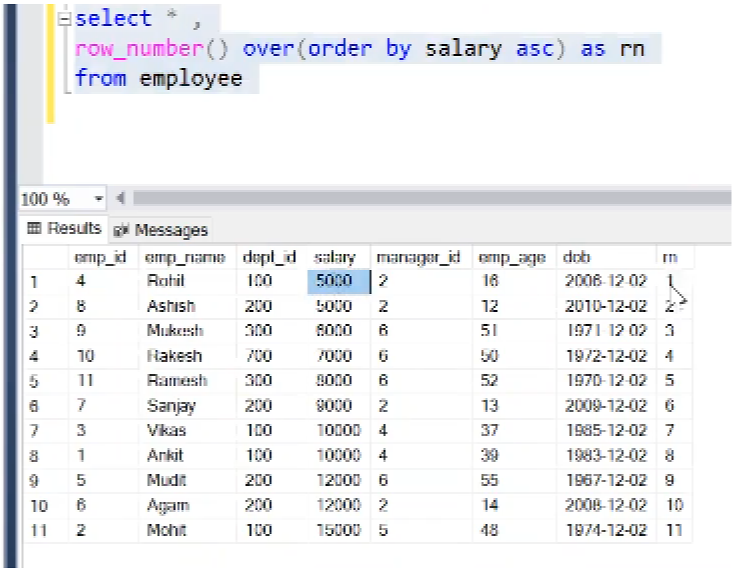
	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob
1	2	Mohit	100	15000	5	48	1974-12-02
2	3	Vikas	100	10000	4	37	1985-12-02
3	1	Ankit	100	10000	4	39	1983-12-02
4	4	Rohit	100	5000	2	16	2006-12-02
5	5	Mudit	200	12000	6	55	1967-12-02
6	6	Agam	200	12000	2	14	2008-12-02
7	7	Sanjay	200	9000	2	13	2009-12-02
8	8	Ashish	200	5000	2	12	2010-12-02
9	11	Ramesh	300	8000	6	52	1970-12-02
10	9	Mukesh	300	6000	6	51	1971-12-02
11	10	Rakesh	700	7000	6	50	1972-12-02

2. Here we have..employee table..with highest salaries in each dept
3. We can retrieve highest salary in each dept…but can we retrieve top 2 highest salaries in each dept?
4. With what we have learned so far…we cannot solve this
5. We have to use window function
6. 

```sql
select * ,
row_number() over(order by salary desc) as rn
from employee
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn
1	2	Mohit	100	15000	5	48	1974-12-02	1
2	5	Mudit	200	12000	6	55	1967-12-02	2
3	6	Agam	200	12000	2	14	2008-12-02	3
4	3	Vikas	100	10000	4	37	1985-12-02	4
5	1	Ankit	100	10000	4	39	1983-12-02	5
6	7	Sanjay	200	9000	2	13	2009-12-02	6
7	11	Ramesh	300	8000	6	52	1970-12-02	7
8	10	Rakesh	700	7000	6	50	1972-12-02	8
9	9	Mukesh	300	6000	6	51	1971-12-02	9
10	4	Rohit	100	5000	2	16	2006-12-02	10
11	8	Ashish	200	5000	2	12	2010-12-02	11

7. Here we have generated row_number for salaries in desc order ..with the help of over()
8. We can see the result…highest salary has rn = 1,2nd highest has rn=2..similarly for all salaries

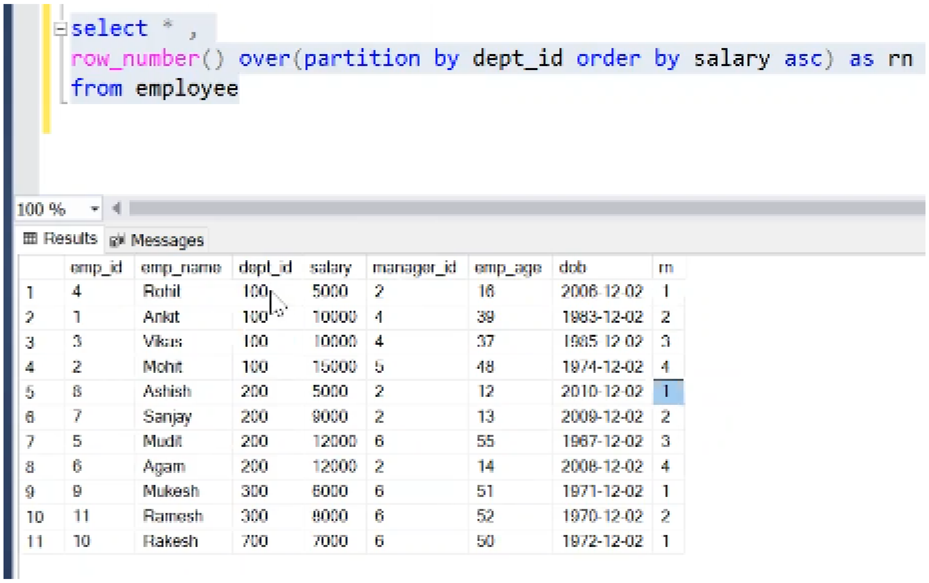


```
select * ,
row_number() over(order by salary asc) as rn
from employee
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn
1	4	Rohit	100	5000	2	16	2006-12-02	1
2	8	Ashish	200	5000	2	12	2010-12-02	2
3	9	Mukesh	300	6000	6	51	1971-12-02	3
4	10	Rakesh	700	7000	6	50	1972-12-02	4
5	11	Ramesh	300	8000	6	52	1970-12-02	5
6	7	Sanjay	200	9000	2	13	2009-12-02	6
7	3	Vikas	100	10000	4	37	1985-12-02	7
8	1	Ankit	100	10000	4	39	1983-12-02	8
9	5	Mudit	200	12000	6	55	1967-12-02	9
10	6	Agam	200	12000	2	14	2008-12-02	10
11	2	Mohit	100	15000	5	48	1974-12-02	11

9. row_number function

10. The same result..with order by ascending salaries

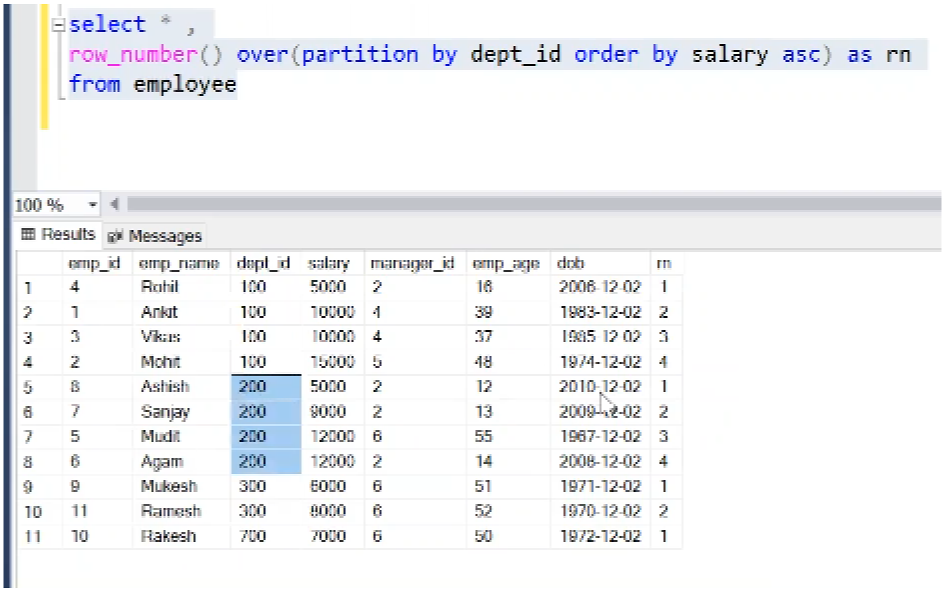


```
select * ,
row_number() over(partition by dept_id order by salary asc) as rn
from employee
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn
1	4	Rohit	100	5000	2	16	2006-12-02	1
2	1	Ankit	100	10000	4	39	1983-12-02	2
3	3	Vikas	100	10000	4	37	1985-12-02	3
4	2	Mohit	100	15000	5	48	1974-12-02	4
5	8	Ashish	200	5000	2	12	2010-12-02	1
6	7	Sanjay	200	9000	2	13	2009-12-02	2
7	5	Mudit	200	12000	6	55	1967-12-02	3
8	6	Agam	200	12000	2	14	2008-12-02	4
9	9	Mukesh	300	6000	6	51	1971-12-02	1
10	11	Ramesh	300	8000	6	52	1970-12-02	2
11	10	Rakesh	700	7000	6	50	1972-12-02	1

11.

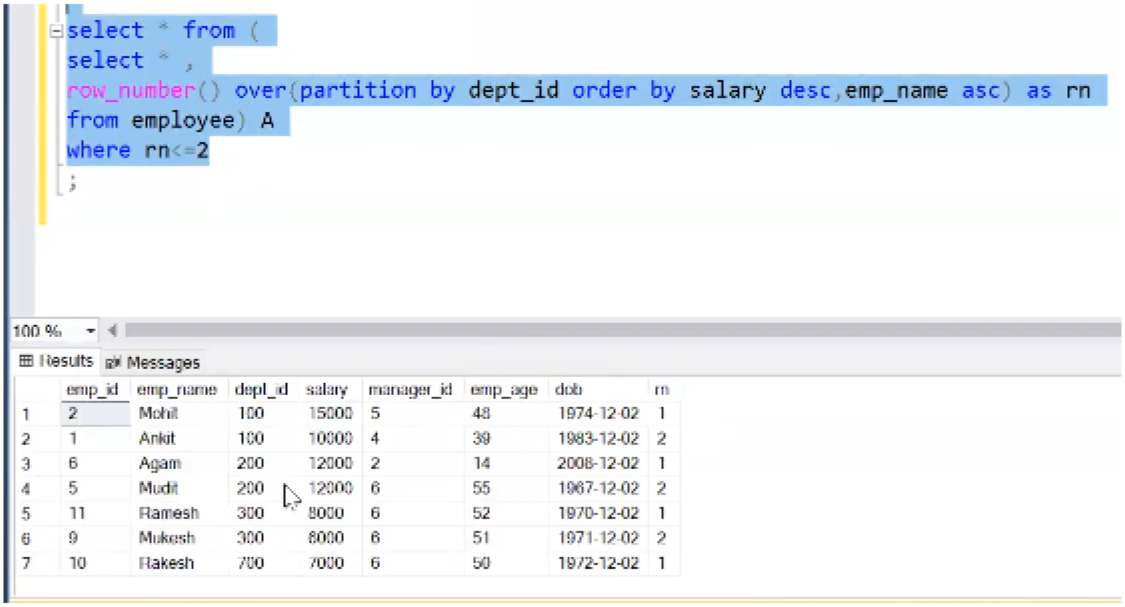
12. Partition

13. 

```
select * ,
row_number() over(partition by dept_id order by salary asc) as rn
from employee
```

100 %

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn
1	4	Rohit	100	5000	2	16	2006-12-02	1
2	1	Ankit	100	10000	4	39	1983-12-02	2
3	3	Vikas	100	10000	4	37	1985-12-02	3
4	2	Mohit	100	15000	5	48	1974-12-02	4
5	8	Ashish	200	5000	2	12	2010-12-02	1
6	7	Sanjay	200	9000	2	13	2009-12-02	2
7	5	Mudit	200	12000	6	55	1967-12-02	3
8	6	Agam	200	12000	2	14	2008-12-02	4
9	9	Mukesh	300	6000	6	51	1971-12-02	1
10	11	Ramesh	300	8000	6	52	1970-12-02	2
11	10	Rakesh	700	7000	6	50	1972-12-02	1

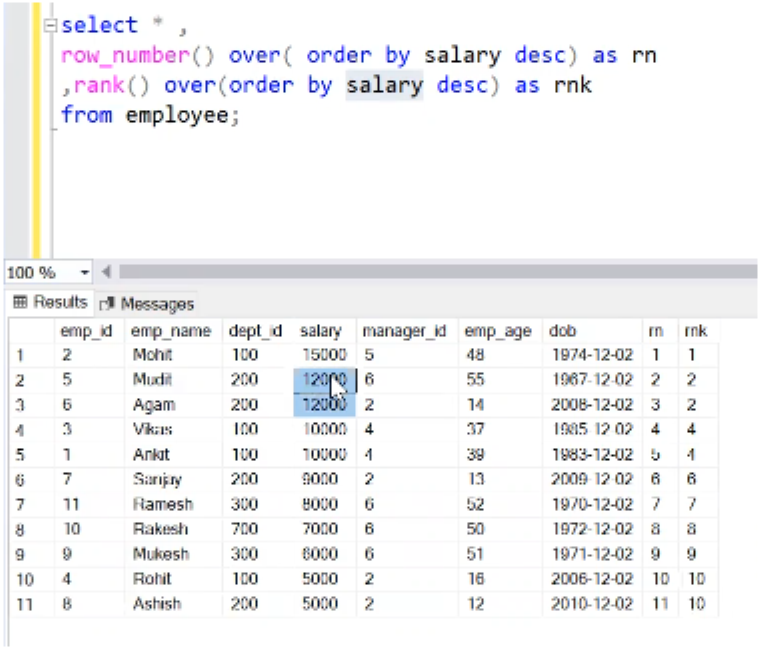
14. Partition creates windows based on a column which we provide…it is similar to group by…
15. In the above query..we have used partition by dept_id and order by salary asc..in over function
16. Based on this result..row_number() functions assigns row number to the result
17. Basically if we can see the output..we gave row_numbers to each windows of dept_id..provided from partition
18. Syntax for window function refer online..
19. Now to get top2 highest salaries from each dept..we have to use subqueries or CTE on our partition result
20. 

```
select * from (
select * ,
row_number() over(partition by dept_id order by salary desc,emp_name asc) as rn
from employee) A
where rn<=2
;
```

100 %

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn
1	2	Mohit	100	15000	5	48	1974-12-02	1
2	1	Ankit	100	10000	4	39	1983-12-02	2
3	6	Agam	200	12000	2	14	2008-12-02	1
4	5	Mudit	200	12000	6	55	1967-12-02	2
5	11	Ramesh	300	8000	6	52	1970-12-02	1
6	9	Mukesh	300	8000	6	51	1971-12-02	2
7	10	Rakesh	700	7000	6	50	1972-12-02	1

21. Here we have retrieved top 2 salaries from each department…with the help of window function,partition and sub queries…you can see the query and get intuition
22. Rank

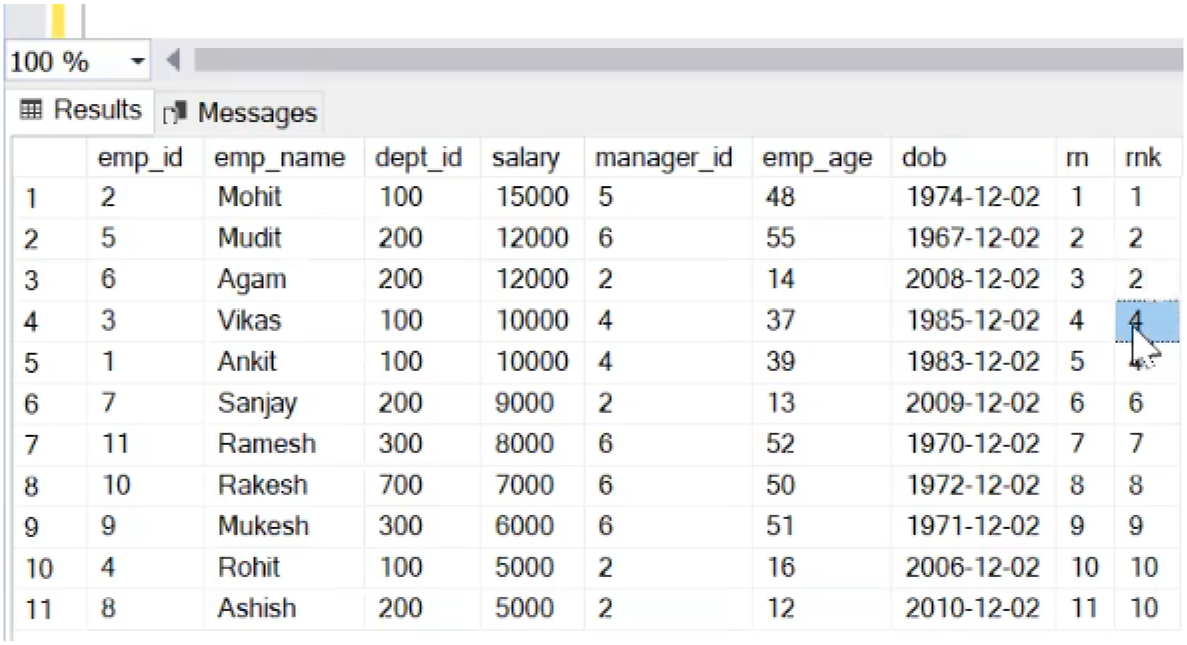
23. 

```
select * ,
row_number() over( order by salary desc) as rn
,rank() over(order by salary desc) as rnk
from employee;
```

100 %

Results | Messages

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn	rnk
1	2	Mohit	100	15000	5	48	1974-12-02	1	1
2	5	Mudit	200	12000	6	55	1967-12-02	2	2
3	6	Agam	200	12000	2	14	2008-12-02	3	2
4	3	Vikas	100	10000	4	37	1985-12-02	4	4
5	1	Ankit	100	10000	4	39	1983-12-02	5	4
6	7	Sanjay	200	9000	2	13	2009-12-02	6	6
7	11	Ramesh	300	8000	6	52	1970-12-02	7	7
8	10	Rakesh	700	7000	6	50	1972-12-02	8	8
9	9	Mukesh	300	6000	6	51	1971-12-02	9	9
10	4	Rohit	100	5000	2	16	2006-12-02	10	10
11	8	Ashish	200	5000	2	12	2010-12-02	11	10

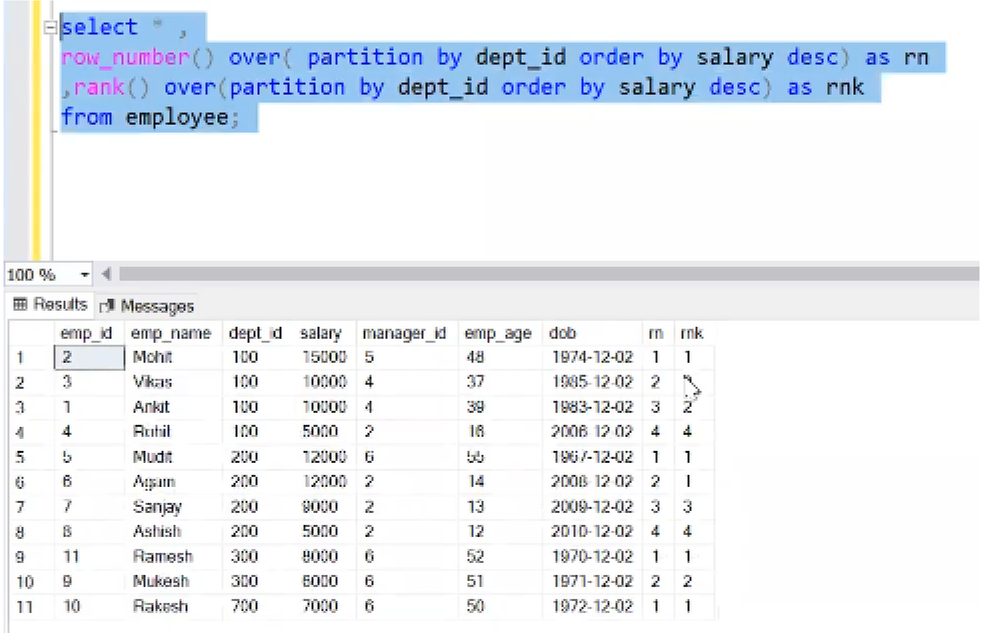
24. Here we have used rank function over salary ordered by desc…now this rank assigns ranks based on the salaries..if two persons has same salary,...it assigns the same rank..
25. The main difference between row_number and rank is….rank() function assigns same number/rank…if it has same column(salary in our case) value
26. 

100 %

Results | Messages

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn	rnk
1	2	Mohit	100	15000	5	48	1974-12-02	1	1
2	5	Mudit	200	12000	6	55	1967-12-02	2	2
3	6	Agam	200	12000	2	14	2008-12-02	3	2
4	3	Vikas	100	10000	4	37	1985-12-02	4	4
5	1	Ankit	100	10000	4	39	1983-12-02	5	4
6	7	Sanjay	200	9000	2	13	2009-12-02	6	6
7	11	Ramesh	300	8000	6	52	1970-12-02	7	7
8	10	Rakesh	700	7000	6	50	1972-12-02	8	8
9	9	Mukesh	300	6000	6	51	1971-12-02	9	9
10	4	Rohit	100	5000	2	16	2006-12-02	10	10
11	8	Ashish	200	5000	2	12	2010-12-02	11	10

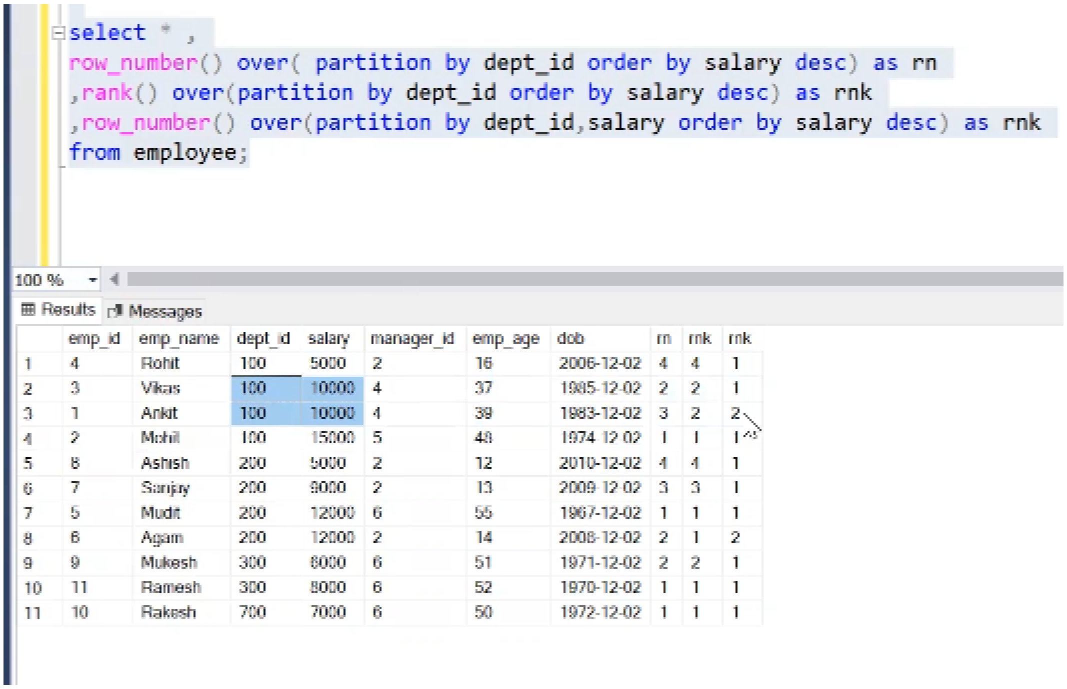
27. Here in rank…we don't have rank 3…because it assigns the rank..based on how many number of ranks are present before it…so here we have 1,2,2,and the next will rank will 4
28. Rank with partition by

29. 

```
select * ,
row_number() over( partition by dept_id order by salary desc) as rn
,rank() over(partition by dept_id order by salary desc) as rnk
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn	rnk
1	2	Mohit	100	15000	5	48	1974-12-02	1	1
2	3	Vikas	100	10000	4	37	1985-12-02	2	[illegible]
3	1	Ankit	100	10000	4	39	1983-12-02	3	2
4	4	Rohit	100	5000	2	16	2006-12-02	4	4
5	5	Mudit	200	12000	6	55	1967-12-02	1	1
6	6	Agam	200	12000	2	14	2008-12-02	2	1
7	7	Sanjay	200	9000	2	13	2009-12-02	3	3
8	8	Ashish	200	5000	2	12	2010-12-02	4	4
9	11	Ramesh	300	8000	6	52	1970-12-02	1	1
10	9	Mukesh	300	6000	6	51	1971-12-02	2	2
11	10	Rakesh	700	7000	6	50	1972-12-02	1	1

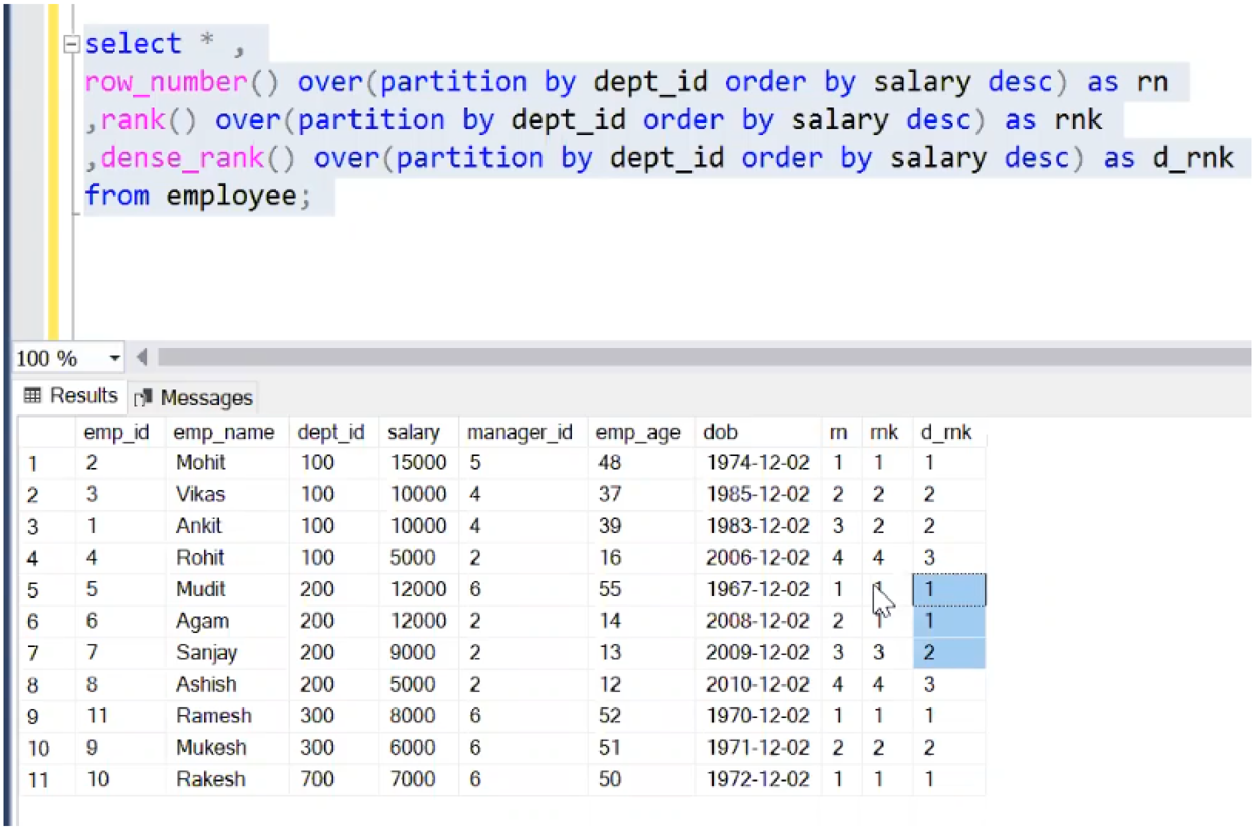
30. Using partition by on two columns

31. 

```
select * ,
row_number() over( partition by dept_id order by salary desc) as rn
,rank() over(partition by dept_id order by salary desc) as rnk
,row_number() over(partition by dept_id,salary order by salary desc) as rnk
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn	rnk	rnk
1	4	Rohit	100	5000	2	16	2006-12-02	4	4	1
2	3	Vikas	100	10000	4	37	1985-12-02	2	2	1
3	1	Ankit	100	10000	4	39	1983-12-02	3	2	2
4	2	Mohit	100	15000	5	48	1974-12-02	1	1	1
5	8	Ashish	200	5000	2	12	2010-12-02	4	4	1
6	7	Sanjay	200	9000	2	13	2009-12-02	3	3	1
7	5	Mudit	200	12000	6	55	1967-12-02	1	1	1
8	6	Agam	200	12000	2	14	2008-12-02	2	1	2
9	9	Mukesh	300	6000	6	51	1971-12-02	2	2	1
10	11	Ramesh	300	8000	6	52	1970-12-02	1	1	1
11	10	Rakesh	700	7000	6	50	1972-12-02	1	1	1

32. Here for every combination of dept_id and salary..it assigns a row number …see pic and understand
33. Next we have dense_rank()

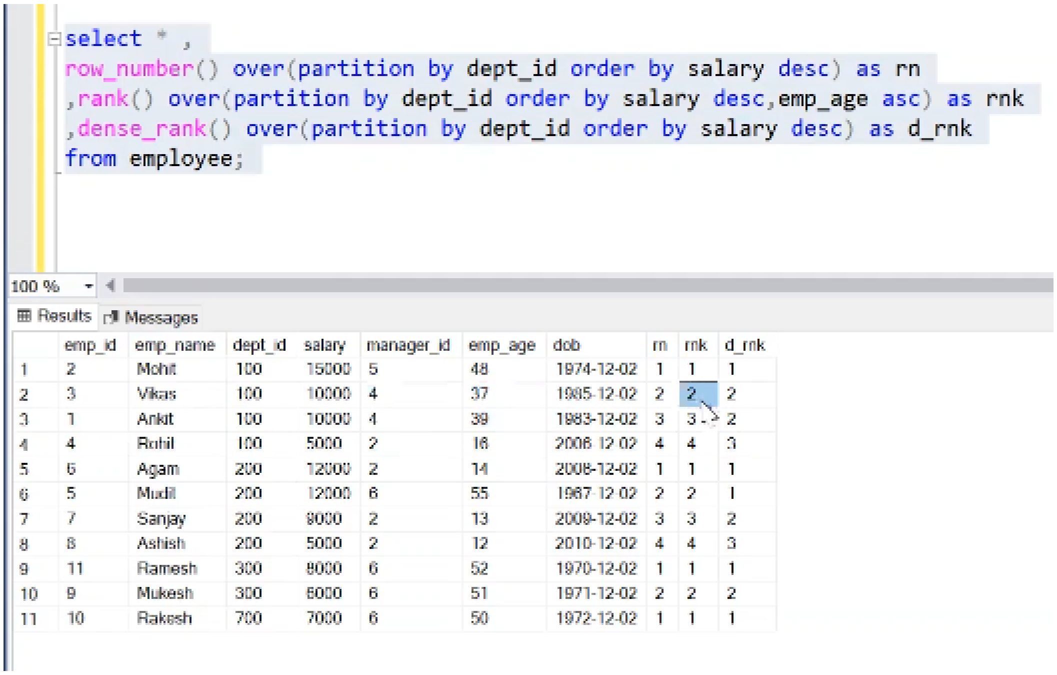


```
select * ,
row_number() over(partition by dept_id order by salary desc) as rn
,rank() over(partition by dept_id order by salary desc) as rnk
,dense_rank() over(partition by dept_id order by salary desc) as d_rnk
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn	rnk	d_rnk
1	2	Mohit	100	15000	5	48	1974-12-02	1	1	1
2	3	Vikas	100	10000	4	37	1985-12-02	2	2	2
3	1	Ankit	100	10000	4	39	1983-12-02	3	2	2
4	4	Rohit	100	5000	2	16	2006-12-02	4	4	3
5	5	Mudit	200	12000	6	55	1967-12-02	1	1	1
6	6	Agam	200	12000	2	14	2008-12-02	2	1	1
7	7	Sanjay	200	9000	2	13	2009-12-02	3	3	2
8	8	Ashish	200	5000	2	12	2010-12-02	4	4	3
9	11	Ramesh	300	8000	6	52	1970-12-02	1	1	1
10	9	Mukesh	300	6000	6	51	1971-12-02	2	2	2
11	10	Rakesh	700	7000	6	50	1972-12-02	1	1	1

34\.

35. It is similar to rank()...main difference is in assigning ranks…dense_rank() will not skip any rank numbers….see pic and understand



```
select * ,
row_number() over(partition by dept_id order by salary desc) as rn
,rank() over(partition by dept_id order by salary desc,emp_age asc) as rnk
,dense_rank() over(partition by dept_id order by salary desc) as d_rnk
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	rn	rnk	d_rnk
1	2	Mohit	100	15000	5	48	1974-12-02	1	1	1
2	3	Vikas	100	10000	4	37	1985-12-02	2	2	2
3	1	Ankit	100	10000	4	39	1983-12-02	3	3	2
4	4	Rohit	100	5000	2	16	2006-12-02	4	4	3
5	6	Agam	200	12000	2	14	2008-12-02	1	1	1
6	5	Mudit	200	12000	6	55	1967-12-02	2	2	1
7	7	Sanjay	200	9000	2	13	2009-12-02	3	3	2
8	8	Ashish	200	5000	2	12	2010-12-02	4	4	3
9	11	Ramesh	300	8000	6	52	1970-12-02	1	1	1
10	9	Mukesh	300	6000	6	51	1971-12-02	2	2	2
11	10	Rakesh	700	7000	6	50	1972-12-02	1	1	1

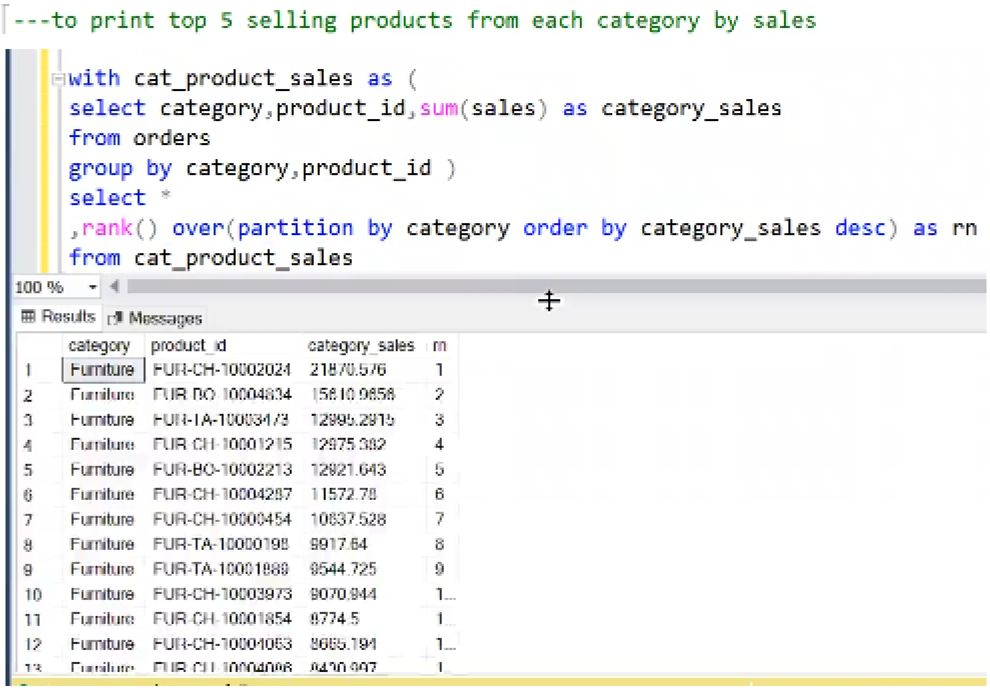
36\.

37. In the above query..rank is based on highest salary and lowest age

38. Row_number,rank and dense rank are very similar functions

39. These functions will be asked in **INTERVIEWS**

40.



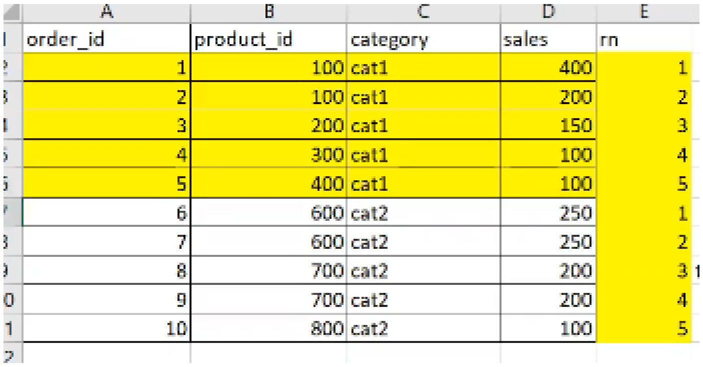
```
---to print top 5 selling products from each category by sales

with cat_product_sales as (
select category,product_id,sum(sales) as category_sales
from orders
group by category,product_id )
select *
,rank() over(partition by category order by category_sales desc) as rn
from cat_product_sales
```

```
     category   product_id         category_sales   rn
1    Furniture  FUR-CH-10002024    21870.576        1
2    Furniture  FUR-BO-10004834    15610.9656       2
3    Furniture  FUR-TA-10003473    12995.2915       3
4    Furniture  FUR-CH-10001215    12975.382        4
5    Furniture  FUR-BO-10002213    12821.643        5
6    Furniture  FUR-CH-10004287    11572.78         6
7    Furniture  FUR-CH-10000454    10637.528        7
8    Furniture  FUR-TA-10000198    9917.64          8
9    Furniture  FUR-TA-10001889    9544.725         9
10   Furniture  FUR-CH-10003973    9070.944         1...
11   Furniture  FUR-CH-10001854    8774.5           1...
12   Furniture  FUR-CH-10004063    8665.194         1...
13   Furniture  FUR-CH-10004086    8430.997         1
```

41.

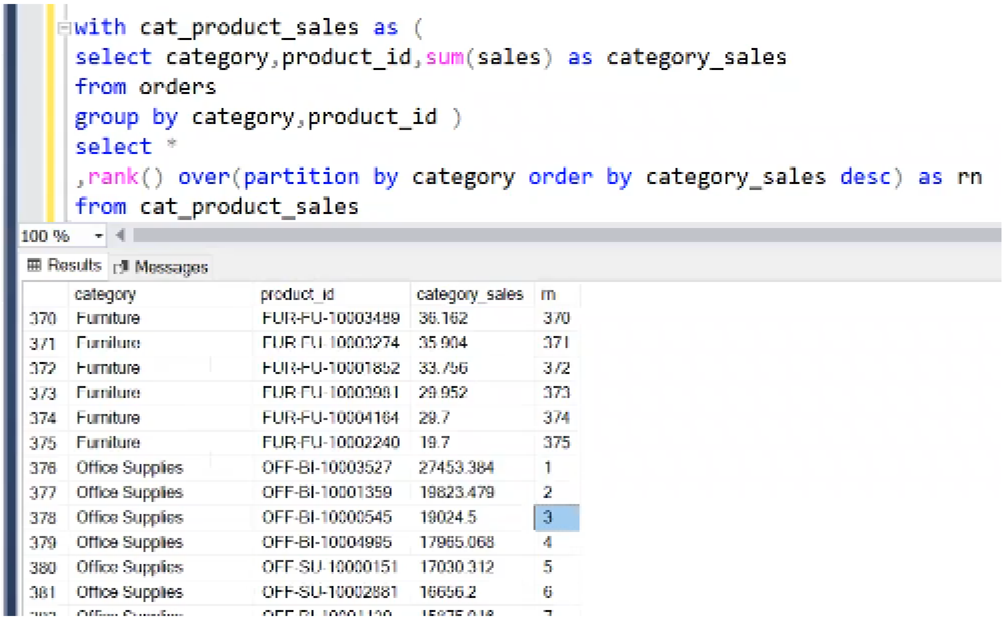
42. Here first we have computed sum(sales) on category and product_id..and we have used CTE …then we have applied rank(), over(partition by category…)

43. If we didnt compute sales based on category and product



order_id	product_id	category	sales	rn
1	100	cat1	400	1
2	100	cat1	200	2
3	200	cat1	150	3
4	300	cat1	100	4
5	400	cat1	100	5
6	600	cat2	250	1
7	600	cat2	250	2
8	700	cat2	200	3
9	700	cat2	200	4
10	800	cat2	100	5

...here for cat1 and product =100….total sale must be 600…so first we have computed total sales based on category and product_id ..and then we have applied rank and partitioned them by category and sales(see query)

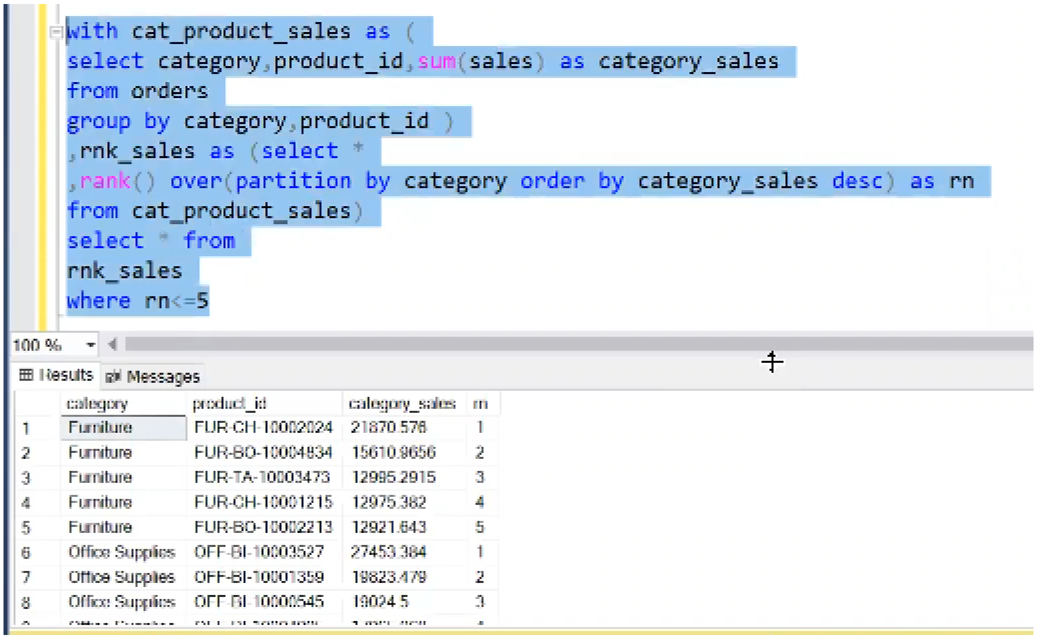
44. 

```
with cat_product_sales as (
select category,product_id,sum(sales) as category_sales
from orders
group by category,product_id )
select *
,rank() over(partition by category order by category_sales desc) as rn
from cat_product_sales
```

100 %

Results | Messages

	category	product_id	category_sales	rn
370	Furniture	FUR-FU-10003489	36.162	370
371	Furniture	FUR-FU-10003274	35.904	371
372	Furniture	FUR-FU-10001852	33.756	372
373	Furniture	FUR-FU-10003981	29.952	373
374	Furniture	FUR-FU-10004164	28.7	374
375	Furniture	FUR-FU-10002240	19.7	375
376	Office Supplies	OFF-BI-10003527	27453.384	1
377	Office Supplies	OFF-BI-10001359	19823.479	2
378	Office Supplies	OFF-BI-10000545	19024.5	3
379	Office Supplies	OFF-BI-10004995	17965.068	4
380	Office Supplies	OFF-SU-10000151	17030.312	5
381	Office Supplies	OFF-SU-10002881	16656.2	6

45. Here we have computed..top selling products in each category…and assigned them ranks..see pic
46. Now to select top 5 from it…we have use this result as an CTE…and on top of we have to retrieve top 5
47. 

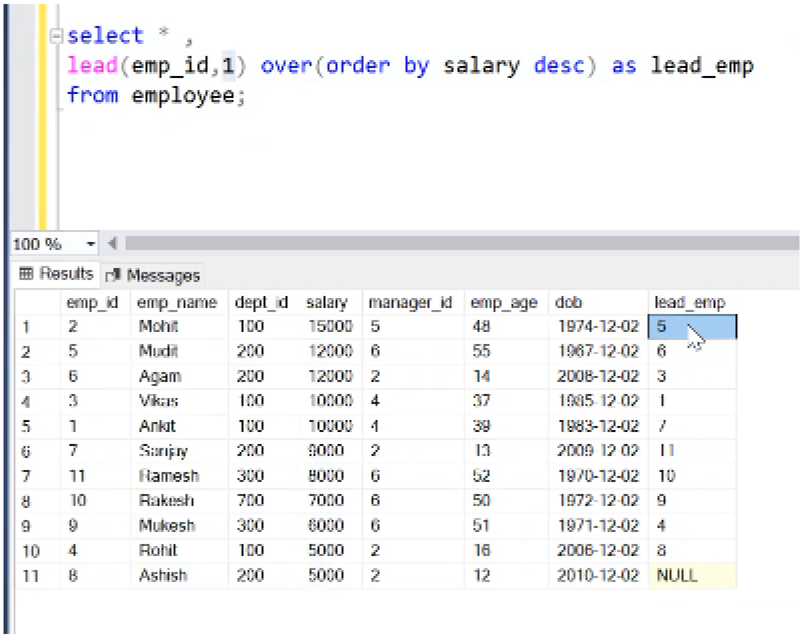
```
with cat_product_sales as (
select category,product_id,sum(sales) as category_sales
from orders
group by category,product_id )
,rnk_sales as (select *
,rank() over(partition by category order by category_sales desc) as rn
from cat_product_sales)
select * from
rnk_sales
where rn<=5
```

100 %

Results | Messages

	category	product_id	category_sales	rn
1	Furniture	FUR-CH-10002024	21870.576	1
2	Furniture	FUR-BO-10004834	15610.9656	2
3	Furniture	FUR-TA-10003473	12995.2915	3
4	Furniture	FUR-CH-10001215	12975.382	4
5	Furniture	FUR-BO-10002213	12921.643	5
6	Office Supplies	OFF-BI-10003527	27453.384	1
7	Office Supplies	OFF-BI-10001359	19823.479	2
8	Office Supplies	OFF-BI-10000545	19024.5	3

48. Here we have retrieved top 5 selling products in each category
49. Lead function

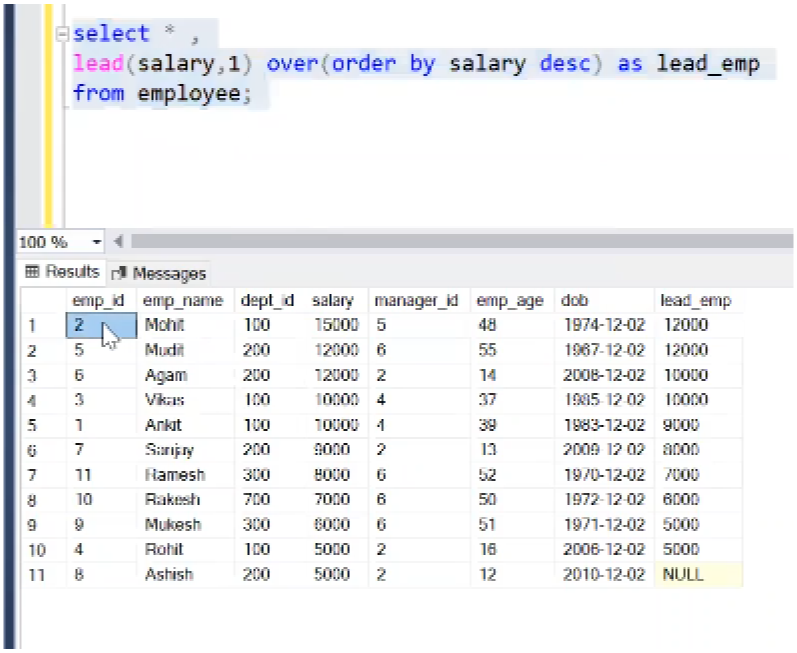


```sql
select * ,
lead(emp_id,1) over(order by salary desc) as lead_emp
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_emp
1	2	Mohit	100	15000	5	48	1974-12-02	5
2	5	Mudit	200	12000	6	55	1967-12-02	6
3	6	Agam	200	12000	2	14	2008-12-02	3
4	3	Vikas	100	10000	4	37	1985-12-02	1
5	1	Ankit	100	10000	4	39	1983-12-02	7
6	7	Sanjay	200	9000	2	13	2009-12-02	11
7	11	Ramesh	300	8000	6	52	1970-12-02	10
8	10	Rakesh	700	7000	6	50	1972-12-02	9
9	9	Mukesh	300	6000	6	51	1971-12-02	4
10	4	Rohit	100	5000	2	16	2006-12-02	8
11	8	Ashish	200	5000	2	12	2010-12-02	NULL

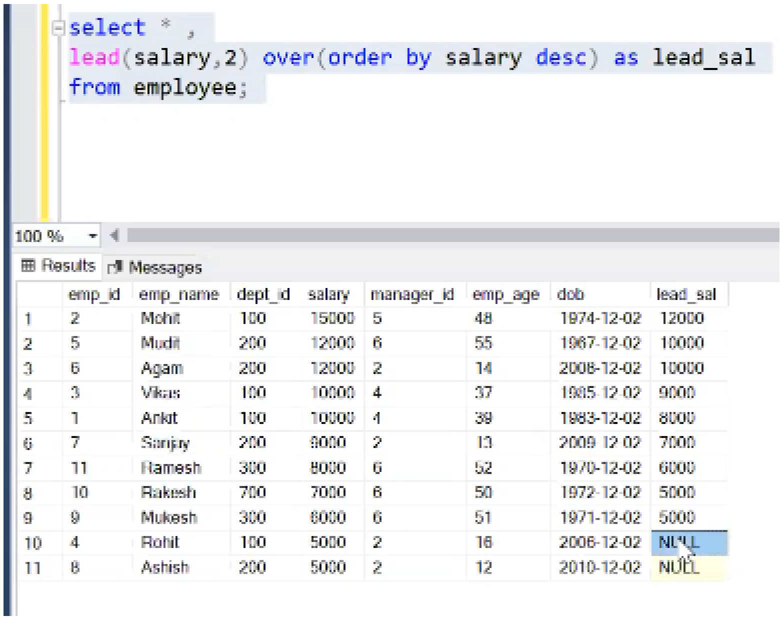
50.

51. Here in lead we gave (emp_id,1) over(order by salary desc)....in the lead_emp…we get id of 12000 salary(emp_id=5)..and for next salary 12000(emp_id=6)...similarly for all salaries of emp_id..it gives their corresponding id…in the lead_emp output
52. Here lead(emp_id,1) means give employee id of next row..if it is lead(emp_id,2)..then,,give employee id of next next row(skips two people in between)
53. Lead on salary…see pic and understand



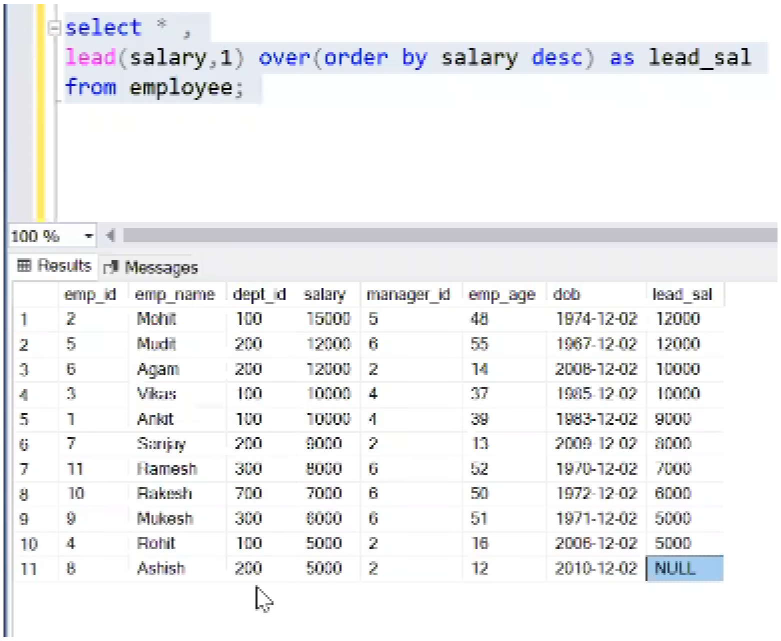
```sql
select * ,
lead(salary,1) over(order by salary desc) as lead_emp
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_emp
1	2	Mohit	100	15000	5	48	1974-12-02	12000
2	5	Mudit	200	12000	6	55	1967-12-02	12000
3	6	Agam	200	12000	2	14	2008-12-02	10000
4	3	Vikas	100	10000	4	37	1985-12-02	10000
5	1	Ankit	100	10000	4	39	1983-12-02	9000
6	7	Sanjay	200	9000	2	13	2009-12-02	8000
7	11	Ramesh	300	8000	6	52	1970-12-02	7000
8	10	Rakesh	700	7000	6	50	1972-12-02	6000
9	9	Mukesh	300	6000	6	51	1971-12-02	5000
10	4	Rohit	100	5000	2	16	2006-12-02	5000
11	8	Ashish	200	5000	2	12	2010-12-02	NULL

54. 

```sql
select * ,
lead(salary,2) over(order by salary desc) as lead_sal
from employee;
```

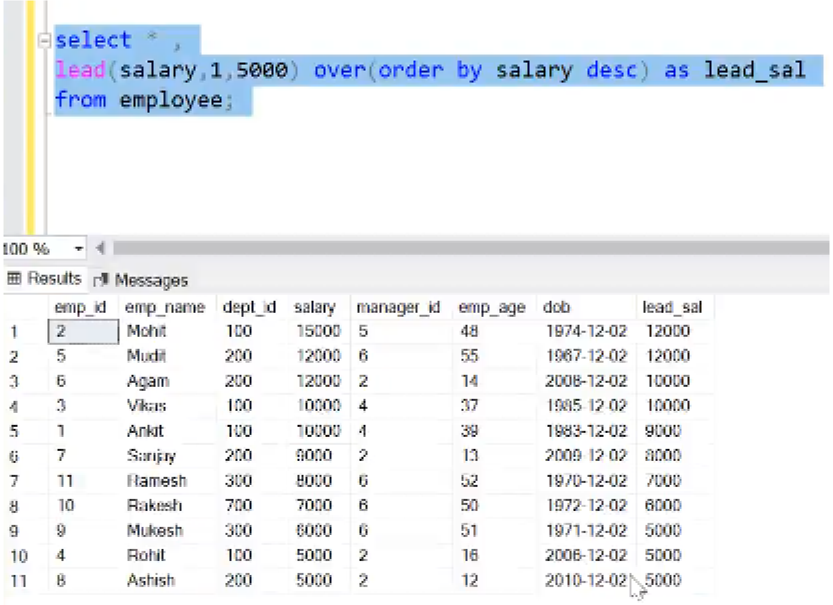
	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	12000
2	5	Mudit	200	12000	6	55	1967-12-02	10000
3	6	Agam	200	12000	2	14	2008-12-02	10000
4	3	Vikas	100	10000	4	37	1985-12-02	9000
5	1	Ankit	100	10000	4	39	1983-12-02	8000
6	7	Sanjay	200	9000	2	13	2009-12-02	7000
7	11	Ramesh	300	8000	6	52	1970-12-02	6000
8	10	Rakesh	700	7000	6	50	1972-12-02	5000
9	9	Mukesh	300	6000	6	51	1971-12-02	5000
10	4	Rohit	100	5000	2	16	2006-12-02	NULL
11	8	Ashish	200	5000	2	12	2010-12-02	NULL

55. 

```sql
select * ,
lead(salary,1) over(order by salary desc) as lead_sal
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	12000
2	5	Mudit	200	12000	6	55	1967-12-02	12000
3	6	Agam	200	12000	2	14	2008-12-02	10000
4	3	Vikas	100	10000	4	37	1985-12-02	10000
5	1	Ankit	100	10000	4	39	1983-12-02	9000
6	7	Sanjay	200	9000	2	13	2009-12-02	8000
7	11	Ramesh	300	8000	6	52	1970-12-02	7000
8	10	Rakesh	700	7000	6	50	1972-12-02	6000
9	9	Mukesh	300	6000	6	51	1971-12-02	5000
10	4	Rohit	100	5000	2	16	2006-12-02	5000
11	8	Ashish	200	5000	2	12	2010-12-02	NULL

56. Here in the below query…we have a default value(5000) in lead(salary,1,5000)..wherever we have null..it replaces the value to default value

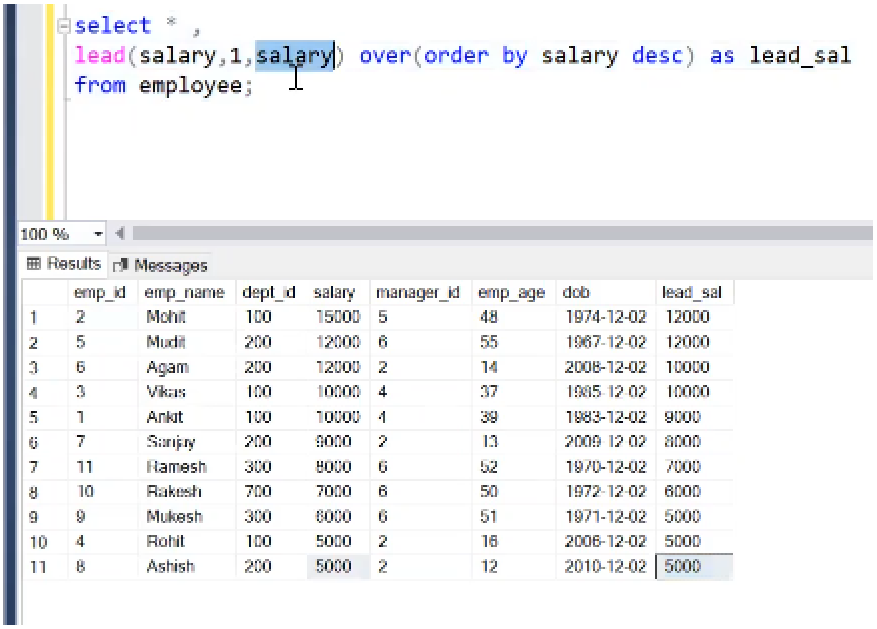


```sql
select * ,
lead(salary,1,5000) over(order by salary desc) as lead_sal
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	12000
2	5	Mudit	200	12000	6	55	1967-12-02	12000
3	6	Agam	200	12000	2	14	2008-12-02	10000
4	3	Vikas	100	10000	4	37	1985-12-02	10000
5	1	Ankit	100	10000	4	39	1983-12-02	9000
6	7	Sanjay	200	9000	2	13	2009-12-02	8000
7	11	Ramesh	300	8000	6	52	1970-12-02	7000
8	10	Rakesh	700	7000	6	50	1972-12-02	6000
9	9	Mukesh	300	6000	6	51	1971-12-02	5000
10	4	Rohit	100	5000	2	16	2006-12-02	5000
11	8	Ashish	200	5000	2	12	2010-12-02	5000

57.

58. In the below query we gave salary as a default value..it gives the corresponding value of row's salary…instead of null

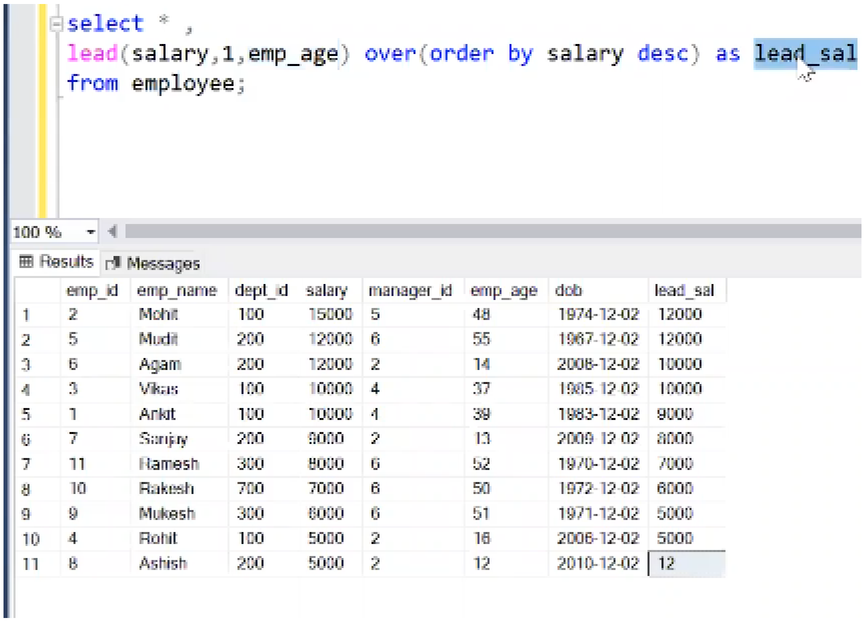


```sql
select * ,
lead(salary,1,salary) over(order by salary desc) as lead_sal
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	12000
2	5	Mudit	200	12000	6	55	1967-12-02	12000
3	6	Agam	200	12000	2	14	2008-12-02	10000
4	3	Vikas	100	10000	4	37	1985-12-02	10000
5	1	Ankit	100	10000	4	39	1983-12-02	9000
6	7	Sanjay	200	9000	2	13	2009-12-02	8000
7	11	Ramesh	300	8000	6	52	1970-12-02	7000
8	10	Rakesh	700	7000	6	50	1972-12-02	6000
9	9	Mukesh	300	6000	6	51	1971-12-02	5000
10	4	Rohit	100	5000	2	16	2006-12-02	5000
11	8	Ashish	200	5000	2	12	2010-12-02	5000

59.

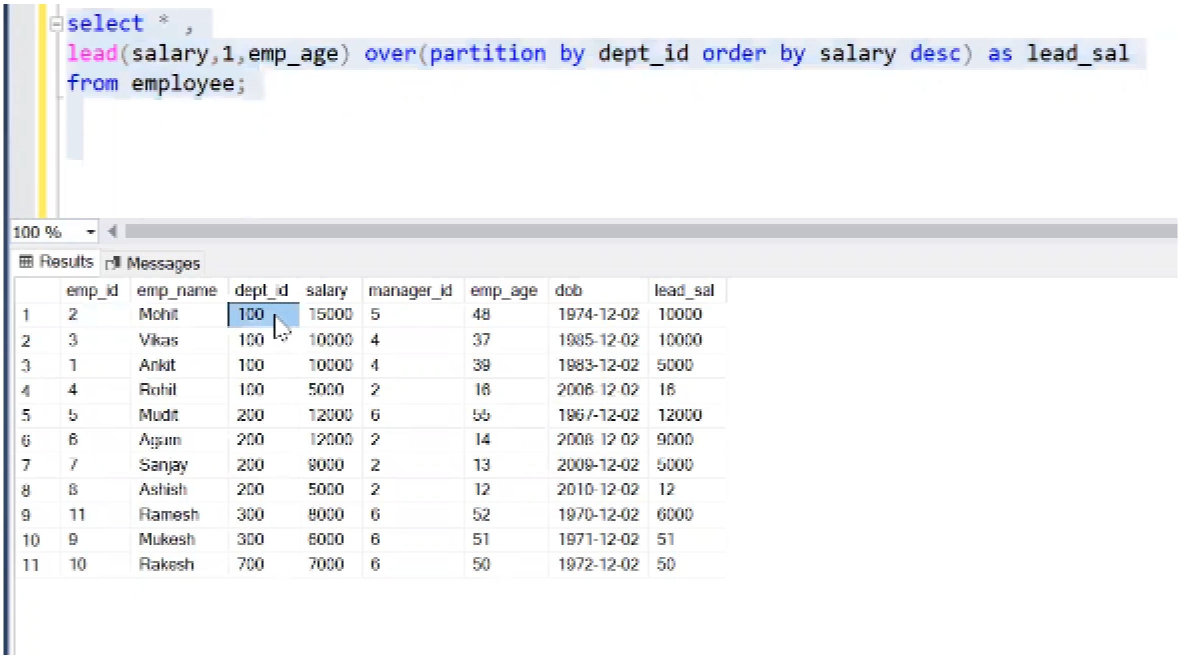
60. In the below query we gave emp_age as default value..see pic

61. 

```sql
select * ,
lead(salary,1,emp_age) over(order by salary desc) as lead_sal
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	12000
2	5	Mudit	200	12000	6	55	1967-12-02	12000
3	6	Agam	200	12000	2	14	2008-12-02	10000
4	3	Vikas	100	10000	4	37	1985-12-02	10000
5	1	Ankit	100	10000	4	39	1983-12-02	9000
6	7	Sanjay	200	9000	2	13	2009-12-02	8000
7	11	Ramesh	300	8000	6	52	1970-12-02	7000
8	10	Rakesh	700	7000	6	50	1972-12-02	6000
9	9	Mukesh	300	6000	6	51	1971-12-02	5000
10	4	Rohit	100	5000	2	16	2006-12-02	5000
11	8	Ashish	200	5000	2	12	2010-12-02	12

62. Using partition and lead

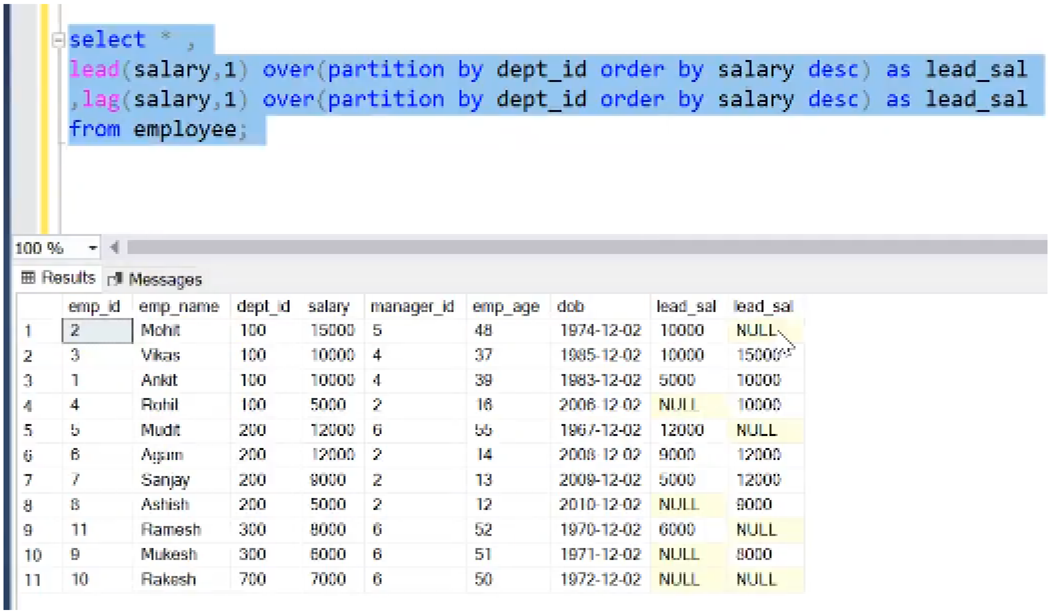
63. 

```sql
select * ,
lead(salary,1,emp_age) over(partition by dept_id order by salary desc) as lead_sal
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	10000
2	3	Vikas	100	10000	4	37	1985-12-02	10000
3	1	Ankit	100	10000	4	39	1983-12-02	5000
4	4	Rohit	100	5000	2	16	2006-12-02	16
5	5	Mudit	200	12000	6	55	1967-12-02	12000
6	6	Agam	200	12000	2	14	2008-12-02	9000
7	7	Sanjay	200	9000	2	13	2009-12-02	5000
8	8	Ashish	200	5000	2	12	2010-12-02	12
9	11	Ramesh	300	8000	6	52	1970-12-02	6000
10	9	Mukesh	300	6000	6	51	1971-12-02	51
11	10	Rakesh	700	7000	6	50	1972-12-02	50

64. Next we have lag…it is similar to lead…in lead..it gives us the next value..LAG is same…but it gives us previous value instead of next

65.

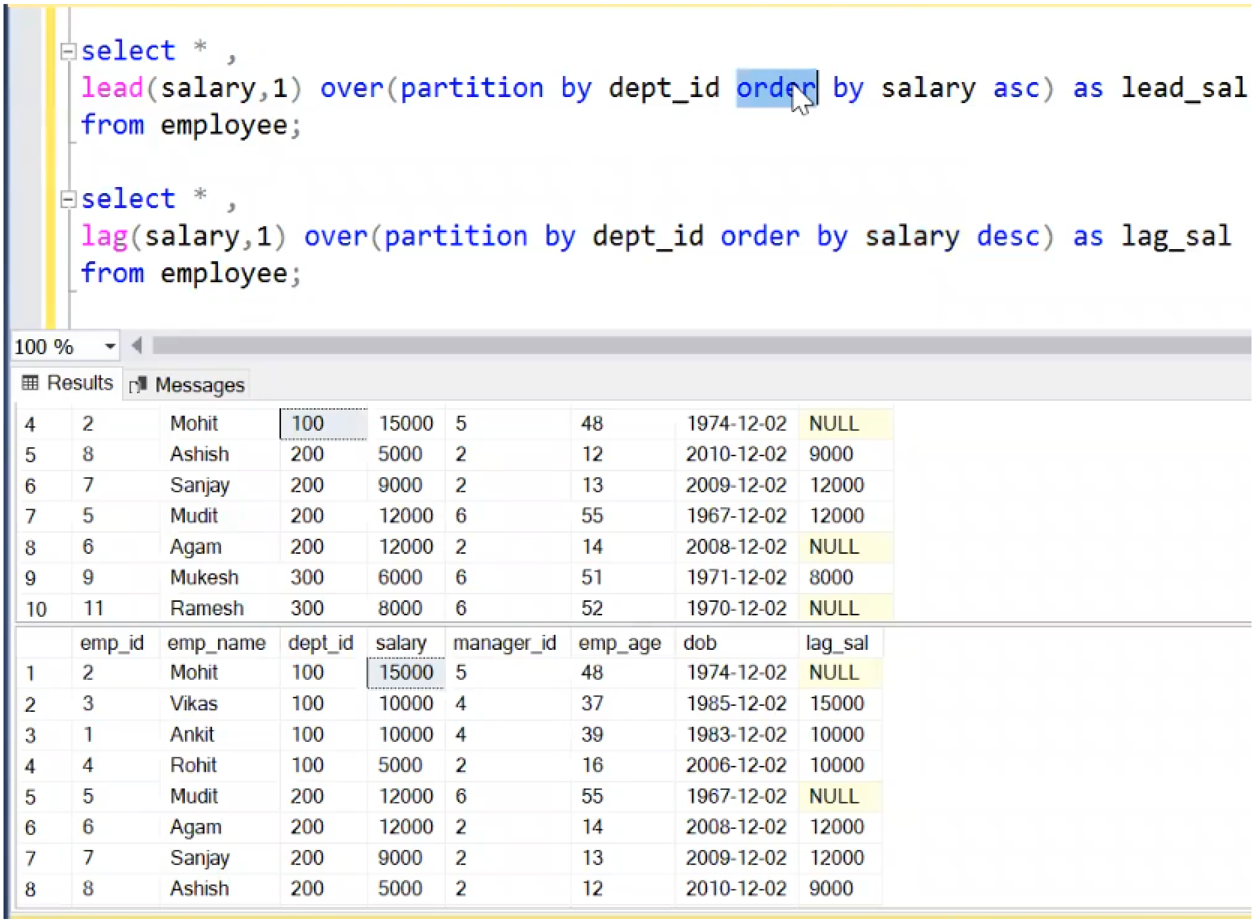


```
select * ,
lead(salary,1) over(partition by dept_id order by salary desc) as lead_sal
,lag(salary,1) over(partition by dept_id order by salary desc) as lead_sal
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal	lead_sal
1	2	Mohit	100	15000	5	48	1974-12-02	10000	NULL
2	3	Vikas	100	10000	4	37	1985-12-02	10000	15000
3	1	Ankit	100	10000	4	39	1983-12-02	5000	10000
4	4	Rohit	100	5000	2	16	2006-12-02	NULL	10000
5	5	Mudit	200	12000	6	55	1967-12-02	12000	NULL
6	6	Agam	200	12000	2	14	2008-12-02	9000	12000
7	7	Sanjay	200	9000	2	13	2009-12-02	5000	12000
8	8	Ashish	200	5000	2	12	2010-12-02	NULL	9000
9	11	Ramesh	300	8000	6	52	1970-12-02	6000	NULL
10	9	Mukesh	300	6000	6	51	1971-12-02	NULL	8000
11	10	Rakesh	700	7000	6	50	1972-12-02	NULL	NULL

66. Basically ..lead will look for the value of next row..based on order_by
67. And lag will look for the value of previous row…based on order_by
68. See the pic and understand

69.



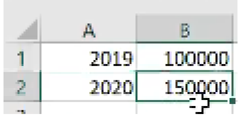
```
select * ,
lead(salary,1) over(partition by dept_id order by salary asc) as lead_sal
from employee;

select * ,
lag(salary,1) over(partition by dept_id order by salary desc) as lag_sal
from employee;
```

4	2	Mohit	100	15000	5	48	1974-12-02	NULL
5	8	Ashish	200	5000	2	12	2010-12-02	9000
6	7	Sanjay	200	9000	2	13	2009-12-02	12000
7	5	Mudit	200	12000	6	55	1967-12-02	12000
8	6	Agam	200	12000	2	14	2008-12-02	NULL
9	9	Mukesh	300	6000	6	51	1971-12-02	8000
10	11	Ramesh	300	8000	6	52	1970-12-02	NULL

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lag_sal
1	2	Mohit	100	15000	5	48	1974-12-02	NULL
2	3	Vikas	100	10000	4	37	1985-12-02	15000
3	1	Ankit	100	10000	4	39	1983-12-02	10000
4	4	Rohit	100	5000	2	16	2006-12-02	10000
5	5	Mudit	200	12000	6	55	1967-12-02	NULL
6	6	Agam	200	12000	2	14	2008-12-02	12000
7	7	Sanjay	200	9000	2	13	2009-12-02	12000
8	8	Ashish	200	5000	2	12	2010-12-02	9000

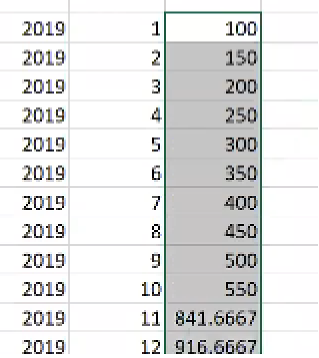
70. Use case

71. If we want to perform year wise sales

72. 

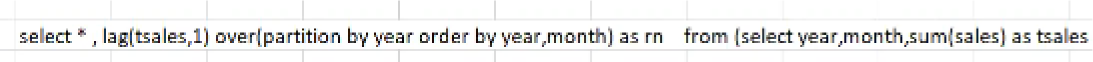
	A	B
1	2019	100000
2	2020	150000

here in 2019 we make 100k and in 2020 we made 150k

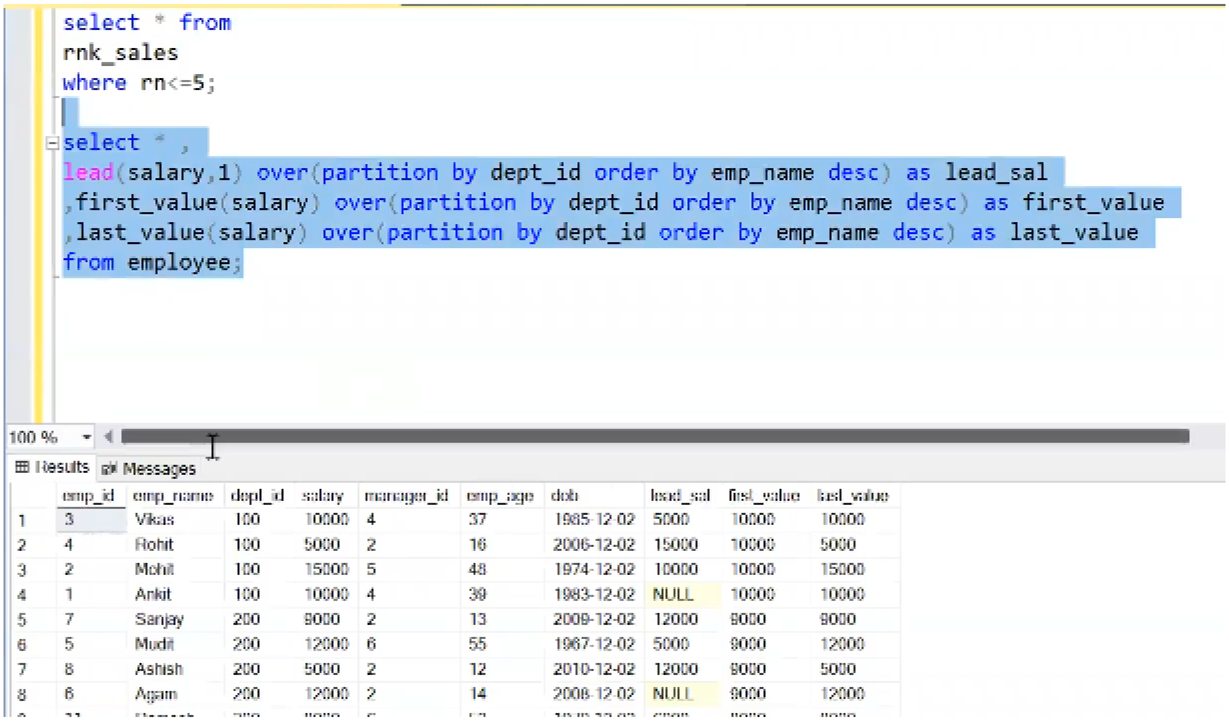
73. 

2019	1	100
2019	2	150
2019	3	200
2019	4	250
2019	5	300
2019	6	350
2019	7	400
2019	8	450
2019	9	500
2019	10	550
2019	11	841.6667
2019	12	916.6667

74. month wise sales …here we will group by year and month

75. 

```
select * , lag(tsales,1) over(partition by year order by year,month) as rn   from (select year,month,sum(sales) as tsales
```

76. 

```
select * from
rnk_sales
where rn<=5;

select * ,
lead(salary,1) over(partition by dept_id order by emp_name desc) as lead_sal
,first_value(salary) over(partition by dept_id order by emp_name desc) as first_value
,last_value(salary) over(partition by dept_id order by emp_name desc) as last_value
from employee;
```

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	lead_sal	first_value	last_value
1	3	Vikas	100	10000	4	37	1985-12-02	5000	10000	10000
2	4	Rohit	100	5000	2	16	2006-12-02	15000	10000	5000
3	2	Mohit	100	15000	5	48	1974-12-02	10000	10000	15000
4	1	Ankit	100	10000	4	39	1983-12-02	NULL	10000	10000
5	7	Sanjay	200	9000	2	13	2009-12-02	12000	9000	9000
6	5	Mudit	200	12000	6	55	1967-12-02	5000	9000	12000
7	8	Ashish	200	5000	2	12	2010-12-02	12000	9000	5000
8	6	Agam	200	12000	2	14	2008-12-02	NULL	9000	12000

77.