FUNCTIONS

Using having and group by clause

select count(name) as count_emp, salary from employee group by salary having salary=50000;

Results Messages

	count_employee	salary
1	2	50000

Using avg function

select avg(salary) from employee where department='HR';

Results Messages

	(No column name)
1	43000

Using count function

select count(id) as count_value from employee where name='stella';

Results Messages

	count_value
1	4

Using group by, having with aggregate function

select max(salary), department from employee group by department having department='HR';

Results Messages

	(No column name)	department
1	50000	HR

Row_number() -giving consecutive numbers to rank

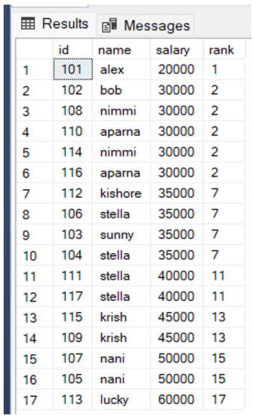
select id, name, salary, ROW_NUMBER() over(order by salary desc) as rownumber from employee;

Results Messages

	id	name	salary	rownumber
1	113	lucky	60000	1
2	105	nani	50000	2
3	107	nani	50000	3
4	115	krish	45000	4
5	109	krish	45000	5
6	111	stella	40000	6
7	117	stella	40000	7
8	112	kishore	35000	8
9	106	stella	35000	9
10	103	sunny	35000	10
11	104	stella	35000	11
12	102	bob	30000	12
13	108	nimmi	30000	13
14	110	aparna	30000	14
15	116	aparna	30000	15
16	114	nimmi	30000	16
17	101	alex	20000	17

rank()-used to give rank if duplicates allowed ranking will be changed based on duplicates

select id,name,salary,rank() over(order by salary) as rank from employee;

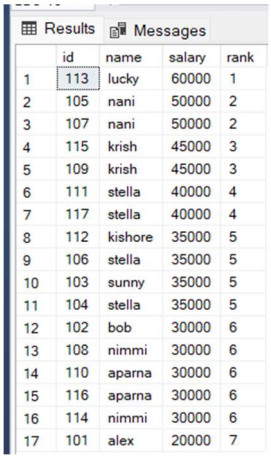


Results Messages

	id	name	salary	rank
1	101	alex	20000	1
2	102	bob	30000	2
3	108	nimmi	30000	2
4	110	aparna	30000	2
5	114	nimmi	30000	2
6	116	aparna	30000	2
7	112	kishore	35000	7
8	106	stella	35000	7
9	103	sunny	35000	7
10	104	stella	35000	7
11	111	stella	40000	11
12	117	stella	40000	11
13	115	krish	45000	13
14	109	krish	45000	13
15	107	nani	50000	15
16	105	nani	50000	15
17	113	lucky	60000	17

dense_rank()-used to give ranks consecutively even if duplicates are allowed

select id,name,salary,dense_rank() over(order by salary desc) as rank from employee;



Results Messages

	id	name	salary	rank
1	113	lucky	60000	1
2	105	nani	50000	2
3	107	nani	50000	2
4	115	krish	45000	3
5	109	krish	45000	3
6	111	stella	40000	4
7	117	stella	40000	4
8	112	kishore	35000	5
9	106	stella	35000	5
10	103	sunny	35000	5
11	104	stella	35000	5
12	102	bob	30000	6
13	108	nimmi	30000	6
14	110	aparna	30000	6
15	116	aparna	30000	6
16	114	nimmi	30000	6
17	101	alex	20000	7

ntile() function- it will divide give the rank in groups

select id,name,salary,ntile(2) over(order by salary) as rank from employee; --without condition

Results Messages

	id	name	salary	rank
1	101	alex	20000	1
2	102	bob	30000	1
3	108	nimmi	30000	1
4	110	aparna	30000	1
5	114	nimmi	30000	1
6	116	aparna	30000	1
7	112	kishore	35000	1
8	106	stella	35000	1
9	103	sunny	35000	1
10	104	stella	35000	2
11	111	stella	40000	2
12	117	stella	40000	2
13	115	krish	45000	2
14	109	krish	45000	2
15	107	nani	50000	2
16	105	nani	50000	2
17	113	lucky	60000	2

```
select name,salary,ntile(4) over(order by salary) as rank from employee where salary>10000;
--with condition
```

Results Messages

	name	salary	rank
1	alex	20000	1
2	bob	30000	1
3	nimmi	30000	1
4	aparna	30000	1
5	nimmi	30000	1
6	aparna	30000	2
7	kishore	35000	2
8	stella	35000	2
9	sunny	35000	2
10	stella	35000	3
11	stella	40000	3
12	stella	40000	3
13	krish	45000	3
14	krish	45000	4
15	nani	50000	4
16	nani	50000	4
17	lucky	60000	4

ROLLUP OPERATION

```
SELECT  SalesYear, SUM(SalesTotal) AS SalesTotal FROM SalesList  GROUP BY  ROLLUP
(SalesYear);
```

Results Messages

	SalesYear	SalesTotal
1	2021	318500.00
2	2022	362000.00
3	2023	375000.00
4	2024	270500.00
5	NULL	1326000.00

```
SELECT  SalesYear,SalesQuartes, SUM(SalesTotal) AS SalesTotal  FROM SalesList GROUP BY
ROLLUP(SalesYear, SalesQuartes)
```

Results Messages

	SalesYear	SalesQuartes	SalesTotal
1	2021	Q1	63000.00
2	2021	Q2	75000.00
3	2021	Q3	83500.00
4	2021	Q4	97000.00
5	2021	NULL	318500.00
6	2022	Q1	65000.00
7	2022	Q2	83000.00
8	2022	Q3	99500.00
9	2022	Q4	114500.00
10	2022	NULL	362000.00
11	2023	Q1	73500.00
12	2023	Q2	88000.00
13	2023	Q3	100500.00
14	2023	Q4	113000.00
15	2023	NULL	375000.00
16	2024	Q1	76500.00
17	2024	Q2	91500.00
18	2024	Q3	102500.00
19	2024	NULL	270500.00
20	NULL	NULL	1326000...

```
SELECT  SalesYear,SalesQuartes,SalesMonth ,SUM(SalesTotal) AS SalesTotal  FROM SalesList
GROUP BY ROLLUP(SalesYear, SalesQuartes, SalesMonth);
```

Results Messages

	SalesYear	SalesQuartes	SalesMonth	SalesTotal
35	2023	Q1	February	24500.00
36	2023	Q1	January	23000.00
37	2023	Q1	March	26000.00
38	2023	Q1	NULL	73500.00
39	2023	Q2	April	28000.00
40	2023	Q2	June	30500.00
41	2023	Q2	May	29500.00
42	2023	Q2	NULL	88000.00
43	2023	Q3	August	33500.00
44	2023	Q3	July	32000.00
45	2023	Q3	September	35000.00
46	2023	Q3	NULL	100500.00
47	2023	Q4	December	39500.00
48	2023	Q4	November	37500.00
49	2023	Q4	October	36000.00
50	2023	Q4	NULL	113000.00
51	2023	NULL	NULL	375000.00
52	2024	Q1	February	25500.00
53	2024	Q1	January	24500.00
54	2024	Q1	March	26500.00
55	2024	Q1	NULL	76500.00
56	2024	Q2	April	29000.00
57	2024	Q2	June	32000.00
58	2024	Q2	May	30500.00
59	2024	Q2	NULL	91500.00
60	2024	Q3	August	34000.00
61	2024	Q3	July	33000.00
62	2024	Q3	September	35500.00
63	2024	Q3	NULL	102500.00
64	2024	NULL	NULL	270500.00
65	NULL	NULL	NULL	1326000...

STORED PROCEDURE

```
CREATE PROCEDURE dept_102
AS
BEGIN
   SELECT EmployeeID, FirstName
   FROM Employee_New
   WHERE DepartmentID = 102;
END;
```

---TO RUN--

EXEC dept_102;

Results Messages

	EmployeeID	FirstName
1	107	Karthik
2	108	Megha
3	109	Ananya
4	128	Dinesh
5	129	Madhavi
6	130	Preeti

```
CREATE PROCEDURE GetEmployeeDetails
    @DepartmentID INT
AS
BEGIN
    -- Selecting Employee details for the provided DepartmentID
    SELECT EmployeeID, FirstName, LastName
    FROM Employee_New
    WHERE DepartmentID = @DepartmentID;
END;
```

-- call the procedure

EXEC GetEmployeeDetails @DepartmentID = 105;

Results Messages

	EmployeeID	FirstName	LastName
1	116	Nisha	Kumar
2	117	Deepak	Singh
3	118	Swati	Rana
4	137	Ishita	Raval
5	138	Anup	Saxena
6	139	Lakshmi	Shekar