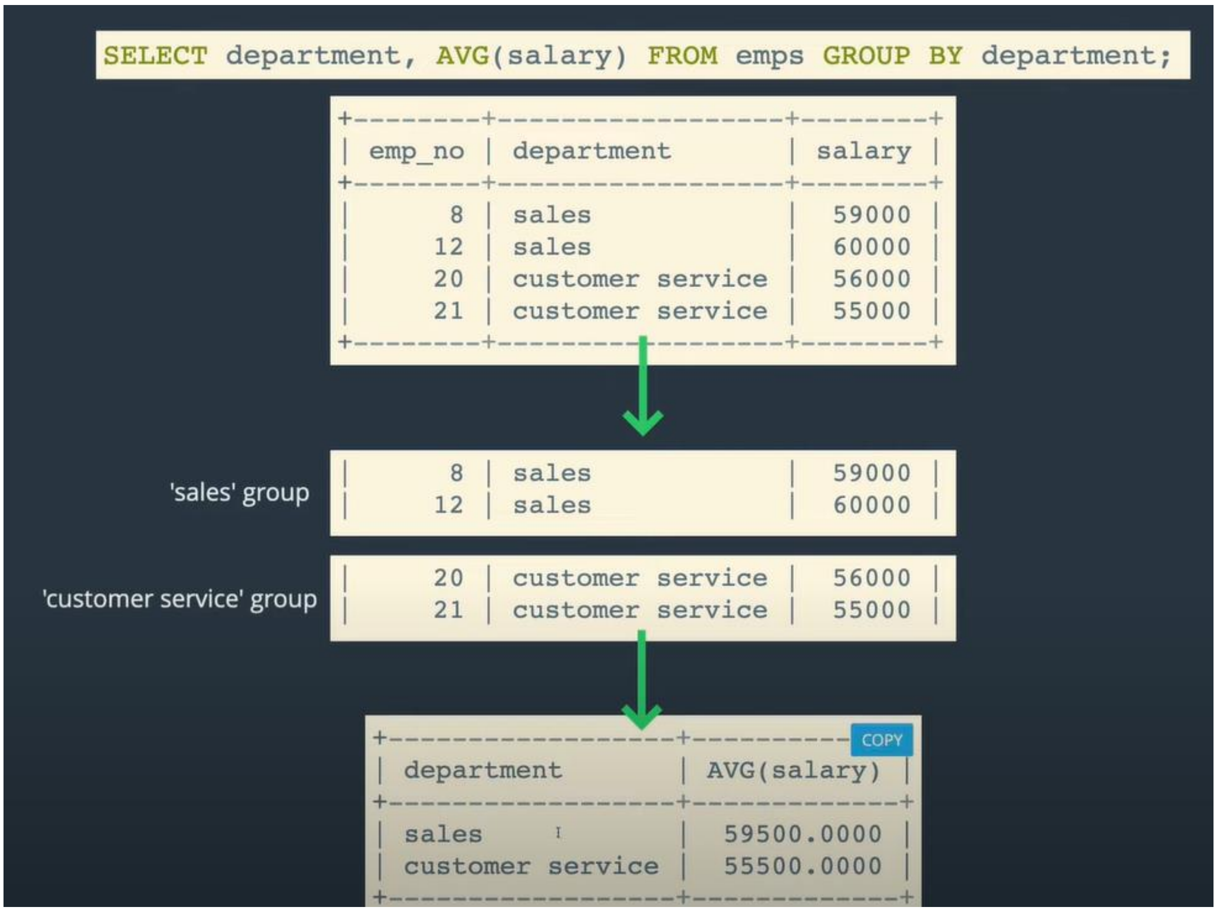
Group by collapses groups of rows into a single result row



```
SELECT department, AVG(salary) FROM emps GROUP BY department;
```

```
+--------+-------------------+--------+
| emp_no | department        | salary |
+--------+-------------------+--------+
|      8 | sales             |  59000 |
|     12 | sales             |  60000 |
|     20 | customer service  |  56000 |
|     21 | customer service  |  55000 |
+--------+-------------------+--------+
```

'sales' group

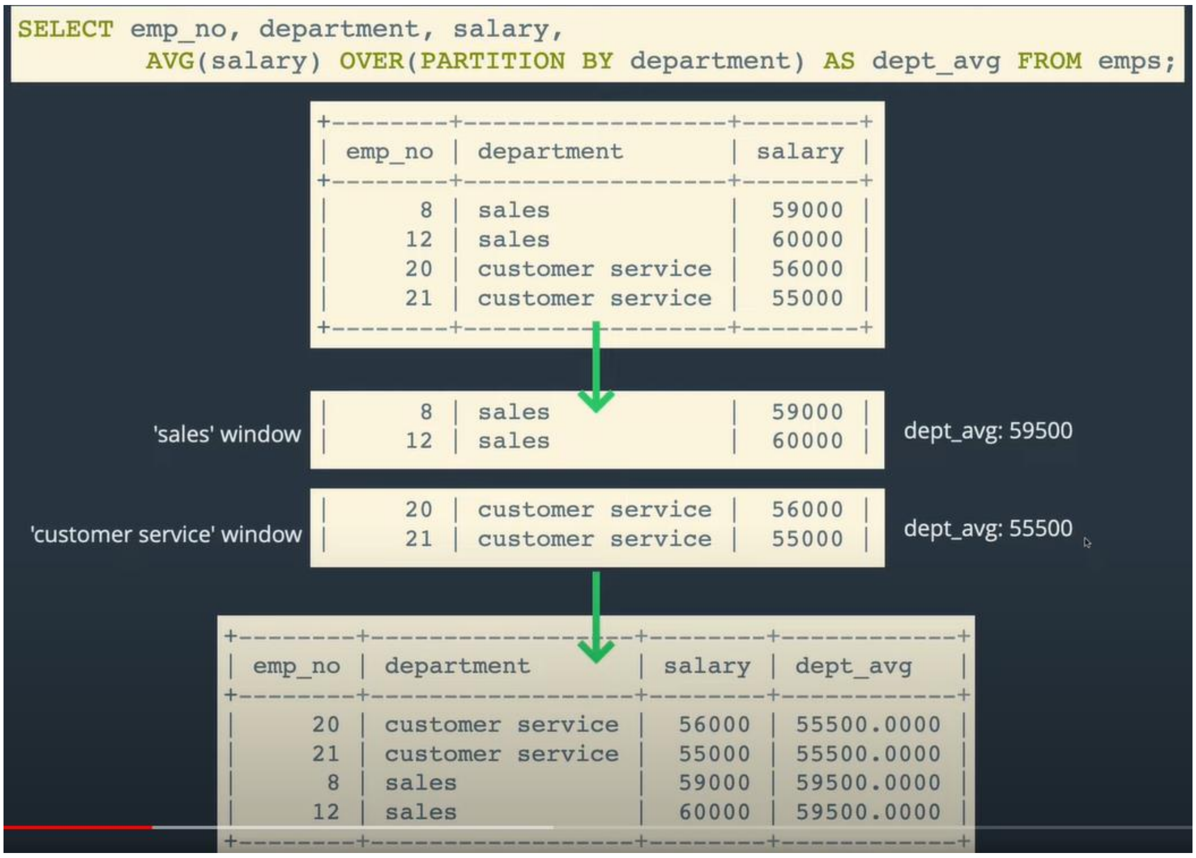
```
|      8 | sales             |  59000 |
|     12 | sales             |  60000 |
```

'customer service' group

```
|     20 | customer service  |  56000 |
|     21 | customer service  |  55000 |
```

```
+-------------------+-------------+
| department        | AVG(salary) |
+-------------------+-------------+
| sales             |  59500.0000 |
| customer service  |  55500.0000 |
+-------------------+-------------+
```

Window functions perform aggregate operations on groups of rows, but they produce a result for each row



```
SELECT emp_no, department, salary,
        AVG(salary) OVER(PARTITION BY department) AS dept_avg FROM emps;
```

```
+--------+-------------------+--------+
| emp_no | department        | salary |
+--------+-------------------+--------+
|      8 | sales             |  59000 |
|     12 | sales             |  60000 |
|     20 | customer service  |  56000 |
|     21 | customer service  |  55000 |
+--------+-------------------+--------+
```

'sales' window

```
|      8 | sales             |  59000 |
|     12 | sales             |  60000 |
```

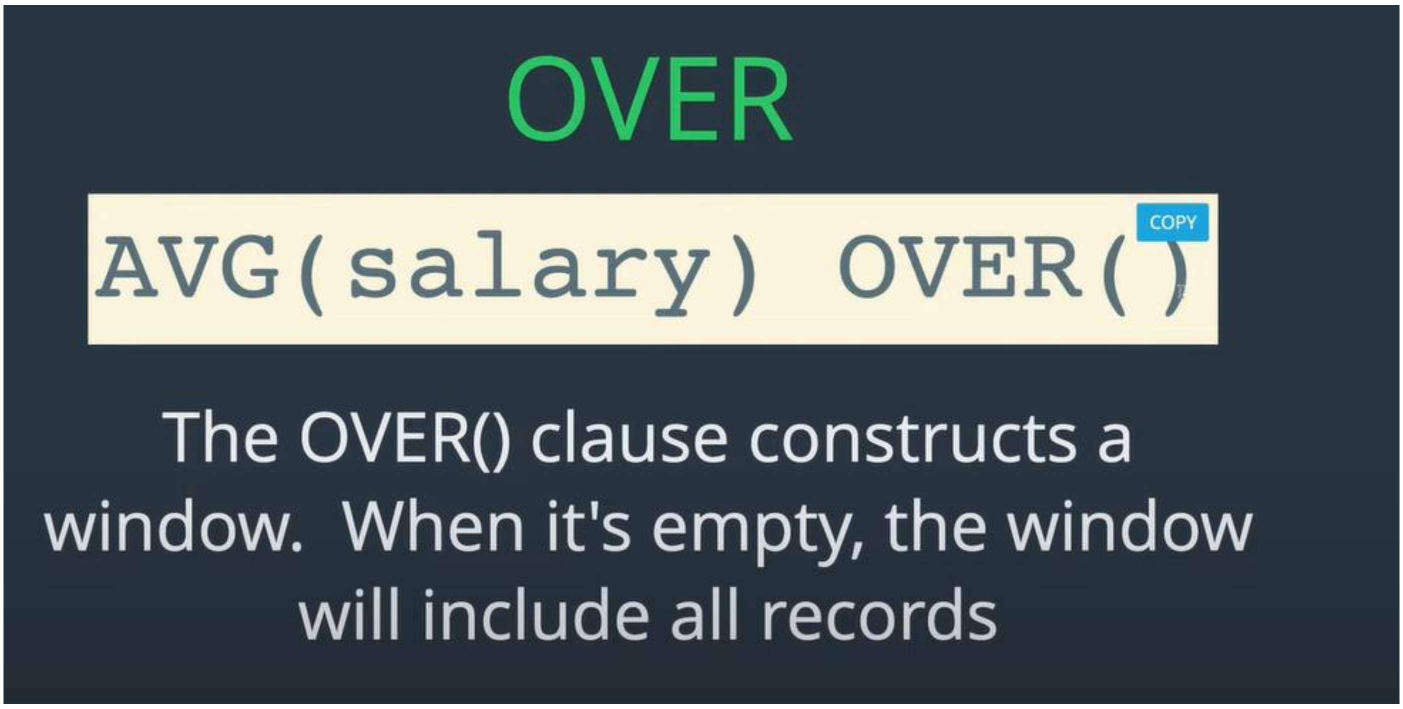
dept_avg: 59500

'customer service' window

```
|     20 | customer service  |  56000 |
|     21 | customer service  |  55000 |
```

dept_avg: 55500

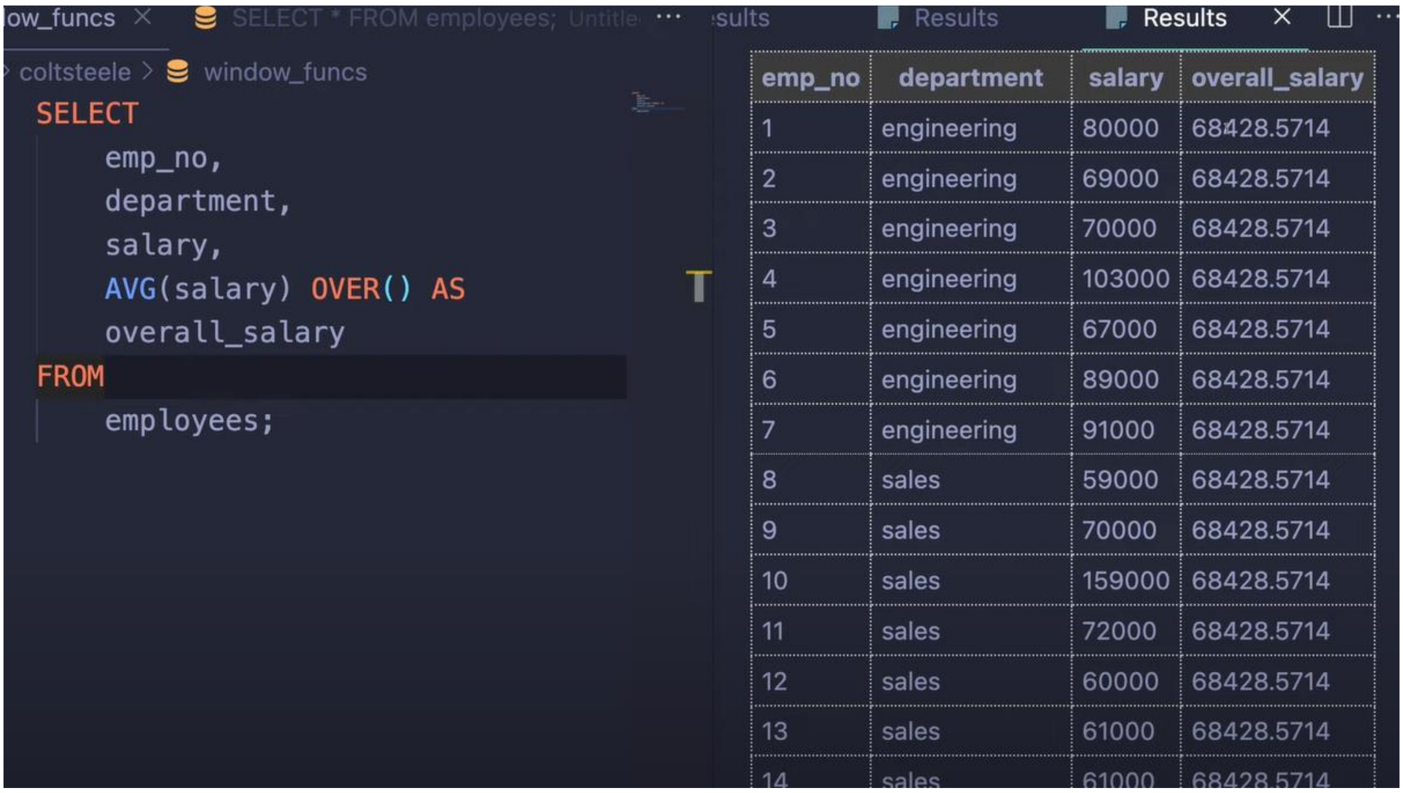
```
+--------+-------------------+--------+------------+
| emp_no | department        | salary | dept_avg   |
+--------+-------------------+--------+------------+
|     20 | customer service  |  56000 | 55500.0000 |
|     21 | customer service  |  55000 | 55500.0000 |
|      8 | sales             |  59000 | 59500.0000 |
|     12 | sales             |  60000 | 59500.0000 |
+--------+-------------------+--------+------------+
```



OVER

`AVG(salary) OVER()`

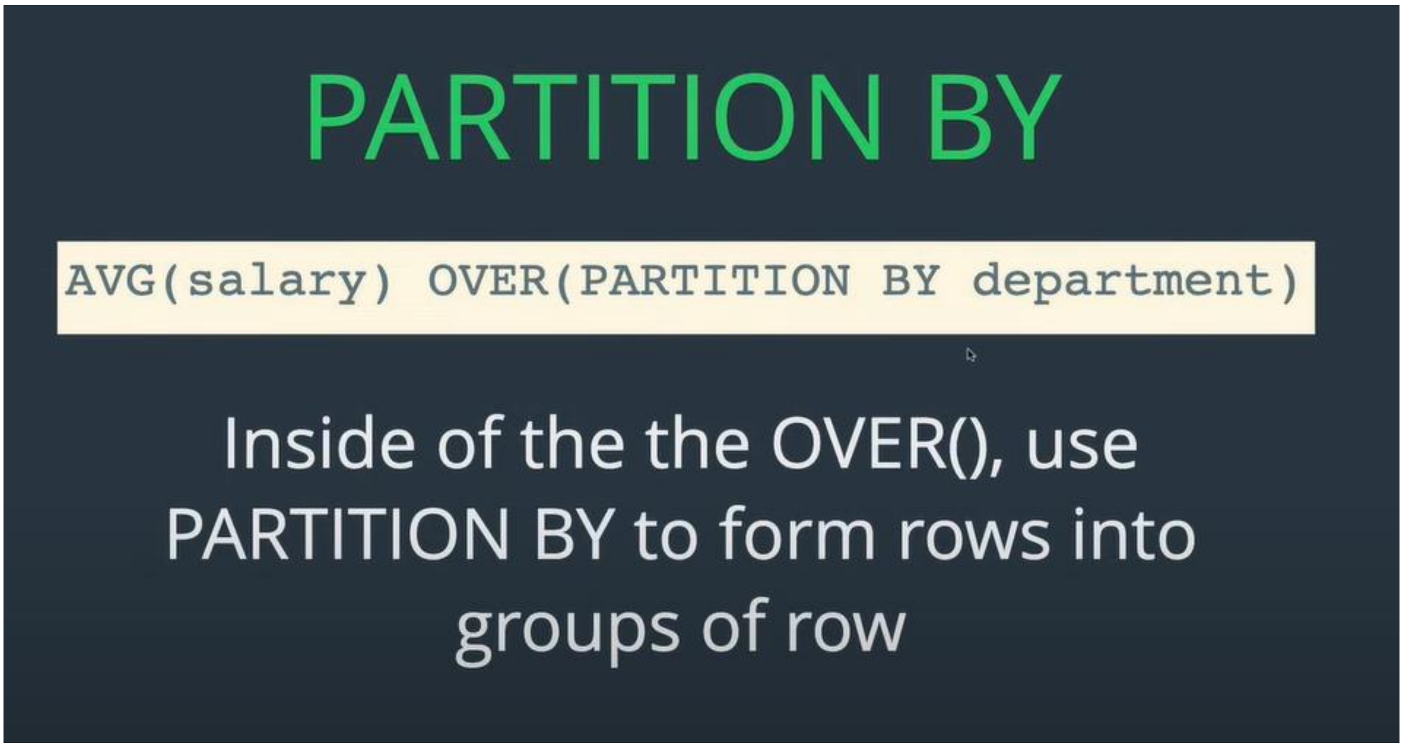
The OVER() clause constructs a window. When it's empty, the window will include all records



```sql
SELECT
    emp_no,
    department,
    salary,
    AVG(salary) OVER() AS
    overall_salary
FROM
    employees;
```

emp_no	department	salary	overall_salary
1	engineering	80000	68428.5714
2	engineering	69000	68428.5714
3	engineering	70000	68428.5714
4	engineering	103000	68428.5714
5	engineering	67000	68428.5714
6	engineering	89000	68428.5714
7	engineering	91000	68428.5714
8	sales	59000	68428.5714
9	sales	70000	68428.5714
10	sales	159000	68428.5714
11	sales	72000	68428.5714
12	sales	60000	68428.5714
13	sales	61000	68428.5714
14	sales	61000	68428.5714

- I can see that now the overal salary is the same for all the rows



PARTITION BY

```
AVG(salary) OVER(PARTITION BY department)
```

Inside of the the OVER(), use PARTITION BY to form rows into groups of row

- Instead of one massive window with all the rows and calculating the mean for them, I want to calculate the average for all the rows in each window (basically window = group) partitioned by department

Some functins can only be used as a window functions:

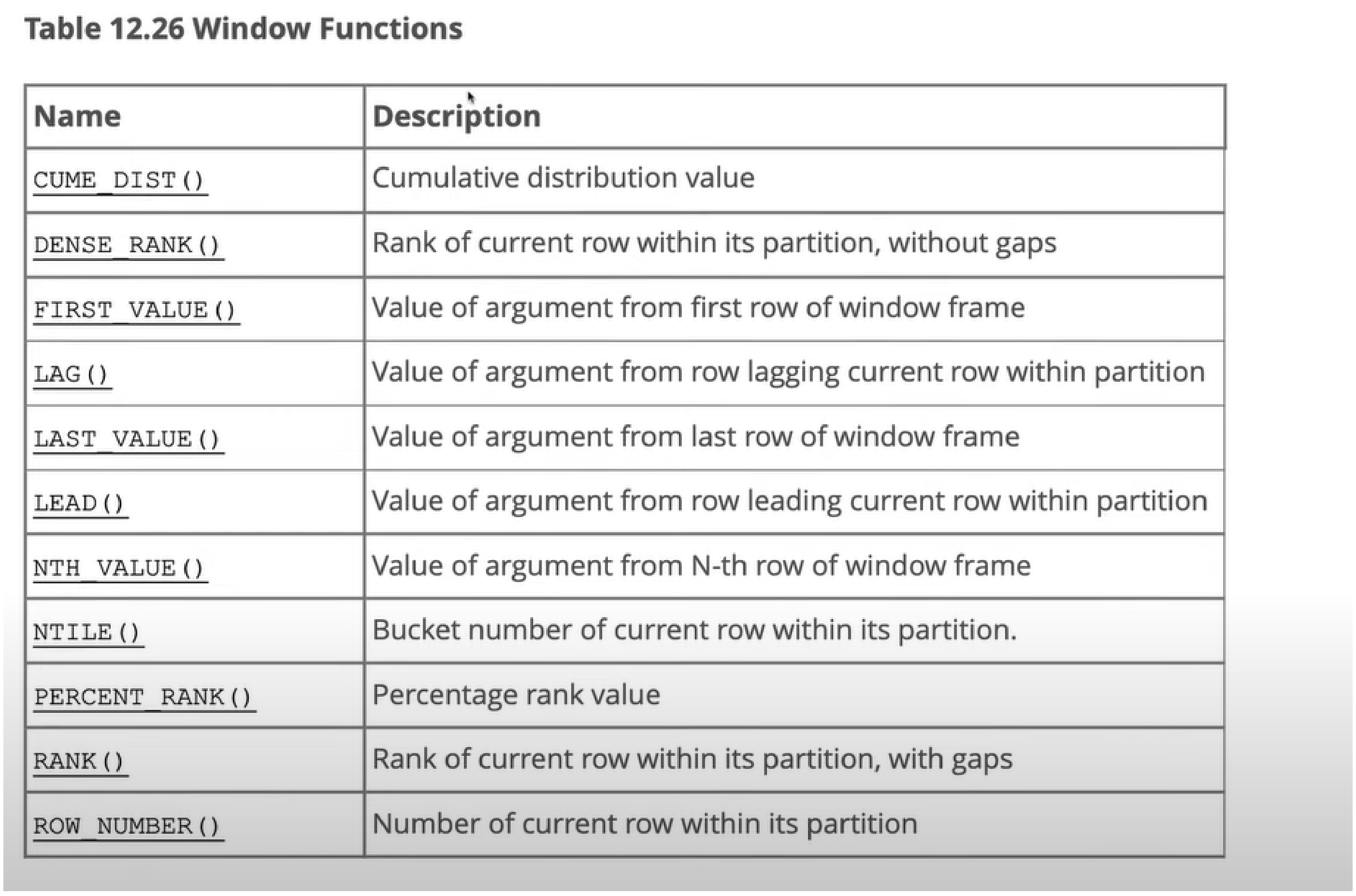
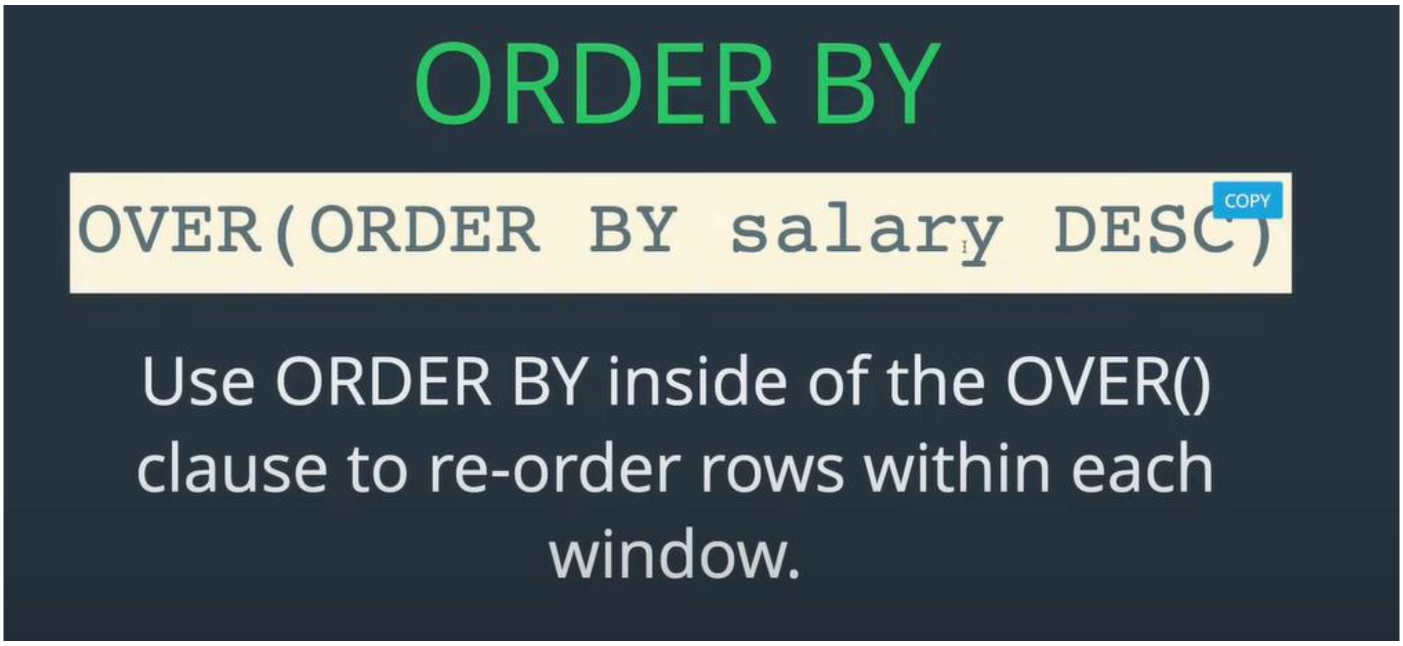


Table 12.26 Window Functions

Name	Description
`CUME_DIST()`	Cumulative distribution value
`DENSE_RANK()`	Rank of current row within its partition, without gaps
`FIRST_VALUE()`	Value of argument from first row of window frame
`LAG()`	Value of argument from row lagging current row within partition
`LAST_VALUE()`	Value of argument from last row of window frame
`LEAD()`	Value of argument from row leading current row within partition
`NTH_VALUE()`	Value of argument from N-th row of window frame
`NTILE()`	Bucket number of current row within its partition.
`PERCENT_RANK()`	Percentage rank value
`RANK()`	Rank of current row within its partition, with gaps
`ROW_NUMBER()`	Number of current row within its partition

Funkce RANK() – chci vyjádřit pořadí salary within a window (group) – I need to include new piece of syntax



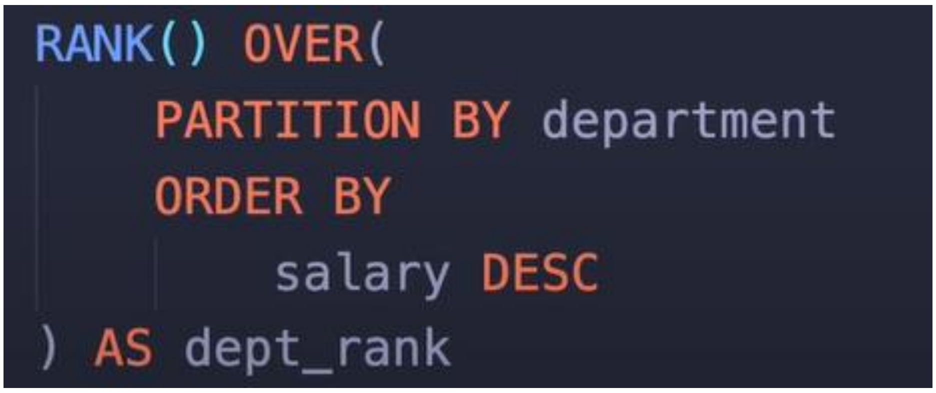
ORDER BY

```
OVER(ORDER BY salary DESC)
```

Use ORDER BY inside of the OVER() clause to re-order rows within each window.

- This gives me just one big window ordered by salary

If I want to also get the order within windows (groups) – here for example based on department column



```
RANK() OVER(
    PARTITION BY department
    ORDER BY
        salary DESC
) AS dept_rank
```

https://youtu.be/y1KCM8vbYe4?feature=shared