Методы сбора, хранения, обработки и анализа данных

Лекция 4

Оконные функции

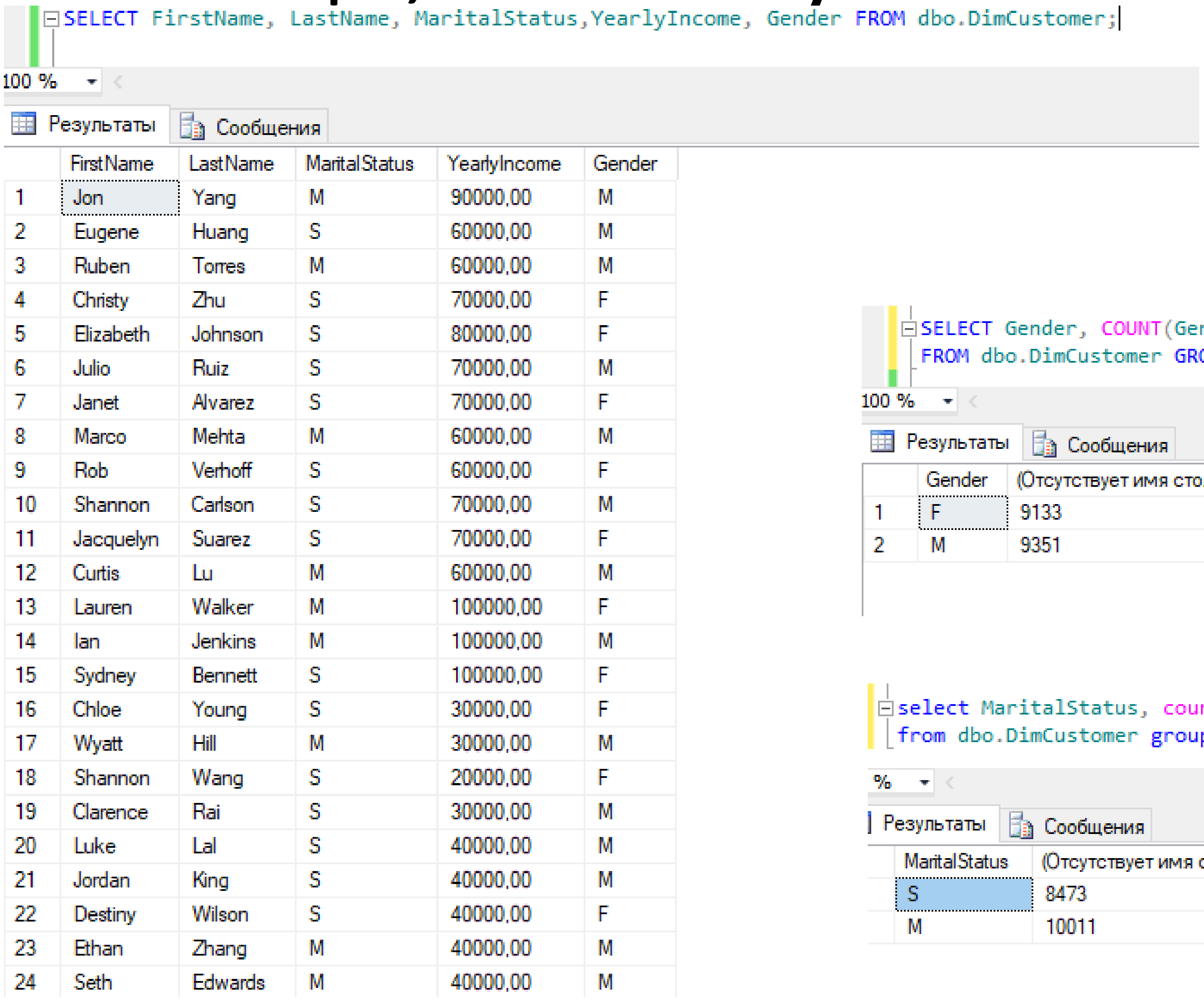
Обработка наборов строк

- Требуется:
  - соотносить строки с предыдущими или последующими строками
  - выделять группы строк, обрабатываемые независимо от других
- Низкая производительность
- Сложность

Типичные задачи

- Типичные задачи:
  - Подсчет промежуточной суммы
  - Подсчет процентов в группе
  - Запросы первых N
  - Подсчет скользящего среднего
  - Выполнение ранжирующих запросов

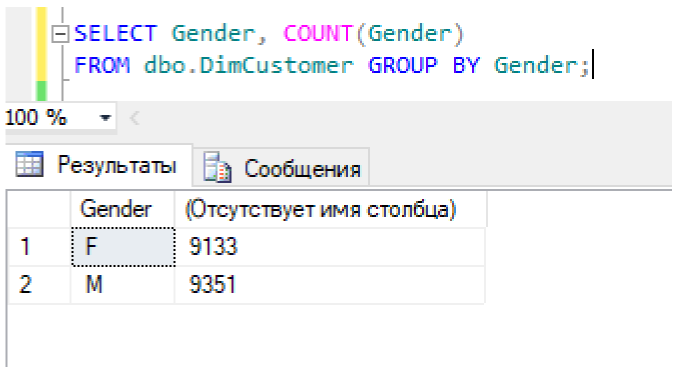
Таблицы, используемые в лекции



```
SELECT FirstName, LastName, MaritalStatus,YearlyIncome, Gender FROM dbo.DimCustomer;
```

Результаты | Сообщения

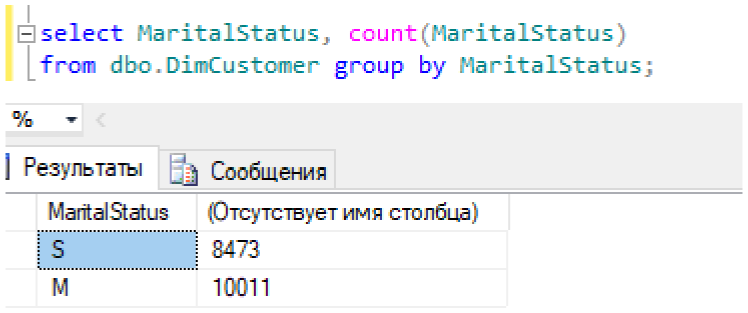
	FirstName	LastName	MaritalStatus	YearlyIncome	Gender
1	Jon	Yang	M	90000,00	M
2	Eugene	Huang	S	60000,00	M
3	Ruben	Torres	M	60000,00	M
4	Christy	Zhu	S	70000,00	F
5	Elizabeth	Johnson	S	80000,00	F
6	Julio	Ruiz	S	70000,00	M
7	Janet	Alvarez	S	70000,00	F
8	Marco	Mehta	M	60000,00	M
9	Rob	Verhoff	S	60000,00	F
10	Shannon	Carlson	S	70000,00	M
11	Jacquelyn	Suarez	S	70000,00	F
12	Curtis	Lu	M	60000,00	M
13	Lauren	Walker	M	100000,00	F
14	Ian	Jenkins	M	100000,00	M
15	Sydney	Bennett	S	100000,00	F
16	Chloe	Young	S	30000,00	F
17	Wyatt	Hill	M	30000,00	M
18	Shannon	Wang	S	20000,00	F
19	Clarence	Rai	S	30000,00	M
20	Luke	Lal	S	40000,00	M
21	Jordan	King	S	40000,00	M
22	Destiny	Wilson	S	40000,00	F
23	Ethan	Zhang	M	40000,00	M
24	Seth	Edwards	M	40000,00	M



```
SELECT Gender, COUNT(Gender)
FROM dbo.DimCustomer GROUP BY Gender;
```

Результаты | Сообщения

	Gender	(Отсутствует имя столбца)
1	F	9133
2	M	9351

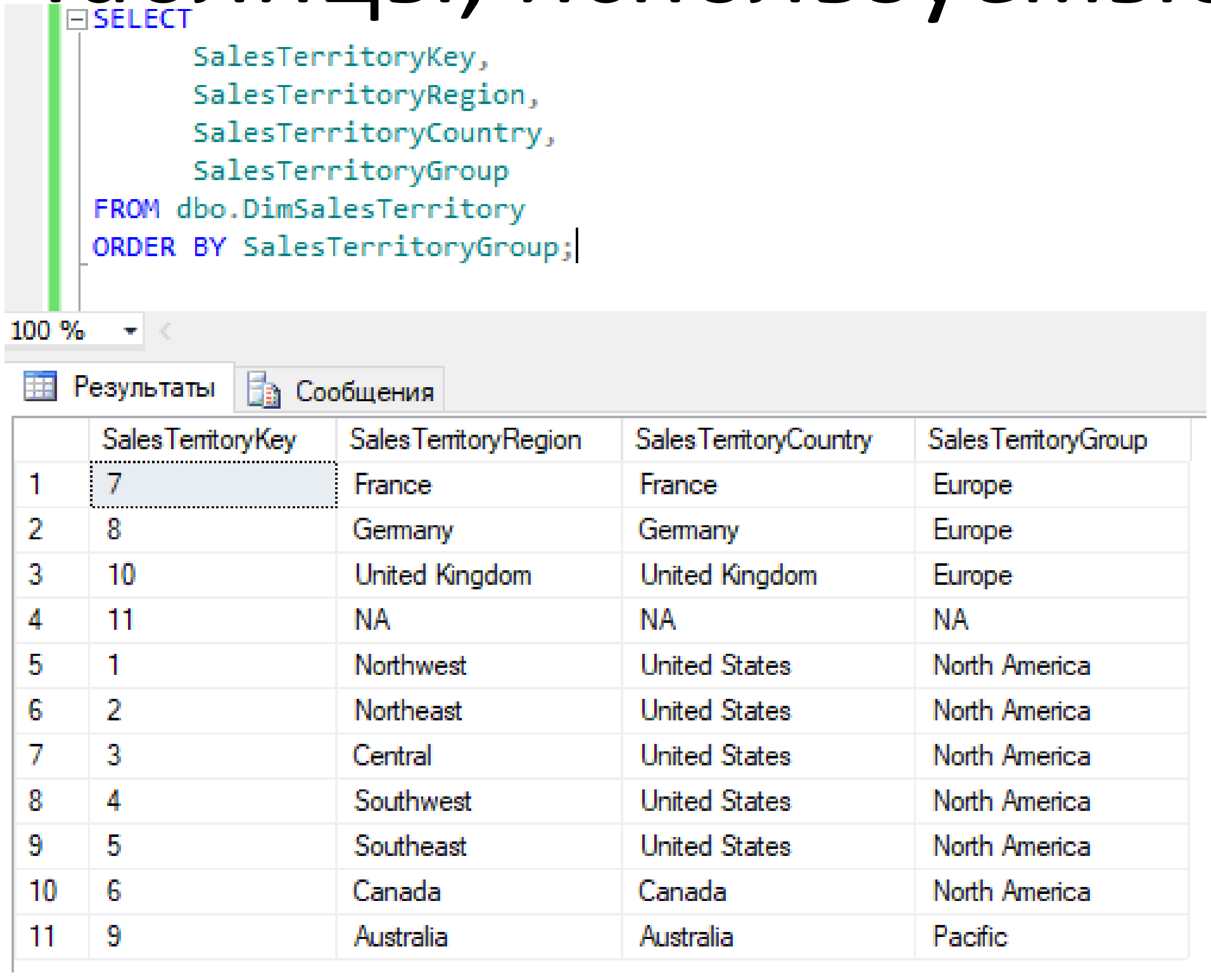


```
select MaritalStatus, count(MaritalStatus)
from dbo.DimCustomer group by MaritalStatus;
```

Результаты | Сообщения

MaritalStatus	(Отсутствует имя столбца)
S	8473
M	10011

Таблицы, используемые в лекции



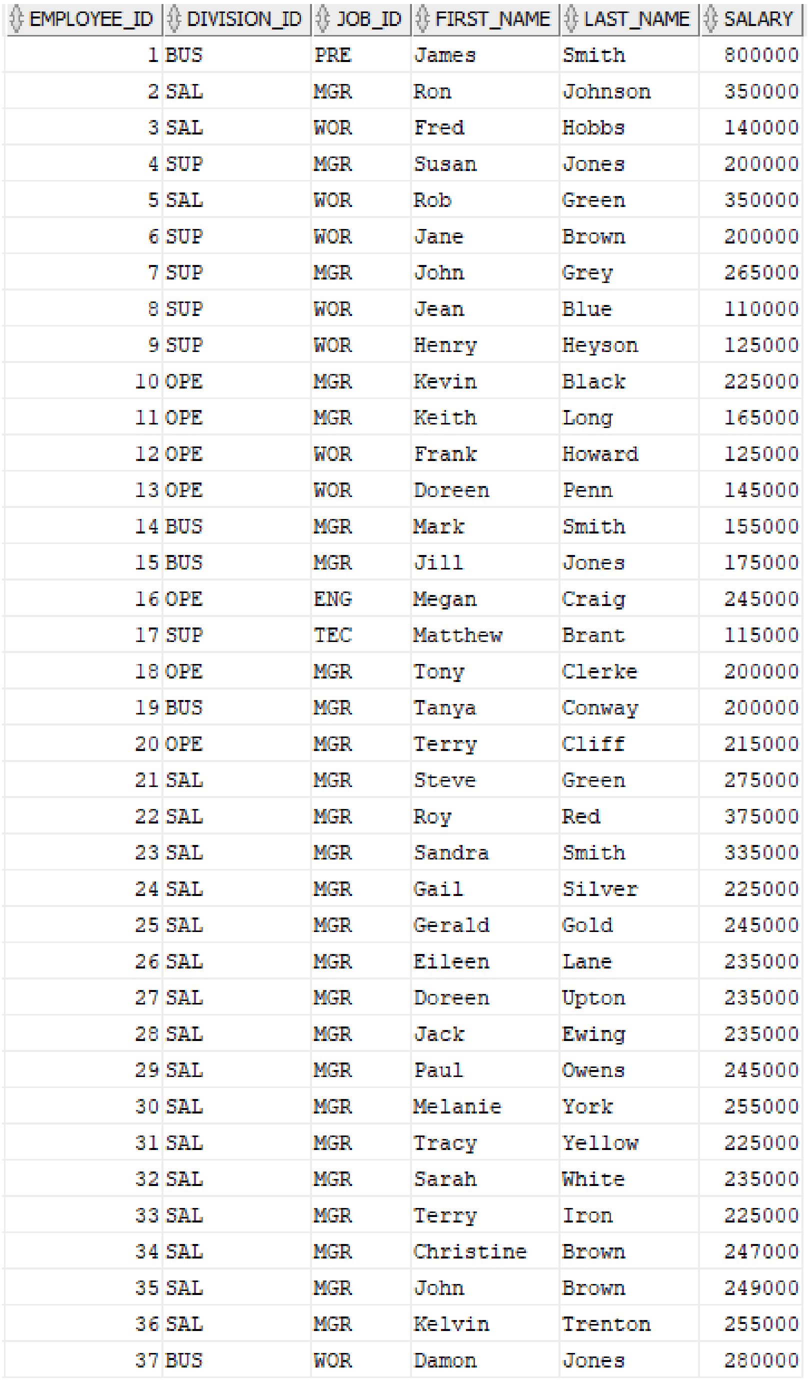
```
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup
FROM dbo.DimSalesTerritory
ORDER BY SalesTerritoryGroup;
```

100 %

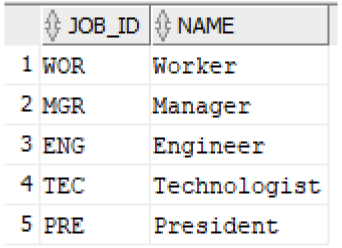
Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup
1	7	France	France	Europe
2	8	Germany	Germany	Europe
3	10	United Kingdom	United Kingdom	Europe
4	11	NA	NA	NA
5	1	Northwest	United States	North America
6	2	Northeast	United States	North America
7	3	Central	United States	North America
8	4	Southwest	United States	North America
9	5	Southeast	United States	North America
10	6	Canada	Canada	North America
11	9	Australia	Australia	Pacific

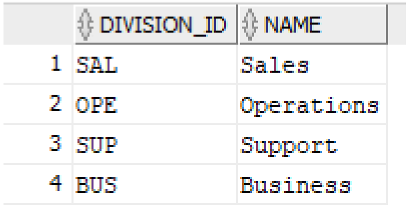
Таблицы, используемые в лекции



EMPLOYEE_ID	DIVISION_ID	JOB_ID	FIRST_NAME	LAST_NAME	SALARY
1	BUS	PRE	James	Smith	800000
2	SAL	MGR	Ron	Johnson	350000
3	SAL	WOR	Fred	Hobbs	140000
4	SUP	MGR	Susan	Jones	200000
5	SAL	WOR	Rob	Green	350000
6	SUP	WOR	Jane	Brown	200000
7	SUP	MGR	John	Grey	265000
8	SUP	WOR	Jean	Blue	110000
9	SUP	WOR	Henry	Heyson	125000
10	OPE	MGR	Kevin	Black	225000
11	OPE	MGR	Keith	Long	165000
12	OPE	WOR	Frank	Howard	125000
13	OPE	WOR	Doreen	Penn	145000
14	BUS	MGR	Mark	Smith	155000
15	BUS	MGR	Jill	Jones	175000
16	OPE	ENG	Megan	Craig	245000
17	SUP	TEC	Matthew	Brant	115000
18	OPE	MGR	Tony	Clerke	200000
19	BUS	MGR	Tanya	Conway	200000
20	OPE	MGR	Terry	Cliff	215000
21	SAL	MGR	Steve	Green	275000
22	SAL	MGR	Roy	Red	375000
23	SAL	MGR	Sandra	Smith	335000
24	SAL	MGR	Gail	Silver	225000
25	SAL	MGR	Gerald	Gold	245000
26	SAL	MGR	Eileen	Lane	235000
27	SAL	MGR	Doreen	Upton	235000
28	SAL	MGR	Jack	Ewing	235000
29	SAL	MGR	Paul	Owens	245000
30	SAL	MGR	Melanie	York	255000
31	SAL	MGR	Tracy	Yellow	225000
32	SAL	MGR	Sarah	White	235000
33	SAL	MGR	Terry	Iron	225000
34	SAL	MGR	Christine	Brown	247000
35	SAL	MGR	John	Brown	249000
36	SAL	MGR	Kelvin	Trenton	255000
37	BUS	WOR	Damon	Jones	280000



	JOB_ID	NAME
1	WOR	Worker
2	MGR	Manager
3	ENG	Engineer
4	TEC	Technologist
5	PRE	President



	DIVISION_ID	NAME
1	SAL	Sales
2	OPE	Operations
3	SUP	Support
4	BUS	Business

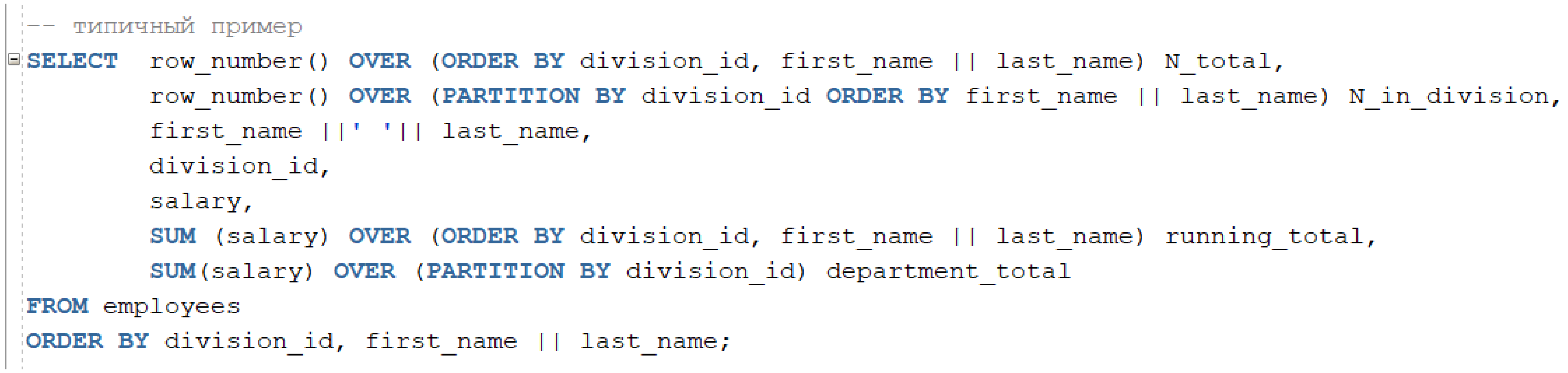
Оконные функции

- Функции, которые позволяют осуществлять вычисления в заданном диапазоне строк внутри одного предложения SELECT

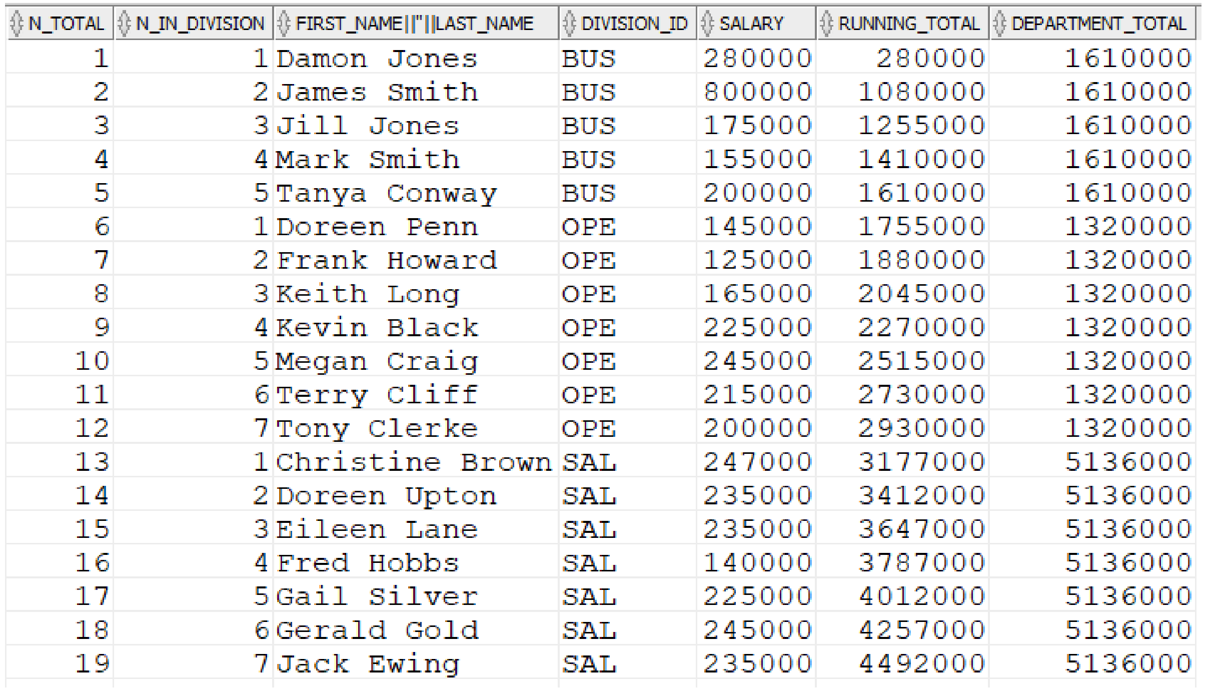
Понятие окна

- Окно – набор строк, в рамках которого происходит вычисление
- Оконная функция позволяет разбивать весь набор данных на окна
- Основное преимущество - оконные функции не приводят к группированию строк
- Строки сохраняют идентификаторы, а агрегированное значение добавляется к каждой строке

Типичный синтаксис



```
-- типичный пример
SELECT  row_number() OVER (ORDER BY division_id, first_name || last_name) N_total,
        row_number() OVER (PARTITION BY division_id ORDER BY first_name || last_name) N_in_division,
        first_name ||' '|| last_name,
        division_id,
        salary,
        SUM (salary) OVER (ORDER BY division_id, first_name || last_name) running_total,
        SUM(salary) OVER (PARTITION BY division_id) department_total
FROM employees
ORDER BY division_id, first_name || last_name;
```



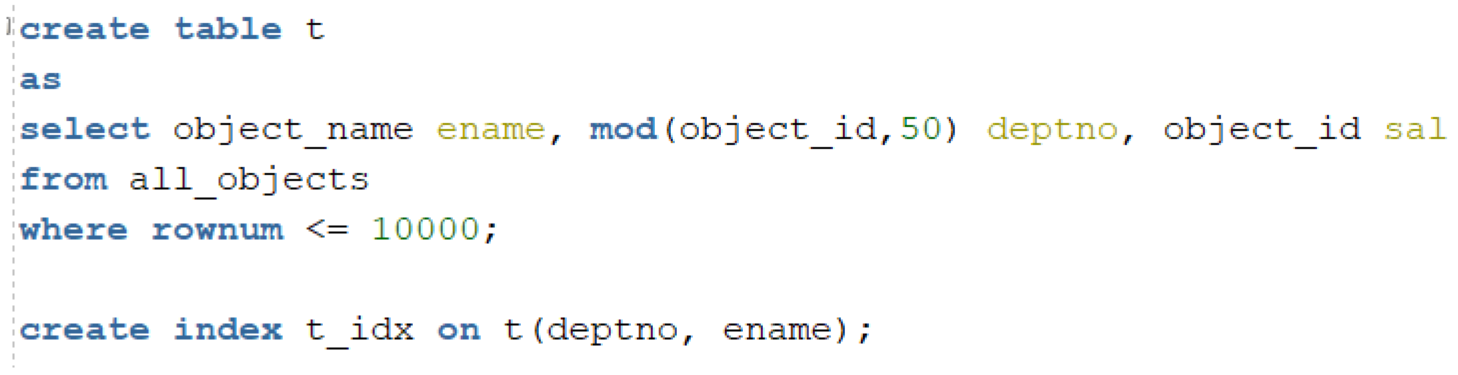
N_TOTAL	N_IN_DIVISION	FIRST_NAME||''||LAST_NAME	DIVISION_ID	SALARY	RUNNING_TOTAL	DEPARTMENT_TOTAL
1	1	Damon Jones	BUS	280000	280000	1610000
2	2	James Smith	BUS	800000	1080000	1610000
3	3	Jill Jones	BUS	175000	1255000	1610000
4	4	Mark Smith	BUS	155000	1410000	1610000
5	5	Tanya Conway	BUS	200000	1610000	1610000
6	1	Doreen Penn	OPE	145000	1755000	1320000
7	2	Frank Howard	OPE	125000	1880000	1320000
8	3	Keith Long	OPE	165000	2045000	1320000
9	4	Kevin Black	OPE	225000	2270000	1320000
10	5	Megan Craig	OPE	245000	2515000	1320000
11	6	Terry Cliff	OPE	215000	2730000	1320000
12	7	Tony Clerke	OPE	200000	2930000	1320000
13	1	Christine Brown	SAL	247000	3177000	5136000
14	2	Doreen Upton	SAL	235000	3412000	5136000
15	3	Eileen Lane	SAL	235000	3647000	5136000
16	4	Fred Hobbs	SAL	140000	3787000	5136000
17	5	Gail Silver	SAL	225000	4012000	5136000
18	6	Gerald Gold	SAL	245000	4257000	5136000
19	7	Jack Ewing	SAL	235000	4492000	5136000

Типичный синтаксис

- **Функция – SUM()**
- **Аргумент - SALARY**
- **OVER** – срез данных
- **PARTITION BY** – фрагментация
- **ORDER BY –** сортировка в данном фрагменте
- **ROWS** или **RANGE** – выражение для ограничения окна в пределах фрагмента

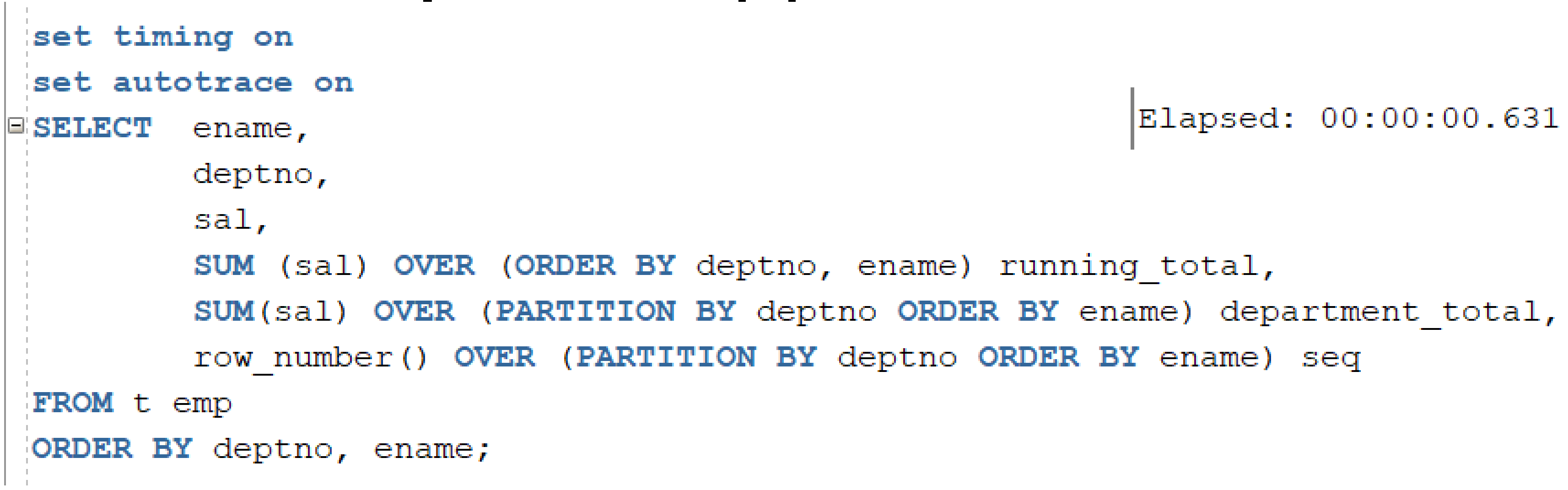
Производительность

- Тест производительности
- Запрос при помощи оконных функций и при помощи подзапросов
- 10 000 строк
- Время выполнения
- Планы запросов



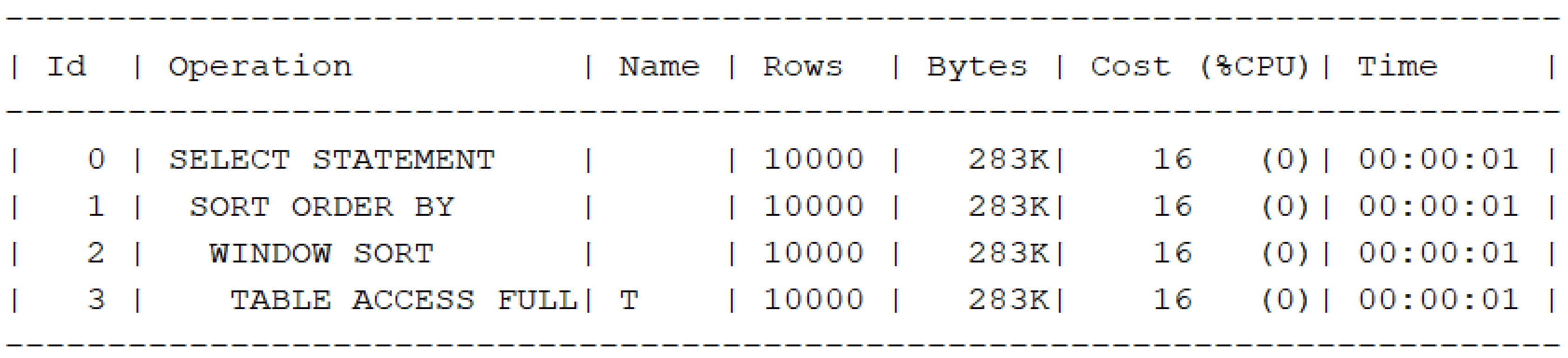
```
create table t
as
select object_name ename, mod(object_id,50) deptno, object_id sal
from all_objects
where rownum <= 10000;

create index t_idx on t(deptno, ename);
```

Производительность



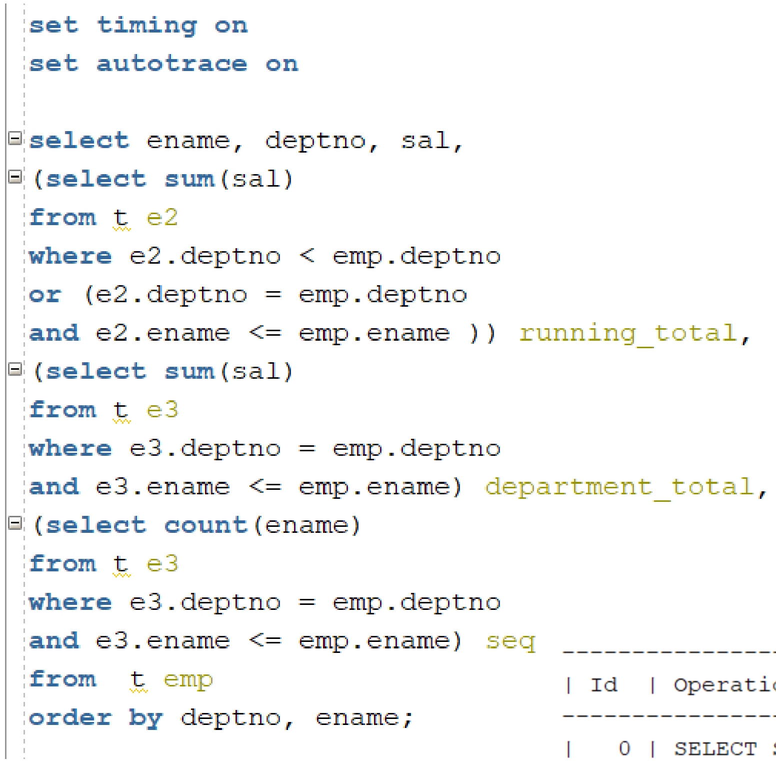
```
set timing on
set autotrace on
SELECT  ename,
        deptno,
        sal,
        SUM (sal) OVER (ORDER BY deptno, ename) running_total,
        SUM(sal) OVER (PARTITION BY deptno ORDER BY ename) department_total,
        row_number() OVER (PARTITION BY deptno ORDER BY ename) seq
FROM t emp
ORDER BY deptno, ename;
```

```
Elapsed: 00:00:00.631
```

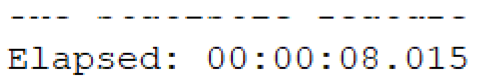


```
-----------------------------------------------------------------------------
| Id  | Operation             | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |      | 10000 |   283K|    16   (0)| 00:00:01 |
|   1 |  SORT ORDER BY        |      | 10000 |   283K|    16   (0)| 00:00:01 |
|   2 |   WINDOW SORT         |      | 10000 |   283K|    16   (0)| 00:00:01 |
|   3 |    TABLE ACCESS FULL| T    | 10000 |   283K|    16   (0)| 00:00:01 |
-----------------------------------------------------------------------------
```

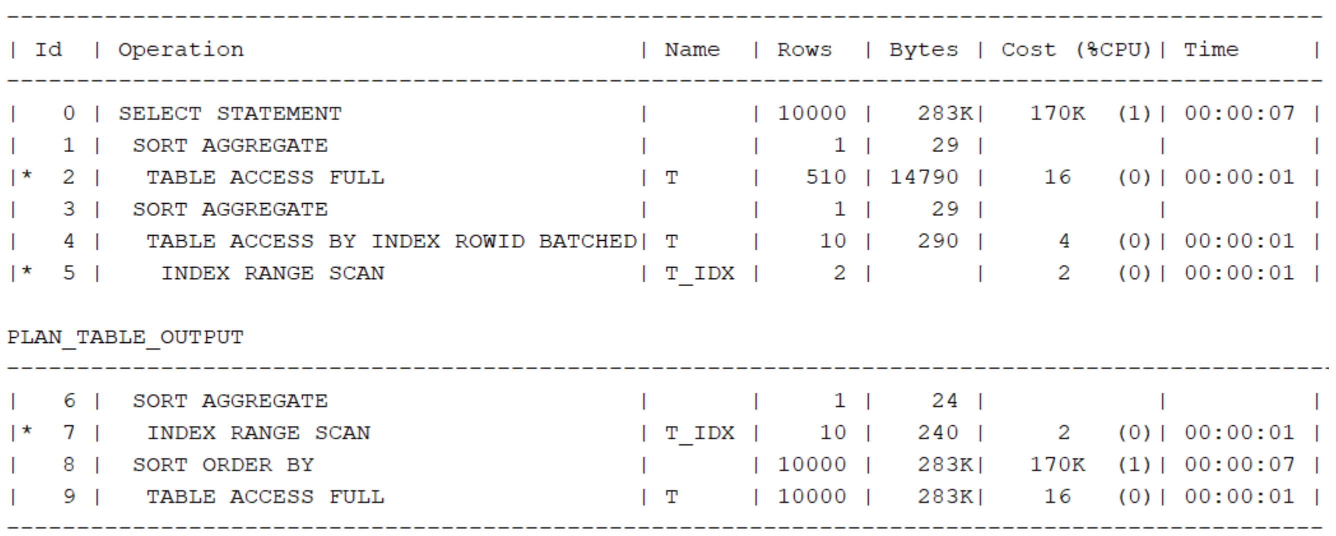
Производительность



```
set timing on
set autotrace on

select ename, deptno, sal,
(select sum(sal)
from t e2
where e2.deptno < emp.deptno
or (e2.deptno = emp.deptno
and e2.ename <= emp.ename )) running_total,
(select sum(sal)
from t e3
where e3.deptno = emp.deptno
and e3.ename <= emp.ename) department_total,
(select count(ename)
from t e3
where e3.deptno = emp.deptno
and e3.ename <= emp.ename) seq
from  t emp
order by deptno, ename;
```



```
Elapsed: 00:00:08.015
```



```
---------------------------------------------------------------------------------------------
| Id  | Operation                            | Name  | Rows  | Bytes | Cost (%CPU)| Time     |
---------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                     |       | 10000 |  283K|   170K   (1)| 00:00:07 |
|   1 |  SORT AGGREGATE                      |       |     1 |   29 |             |          |
|*  2 |   TABLE ACCESS FULL                  | T     |   510 | 14790 |    16   (0)| 00:00:01 |
|   3 |  SORT AGGREGATE                      |       |     1 |   29 |             |          |
|   4 |   TABLE ACCESS BY INDEX ROWID BATCHED| T     |    10 |  290 |     4   (0)| 00:00:01 |
|*  5 |    INDEX RANGE SCAN                  | T_IDX |     2 |       |     2   (0)| 00:00:01 |

PLAN_TABLE_OUTPUT
---------------------------------------------------------------------------------------------
|   6 |  SORT AGGREGATE                      |       |     1 |   24 |             |          |
|*  7 |   INDEX RANGE SCAN                   | T_IDX |    10 |  240 |     2   (0)| 00:00:01 |
|   8 |  SORT ORDER BY                       |       | 10000 |  283K|   170K   (1)| 00:00:07 |
|   9 |   TABLE ACCESS FULL                  | T     | 10000 |  283K|    16   (0)| 00:00:01 |
---------------------------------------------------------------------------------------------
```

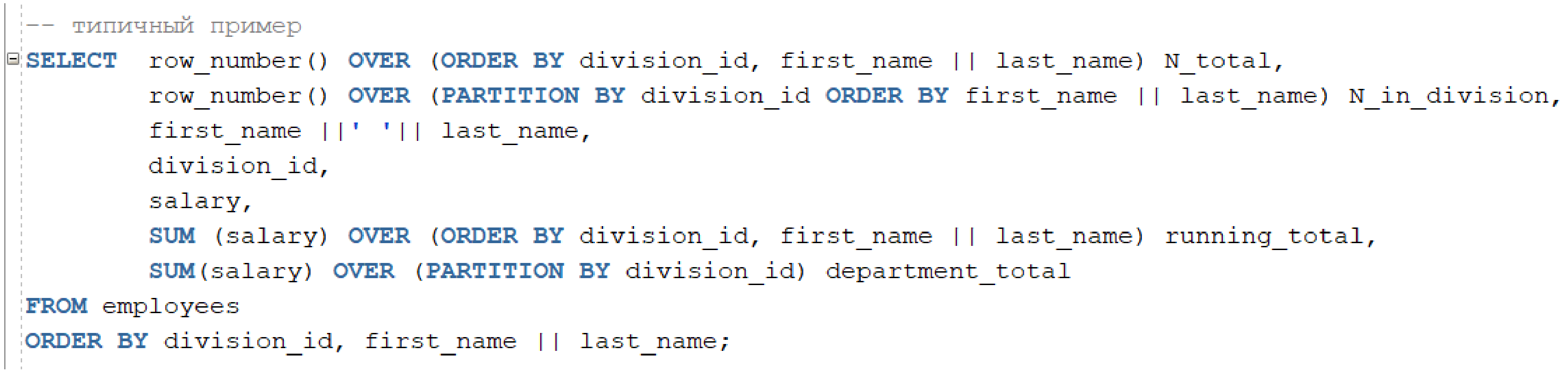
OVER

- Задает область, над которой будут производиться вычисления
- Можно дополнительно ограничить
- Расширить нельзя
- Для каждой функции может задаваться независимо

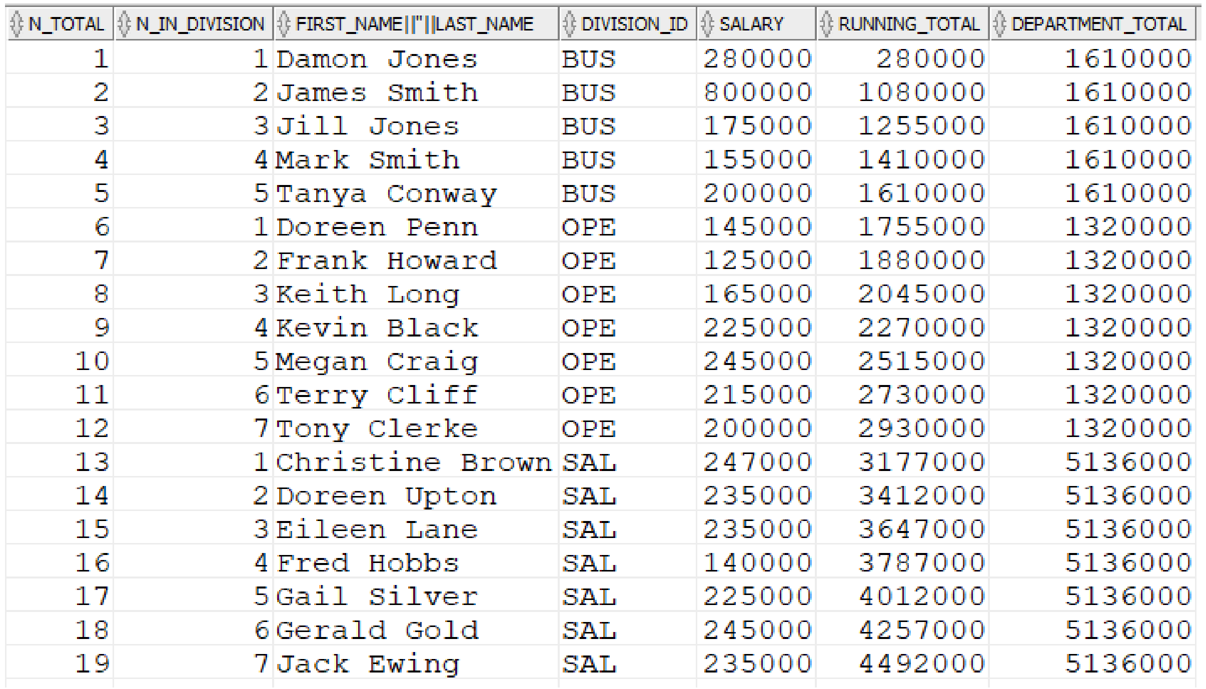
PARTITION

- Разделяет область действия на фрагменты (группы)
- В один фрагмент попадают строки с одинаковыми значениями полей

Пример



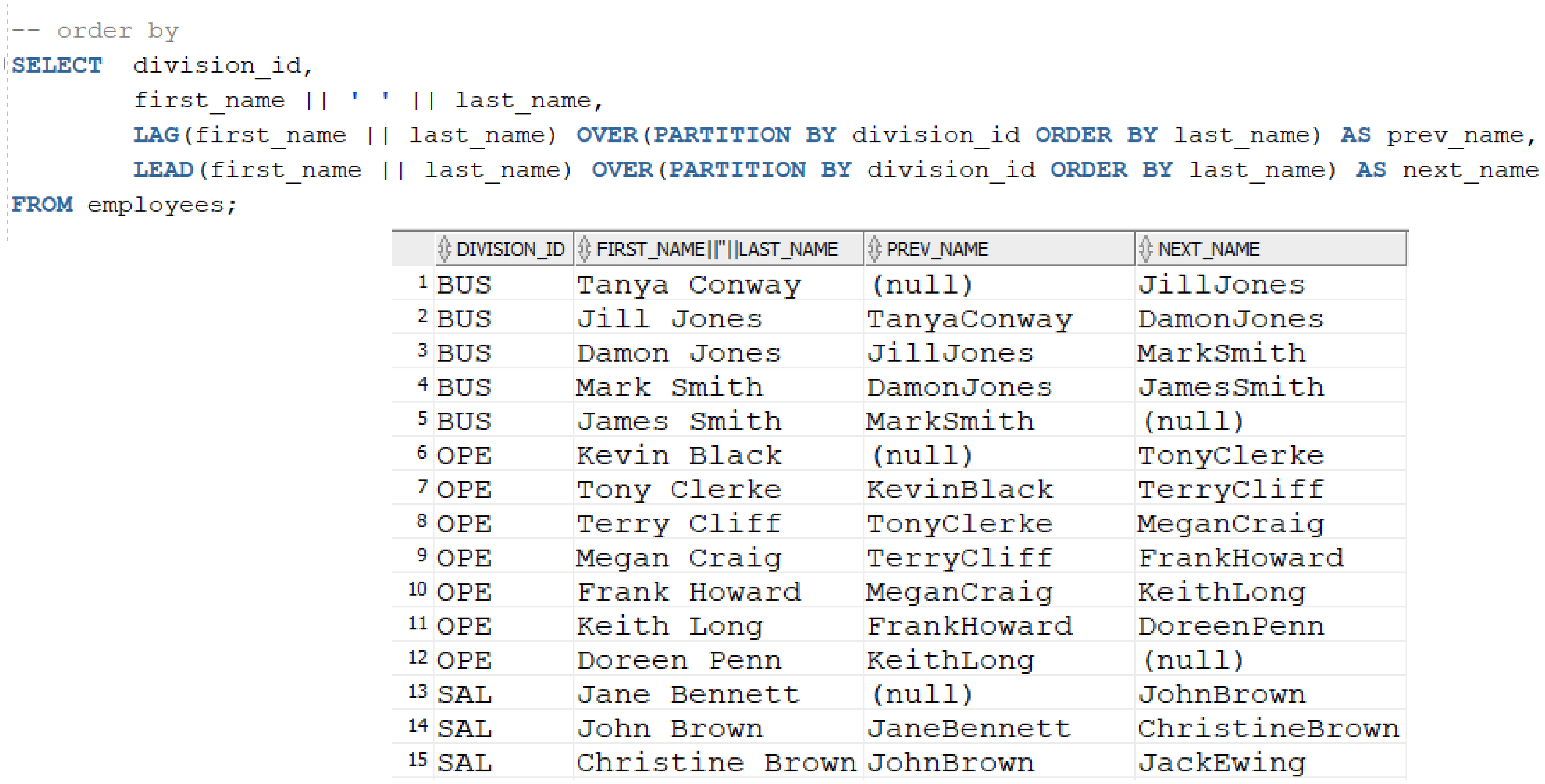
```
-- типичный пример
SELECT  row_number() OVER (ORDER BY division_id, first_name || last_name) N_total,
        row_number() OVER (PARTITION BY division_id ORDER BY first_name || last_name) N_in_division,
        first_name ||' '|| last_name,
        division_id,
        salary,
        SUM (salary) OVER (ORDER BY division_id, first_name || last_name) running_total,
        SUM(salary) OVER (PARTITION BY division_id) department_total
FROM employees
ORDER BY division_id, first_name || last_name;
```



N_TOTAL	N_IN_DIVISION	FIRST_NAME||''||LAST_NAME	DIVISION_ID	SALARY	RUNNING_TOTAL	DEPARTMENT_TOTAL
1	1	Damon Jones	BUS	280000	280000	1610000
2	2	James Smith	BUS	800000	1080000	1610000
3	3	Jill Jones	BUS	175000	1255000	1610000
4	4	Mark Smith	BUS	155000	1410000	1610000
5	5	Tanya Conway	BUS	200000	1610000	1610000
6	1	Doreen Penn	OPE	145000	1755000	1320000
7	2	Frank Howard	OPE	125000	1880000	1320000
8	3	Keith Long	OPE	165000	2045000	1320000
9	4	Kevin Black	OPE	225000	2270000	1320000
10	5	Megan Craig	OPE	245000	2515000	1320000
11	6	Terry Cliff	OPE	215000	2730000	1320000
12	7	Tony Clerke	OPE	200000	2930000	1320000
13	1	Christine Brown	SAL	247000	3177000	5136000
14	2	Doreen Upton	SAL	235000	3412000	5136000
15	3	Eileen Lane	SAL	235000	3647000	5136000
16	4	Fred Hobbs	SAL	140000	3787000	5136000
17	5	Gail Silver	SAL	225000	4012000	5136000
18	6	Gerald Gold	SAL	245000	4257000	5136000
19	7	Jack Ewing	SAL	235000	4492000	5136000

ORDER BY

- Определяет сортировку в группе

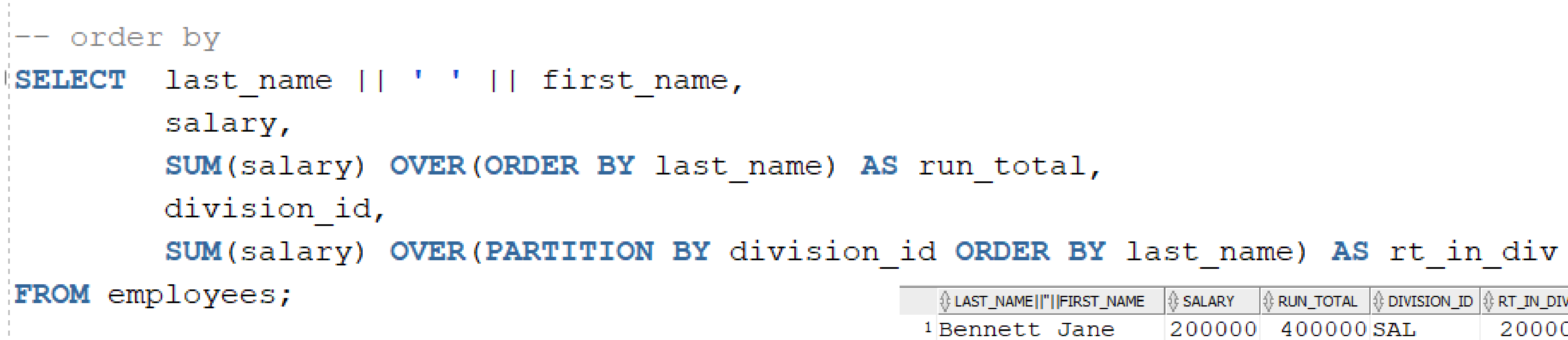


```sql
-- order by
SELECT  division_id,
        first_name || ' ' || last_name,
        LAG(first_name || last_name) OVER(PARTITION BY division_id ORDER BY last_name) AS prev_name,
        LEAD(first_name || last_name) OVER(PARTITION BY division_id ORDER BY last_name) AS next_name
FROM employees;
```

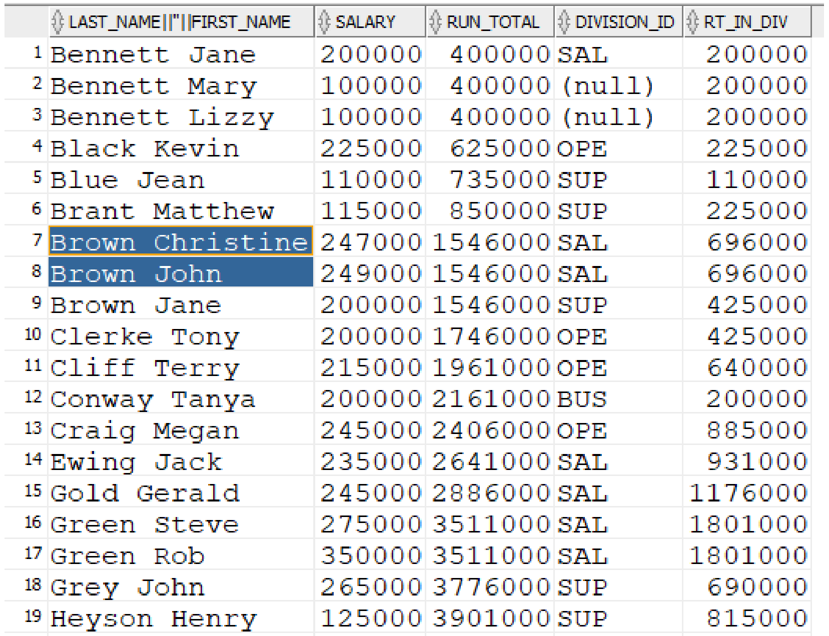
	DIVISION_ID	FIRST_NAME||''||LAST_NAME	PREV_NAME	NEXT_NAME
1	BUS	Tanya Conway	(null)	JillJones
2	BUS	Jill Jones	TanyaConway	DamonJones
3	BUS	Damon Jones	JillJones	MarkSmith
4	BUS	Mark Smith	DamonJones	JamesSmith
5	BUS	James Smith	MarkSmith	(null)
6	OPE	Kevin Black	(null)	TonyClerke
7	OPE	Tony Clerke	KevinBlack	TerryCliff
8	OPE	Terry Cliff	TonyClerke	MeganCraig
9	OPE	Megan Craig	TerryCliff	FrankHoward
10	OPE	Frank Howard	MeganCraig	KeithLong
11	OPE	Keith Long	FrankHoward	DoreenPenn
12	OPE	Doreen Penn	KeithLong	(null)
13	SAL	Jane Bennett	(null)	JohnBrown
14	SAL	John Brown	JaneBennett	ChristineBrown
15	SAL	Christine Brown	JohnBrown	JackEwing

ORDER BY

- В качестве окна использует строки от первой до текущей



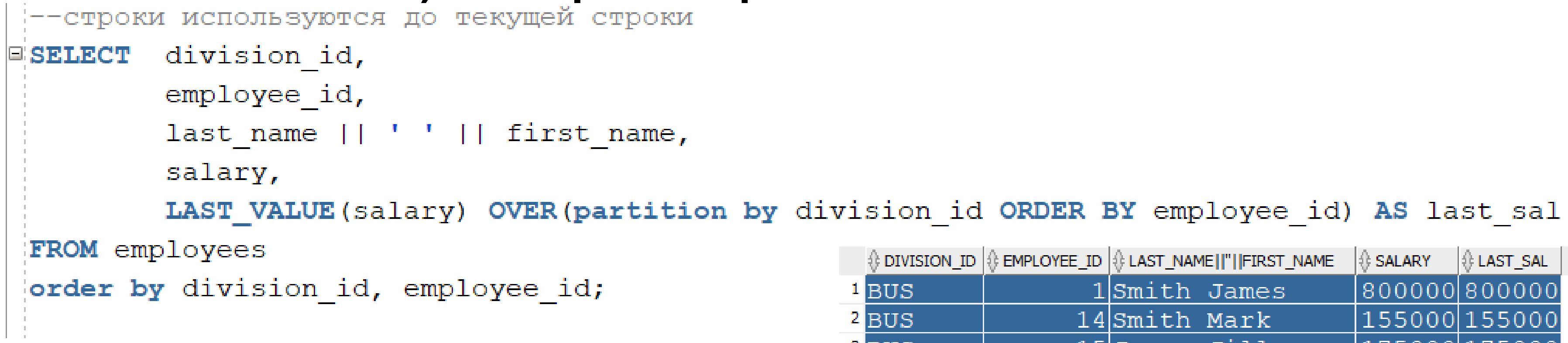
```
-- order by
SELECT  last_name || ' ' || first_name,
        salary,
        SUM(salary) OVER(ORDER BY last_name) AS run_total,
        division_id,
        SUM(salary) OVER(PARTITION BY division_id ORDER BY last_name) AS rt_in_div
FROM employees;
```



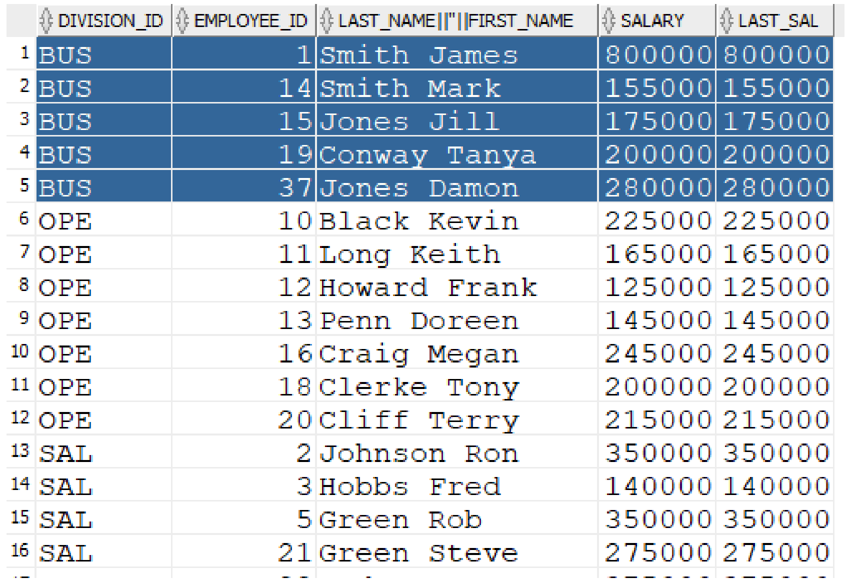
	LAST_NAME||''||FIRST_NAME	SALARY	RUN_TOTAL	DIVISION_ID	RT_IN_DIV
1	Bennett Jane	200000	400000	SAL	200000
2	Bennett Mary	100000	400000	(null)	200000
3	Bennett Lizzy	100000	400000	(null)	200000
4	Black Kevin	225000	625000	OPE	225000
5	Blue Jean	110000	735000	SUP	110000
6	Brant Matthew	115000	850000	SUP	225000
7	Brown Christine	247000	1546000	SAL	696000
8	Brown John	249000	1546000	SAL	696000
9	Brown Jane	200000	1546000	SUP	425000
10	Clerke Tony	200000	1746000	OPE	425000
11	Cliff Terry	215000	1961000	OPE	640000
12	Conway Tanya	200000	2161000	BUS	200000
13	Craig Megan	245000	2406000	OPE	885000
14	Ewing Jack	235000	2641000	SAL	931000
15	Gold Gerald	245000	2886000	SAL	1176000
16	Green Steve	275000	3511000	SAL	1801000
17	Green Rob	350000	3511000	SAL	1801000
18	Grey John	265000	3776000	SUP	690000
19	Heyson Henry	125000	3901000	SUP	815000

ORDER BY

- Сужение диапазона может приводить к ошибкам, например:

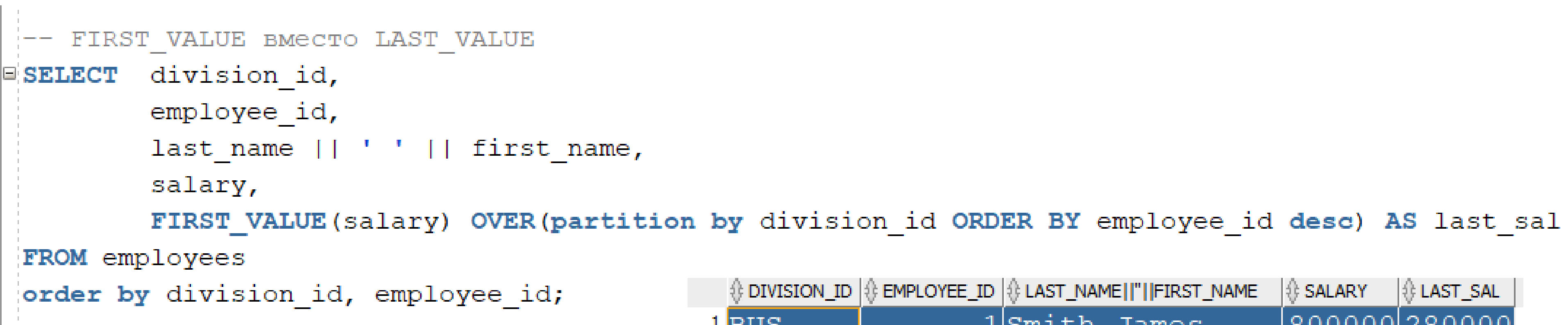


```
--строки используются до текущей строки
SELECT  division_id,
        employee_id,
        last_name || ' ' || first_name,
        salary,
        LAST_VALUE(salary) OVER(partition by division_id ORDER BY employee_id) AS last_sal
FROM employees
order by division_id, employee_id;
```

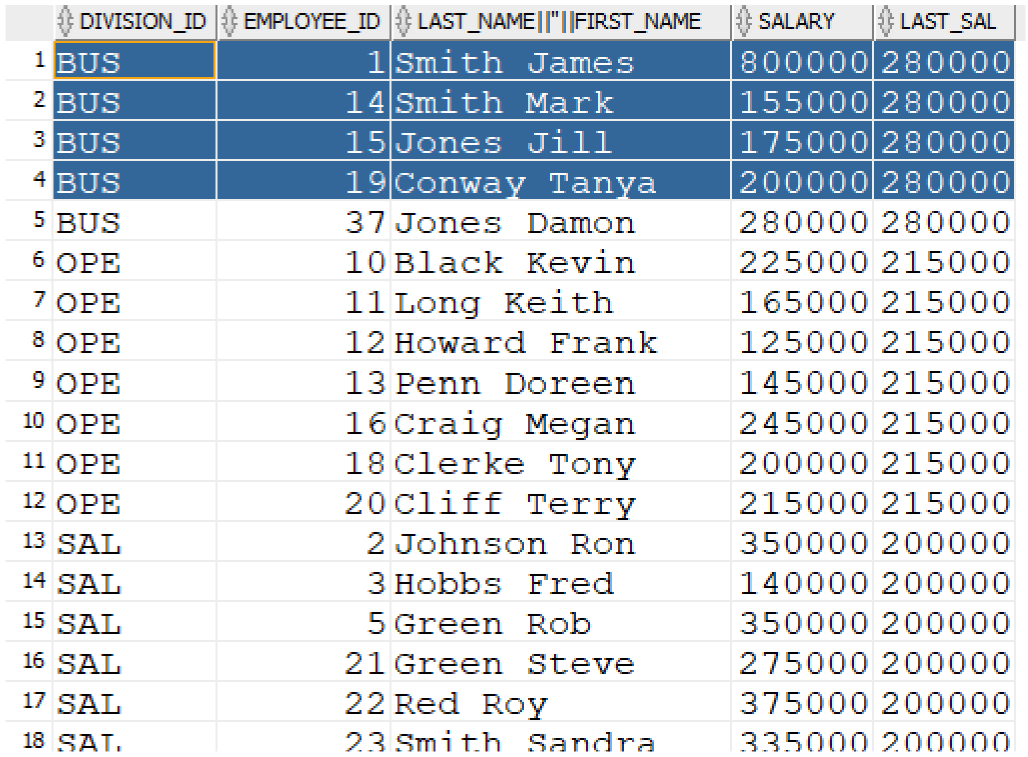


	DIVISION_ID	EMPLOYEE_ID	LAST_NAME||''||FIRST_NAME	SALARY	LAST_SAL
1	BUS	1	Smith James	800000	800000
2	BUS	14	Smith Mark	155000	155000
3	BUS	15	Jones Jill	175000	175000
4	BUS	19	Conway Tanya	200000	200000
5	BUS	37	Jones Damon	280000	280000
6	OPE	10	Black Kevin	225000	225000
7	OPE	11	Long Keith	165000	165000
8	OPE	12	Howard Frank	125000	125000
9	OPE	13	Penn Doreen	145000	145000
10	OPE	16	Craig Megan	245000	245000
11	OPE	18	Clerke Tony	200000	200000
12	OPE	20	Cliff Terry	215000	215000
13	SAL	2	Johnson Ron	350000	350000
14	SAL	3	Hobbs Fred	140000	140000
15	SAL	5	Green Rob	350000	350000
16	SAL	21	Green Steve	275000	275000

ORDER BY

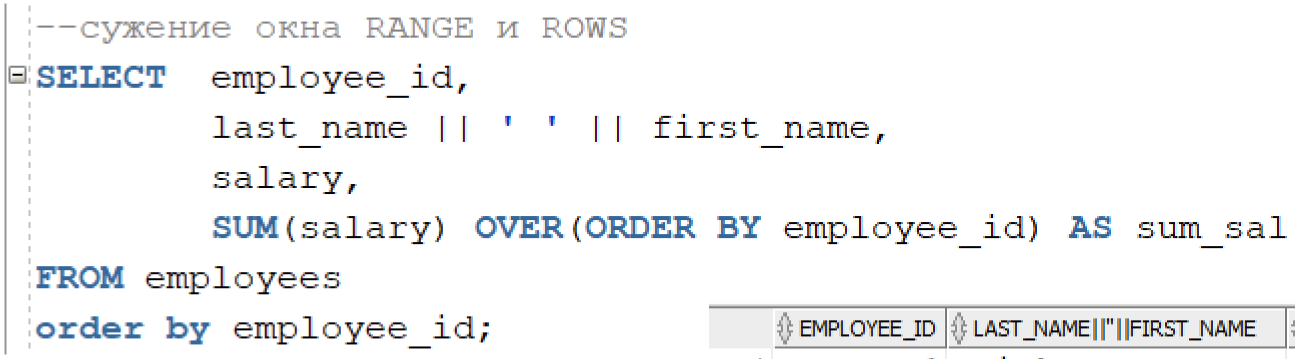


```
-- FIRST_VALUE вместо LAST_VALUE
SELECT  division_id,
        employee_id,
        last_name || ' ' || first_name,
        salary,
        FIRST_VALUE(salary) OVER(partition by division_id ORDER BY employee_id desc) AS last_sal
FROM employees
order by division_id, employee_id;
```

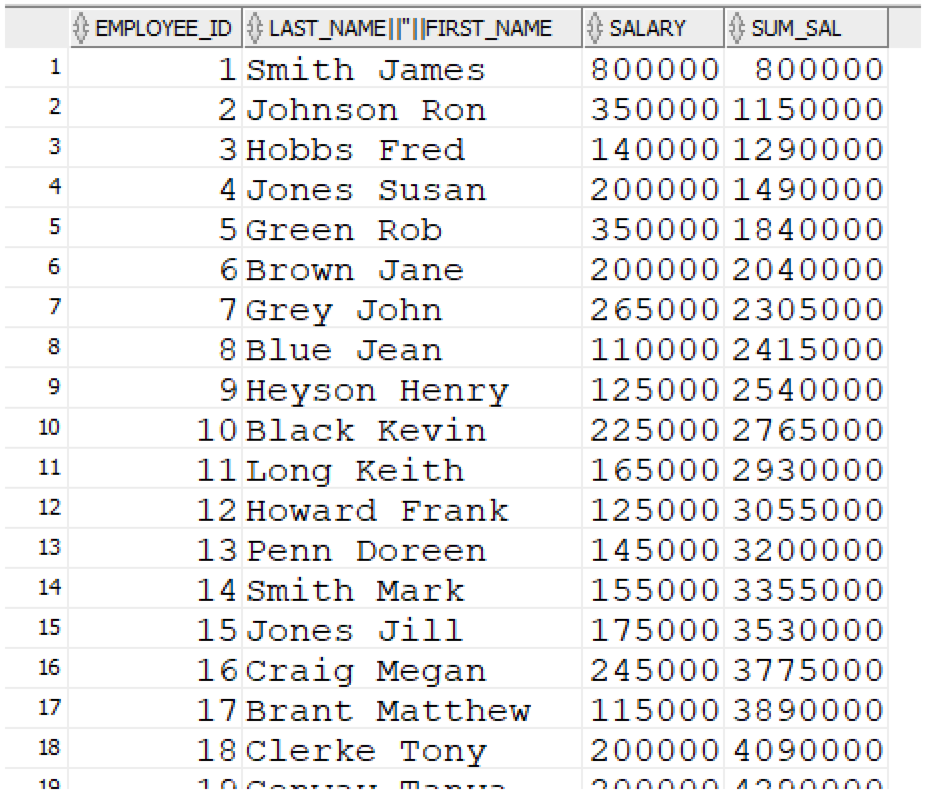


	DIVISION_ID	EMPLOYEE_ID	LAST_NAME||''||FIRST_NAME	SALARY	LAST_SAL
1	BUS	1	Smith James	800000	280000
2	BUS	14	Smith Mark	155000	280000
3	BUS	15	Jones Jill	175000	280000
4	BUS	19	Conway Tanya	200000	280000
5	BUS	37	Jones Damon	280000	280000
6	OPE	10	Black Kevin	225000	215000
7	OPE	11	Long Keith	165000	215000
8	OPE	12	Howard Frank	125000	215000
9	OPE	13	Penn Doreen	145000	215000
10	OPE	16	Craig Megan	245000	215000
11	OPE	18	Clerke Tony	200000	215000
12	OPE	20	Cliff Terry	215000	215000
13	SAL	2	Johnson Ron	350000	200000
14	SAL	3	Hobbs Fred	140000	200000
15	SAL	5	Green Rob	350000	200000
16	SAL	21	Green Steve	275000	200000
17	SAL	22	Red Roy	375000	200000
18	SAL	23	Smith Sandra	335000	200000

Сужение окна

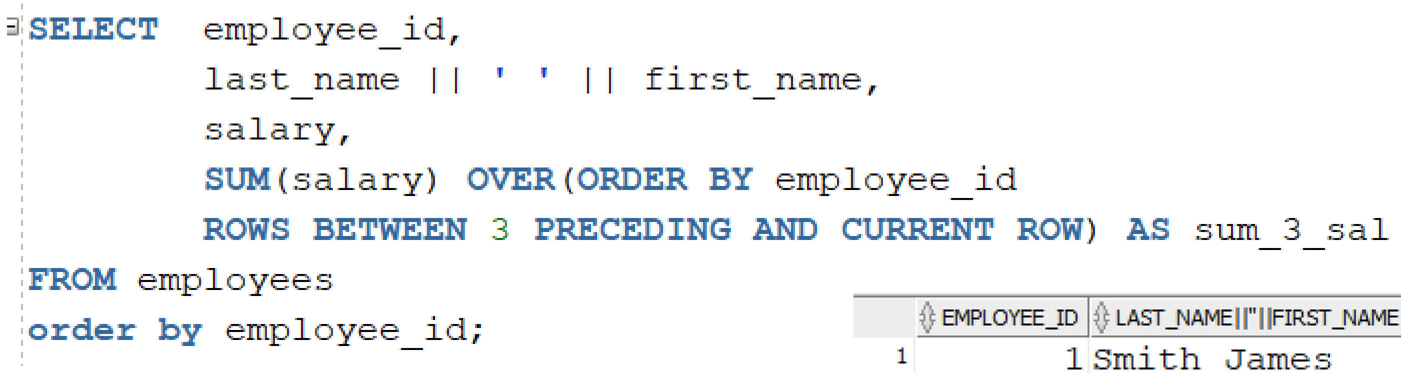


```
--сужение окна RANGE и ROWS
SELECT  employee_id,
        last_name || ' ' || first_name,
        salary,
        SUM(salary) OVER(ORDER BY employee_id) AS sum_sal
FROM employees
order by employee_id;
```

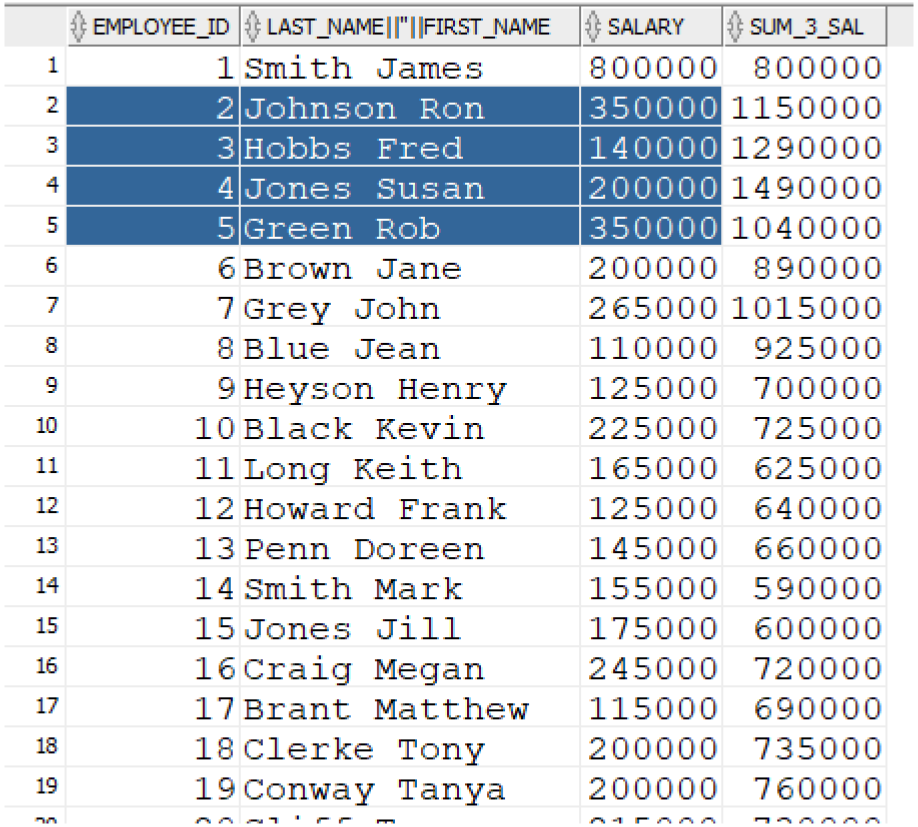


	EMPLOYEE_ID	LAST_NAME||''||FIRST_NAME	SALARY	SUM_SAL
1	1	Smith James	800000	800000
2	2	Johnson Ron	350000	1150000
3	3	Hobbs Fred	140000	1290000
4	4	Jones Susan	200000	1490000
5	5	Green Rob	350000	1840000
6	6	Brown Jane	200000	2040000
7	7	Grey John	265000	2305000
8	8	Blue Jean	110000	2415000
9	9	Heyson Henry	125000	2540000
10	10	Black Kevin	225000	2765000
11	11	Long Keith	165000	2930000
12	12	Howard Frank	125000	3055000
13	13	Penn Doreen	145000	3200000
14	14	Smith Mark	155000	3355000
15	15	Jones Jill	175000	3530000
16	16	Craig Megan	245000	3775000
17	17	Brant Matthew	115000	3890000
18	18	Clerke Tony	200000	4090000
19	[illegible]	[illegible]	[illegible]	[illegible]

Сужение окна



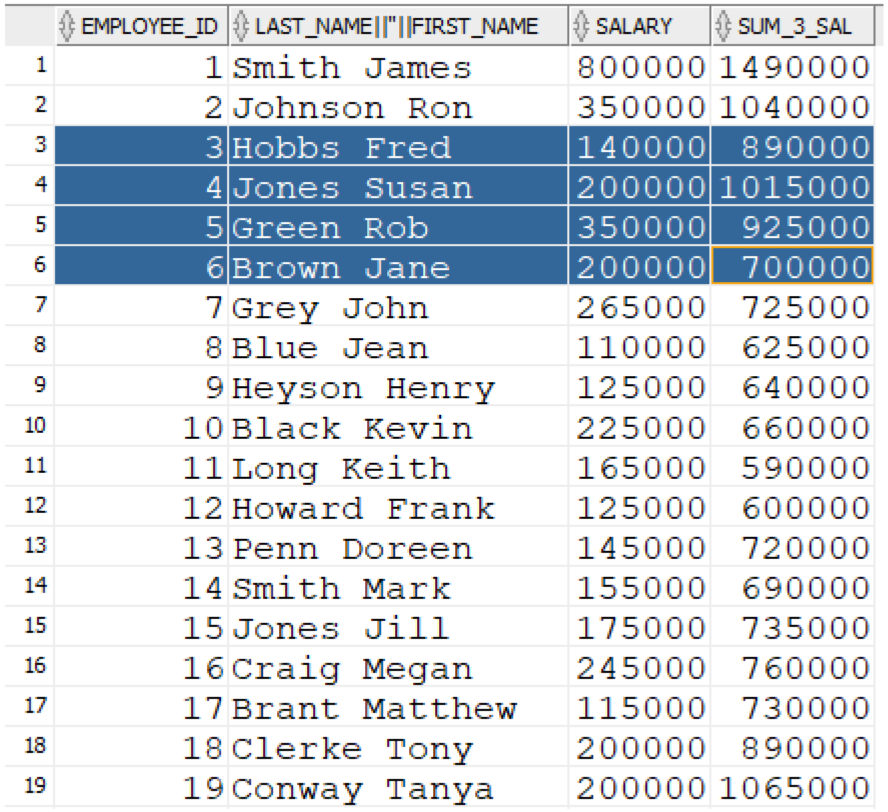
```
SELECT   employee_id,
         last_name || ' ' || first_name,
         salary,
         SUM(salary) OVER(ORDER BY employee_id
         ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) AS sum_3_sal
FROM employees
order by employee_id;
```



	EMPLOYEE_ID	LAST_NAME||''||FIRST_NAME	SALARY	SUM_3_SAL
1	1	Smith James	800000	800000
2	2	Johnson Ron	350000	1150000
3	3	Hobbs Fred	140000	1290000
4	4	Jones Susan	200000	1490000
5	5	Green Rob	350000	1040000
6	6	Brown Jane	200000	890000
7	7	Grey John	265000	1015000
8	8	Blue Jean	110000	925000
9	9	Heyson Henry	125000	700000
10	10	Black Kevin	225000	725000
11	11	Long Keith	165000	625000
12	12	Howard Frank	125000	640000
13	13	Penn Doreen	145000	660000
14	14	Smith Mark	155000	590000
15	15	Jones Jill	175000	600000
16	16	Craig Megan	245000	720000
17	17	Brant Matthew	115000	690000
18	18	Clerke Tony	200000	735000
19	19	Conway Tanya	200000	760000
20	20	[illegible]	[illegible]	[illegible]

Сужение окна

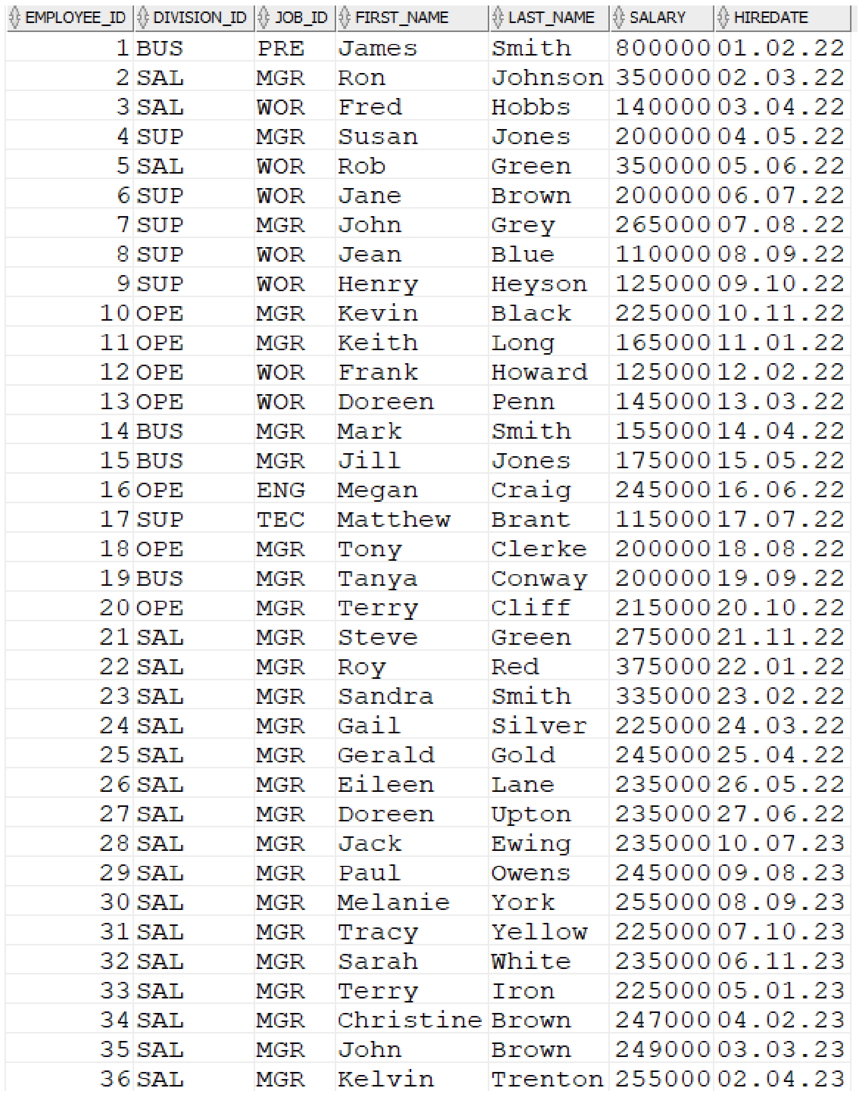
```
SELECT  employee_id,
        last_name || ' ' || first_name,
        salary,
        SUM(salary) OVER(ORDER BY employee_id
        ROWS BETWEEN CURRENT ROW AND 3 FOLLOWING) AS sum_3_sal
FROM employees
order by employee_id;
```



	EMPLOYEE_ID	LAST_NAME||''||FIRST_NAME	SALARY	SUM_3_SAL
1	1	Smith James	800000	1490000
2	2	Johnson Ron	350000	1040000
3	3	Hobbs Fred	140000	890000
4	4	Jones Susan	200000	1015000
5	5	Green Rob	350000	925000
6	6	Brown Jane	200000	700000
7	7	Grey John	265000	725000
8	8	Blue Jean	110000	625000
9	9	Heyson Henry	125000	640000
10	10	Black Kevin	225000	660000
11	11	Long Keith	165000	590000
12	12	Howard Frank	125000	600000
13	13	Penn Doreen	145000	720000
14	14	Smith Mark	155000	690000
15	15	Jones Jill	175000	735000
16	16	Craig Megan	245000	760000
17	17	Brant Matthew	115000	730000
18	18	Clerke Tony	200000	890000
19	19	Conway Tanya	200000	1065000

Конструкции RANGE и ROWS

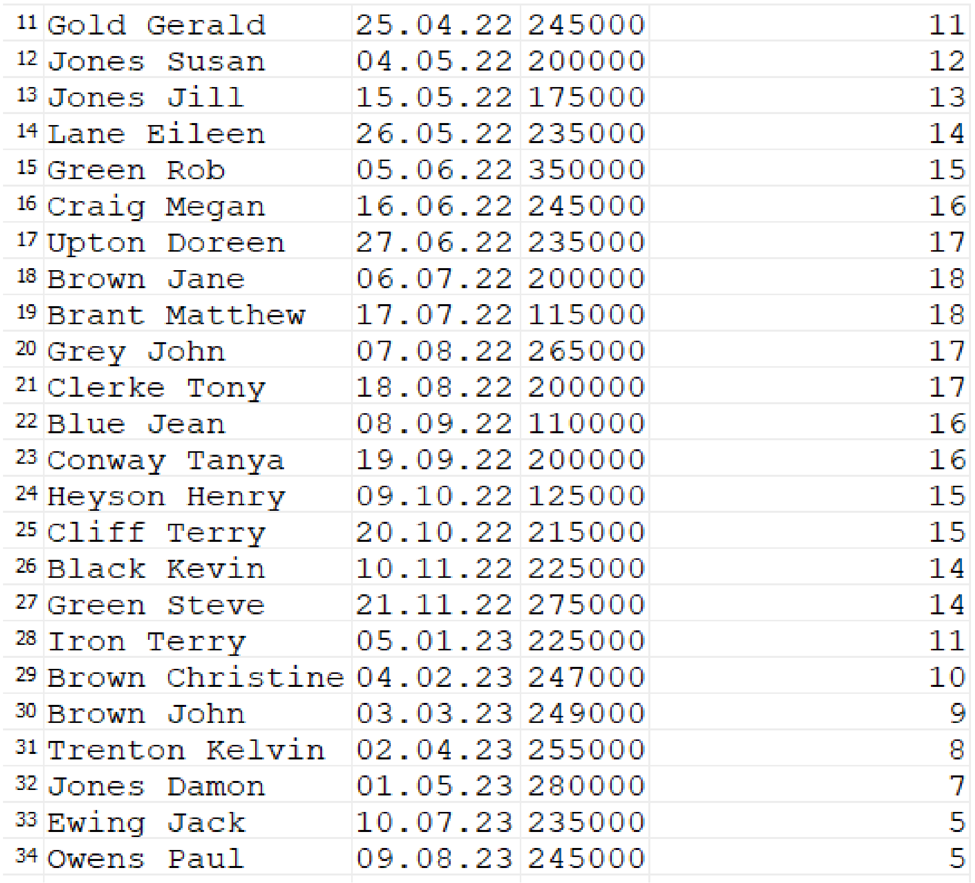
- ROWS – ограничиваются строки
- RANGE – ограничиваются значения в строках (например, даты)



EMPLOYEE_ID	DIVISION_ID	JOB_ID	FIRST_NAME	LAST_NAME	SALARY	HIREDATE
1	BUS	PRE	James	Smith	800000	01.02.22
2	SAL	MGR	Ron	Johnson	350000	02.03.22
3	SAL	WOR	Fred	Hobbs	140000	03.04.22
4	SUP	MGR	Susan	Jones	200000	04.05.22
5	SAL	WOR	Rob	Green	350000	05.06.22
6	SUP	WOR	Jane	Brown	200000	06.07.22
7	SUP	MGR	John	Grey	265000	07.08.22
8	SUP	WOR	Jean	Blue	110000	08.09.22
9	SUP	WOR	Henry	Heyson	125000	09.10.22
10	OPE	MGR	Kevin	Black	225000	10.11.22
11	OPE	MGR	Keith	Long	165000	11.01.22
12	OPE	WOR	Frank	Howard	125000	12.02.22
13	OPE	WOR	Doreen	Penn	145000	13.03.22
14	BUS	MGR	Mark	Smith	155000	14.04.22
15	BUS	MGR	Jill	Jones	175000	15.05.22
16	OPE	ENG	Megan	Craig	245000	16.06.22
17	SUP	TEC	Matthew	Brant	115000	17.07.22
18	OPE	MGR	Tony	Clerke	200000	18.08.22
19	BUS	MGR	Tanya	Conway	200000	19.09.22
20	OPE	MGR	Terry	Cliff	215000	20.10.22
21	SAL	MGR	Steve	Green	275000	21.11.22
22	SAL	MGR	Roy	Red	375000	22.01.22
23	SAL	MGR	Sandra	Smith	335000	23.02.22
24	SAL	MGR	Gail	Silver	225000	24.03.22
25	SAL	MGR	Gerald	Gold	245000	25.04.22
26	SAL	MGR	Eileen	Lane	235000	26.05.22
27	SAL	MGR	Doreen	Upton	235000	27.06.22
28	SAL	MGR	Jack	Ewing	235000	10.07.23
29	SAL	MGR	Paul	Owens	245000	09.08.23
30	SAL	MGR	Melanie	York	255000	08.09.23
31	SAL	MGR	Tracy	Yellow	225000	07.10.23
32	SAL	MGR	Sarah	White	235000	06.11.23
33	SAL	MGR	Terry	Iron	225000	05.01.23
34	SAL	MGR	Christine	Brown	247000	04.02.23
35	SAL	MGR	John	Brown	249000	03.03.23
36	SAL	MGR	Kelvin	Trenton	255000	02.04.23

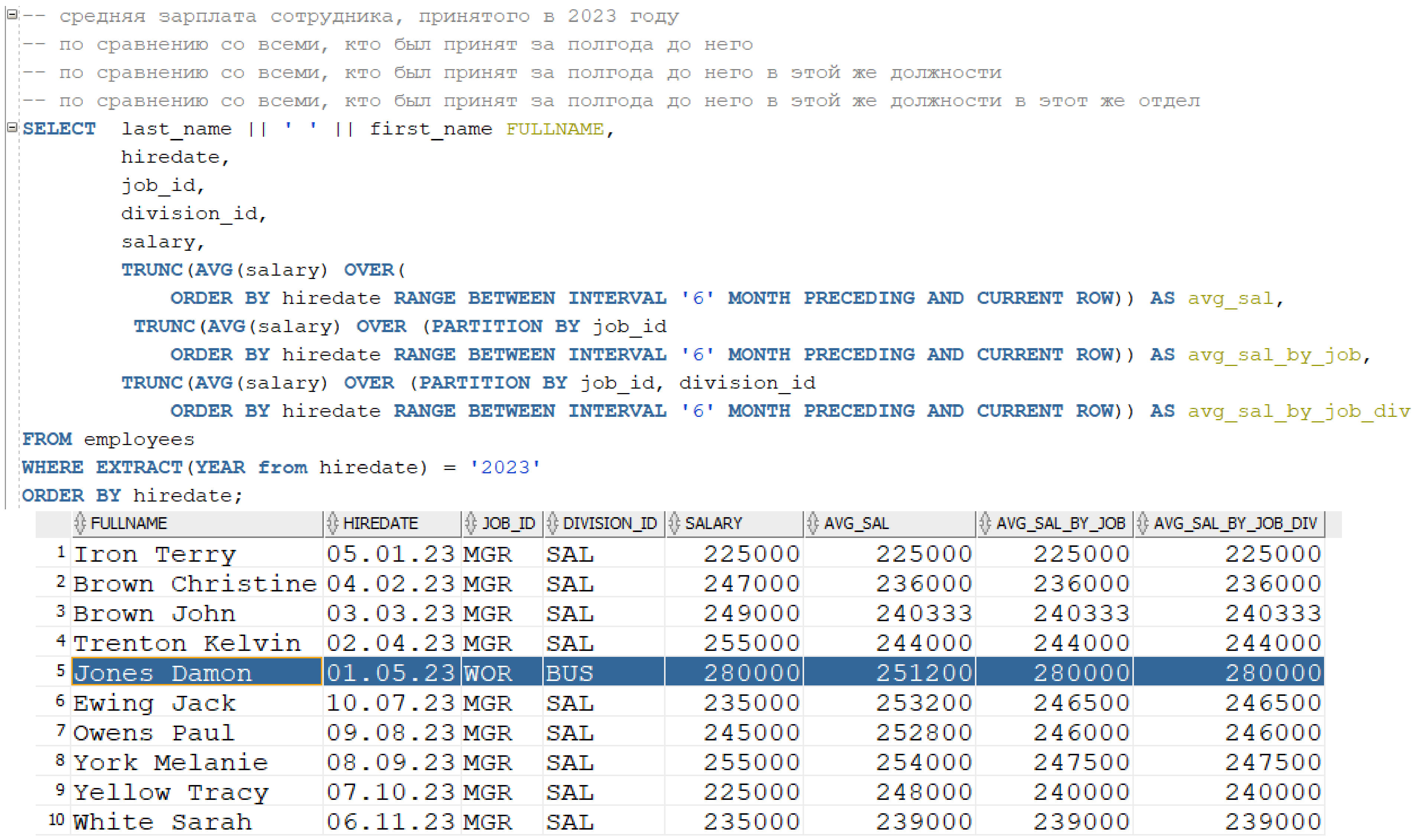
RANGE

```sql
-- сколько сотрудников было принято за полгода до каждого?
SELECT  last_name || ' ' || first_name FULLNAME,
        hiredate,
        COUNT(employee_id) OVER(
            ORDER BY hiredate RANGE INTERVAL '6' MONTH PRECEDING)
            AS count_of_emp_in_6_months
FROM employees
ORDER BY hiredate;
```



11	Gold Gerald	25.04.22	245000	11
12	Jones Susan	04.05.22	200000	12
13	Jones Jill	15.05.22	175000	13
14	Lane Eileen	26.05.22	235000	14
15	Green Rob	05.06.22	350000	15
16	Craig Megan	16.06.22	245000	16
17	Upton Doreen	27.06.22	235000	17
18	Brown Jane	06.07.22	200000	18
19	Brant Matthew	17.07.22	115000	18
20	Grey John	07.08.22	265000	17
21	Clerke Tony	18.08.22	200000	17
22	Blue Jean	08.09.22	110000	16
23	Conway Tanya	19.09.22	200000	16
24	Heyson Henry	09.10.22	125000	15
25	Cliff Terry	20.10.22	215000	15
26	Black Kevin	10.11.22	225000	14
27	Green Steve	21.11.22	275000	14
28	Iron Terry	05.01.23	225000	11
29	Brown Christine	04.02.23	247000	10
30	Brown John	03.03.23	249000	9
31	Trenton Kelvin	02.04.23	255000	8
32	Jones Damon	01.05.23	280000	7
33	Ewing Jack	10.07.23	235000	5
34	Owens Paul	09.08.23	245000	5

RANGE

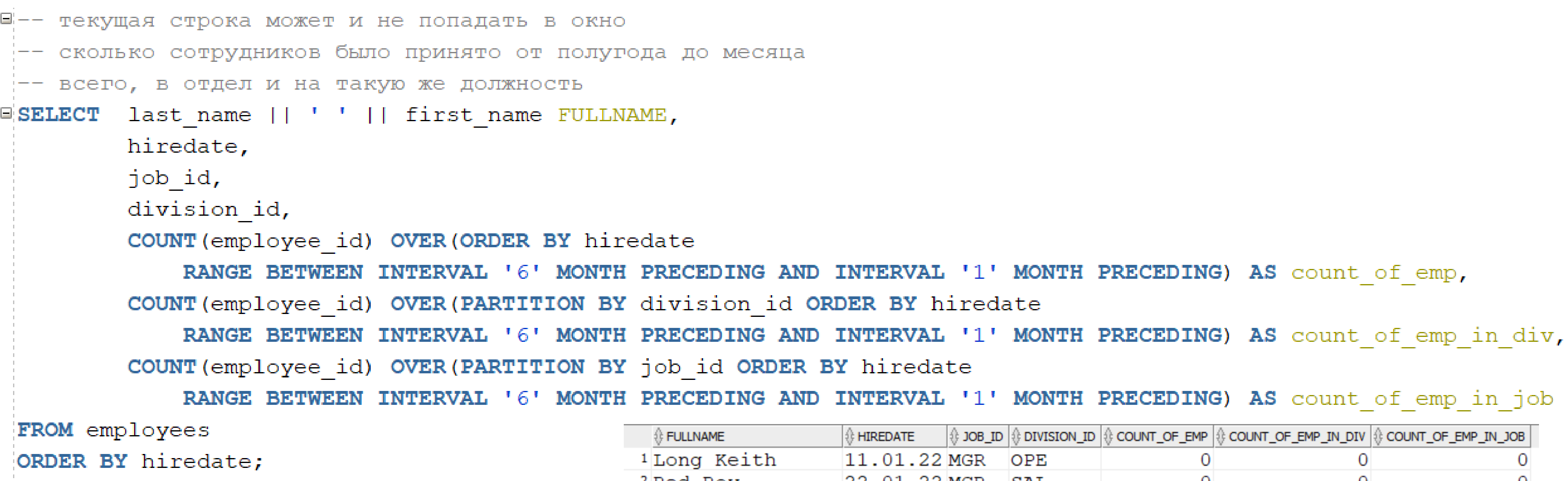


```
-- средняя зарплата сотрудника, принятого в 2023 году
-- по сравнению со всеми, кто был принят за полгода до него
-- по сравнению со всеми, кто был принят за полгода до него в этой же должности
-- по сравнению со всеми, кто был принят за полгода до него в этой же должности в этот же отдел
SELECT  last_name || ' ' || first_name FULLNAME,
        hiredate,
        job_id,
        division_id,
        salary,
        TRUNC(AVG(salary) OVER(
            ORDER BY hiredate RANGE BETWEEN INTERVAL '6' MONTH PRECEDING AND CURRENT ROW)) AS avg_sal,
         TRUNC(AVG(salary) OVER (PARTITION BY job_id
            ORDER BY hiredate RANGE BETWEEN INTERVAL '6' MONTH PRECEDING AND CURRENT ROW)) AS avg_sal_by_job,
        TRUNC(AVG(salary) OVER (PARTITION BY job_id, division_id
            ORDER BY hiredate RANGE BETWEEN INTERVAL '6' MONTH PRECEDING AND CURRENT ROW)) AS avg_sal_by_job_div
FROM employees
WHERE EXTRACT(YEAR from hiredate) = '2023'
ORDER BY hiredate;
```

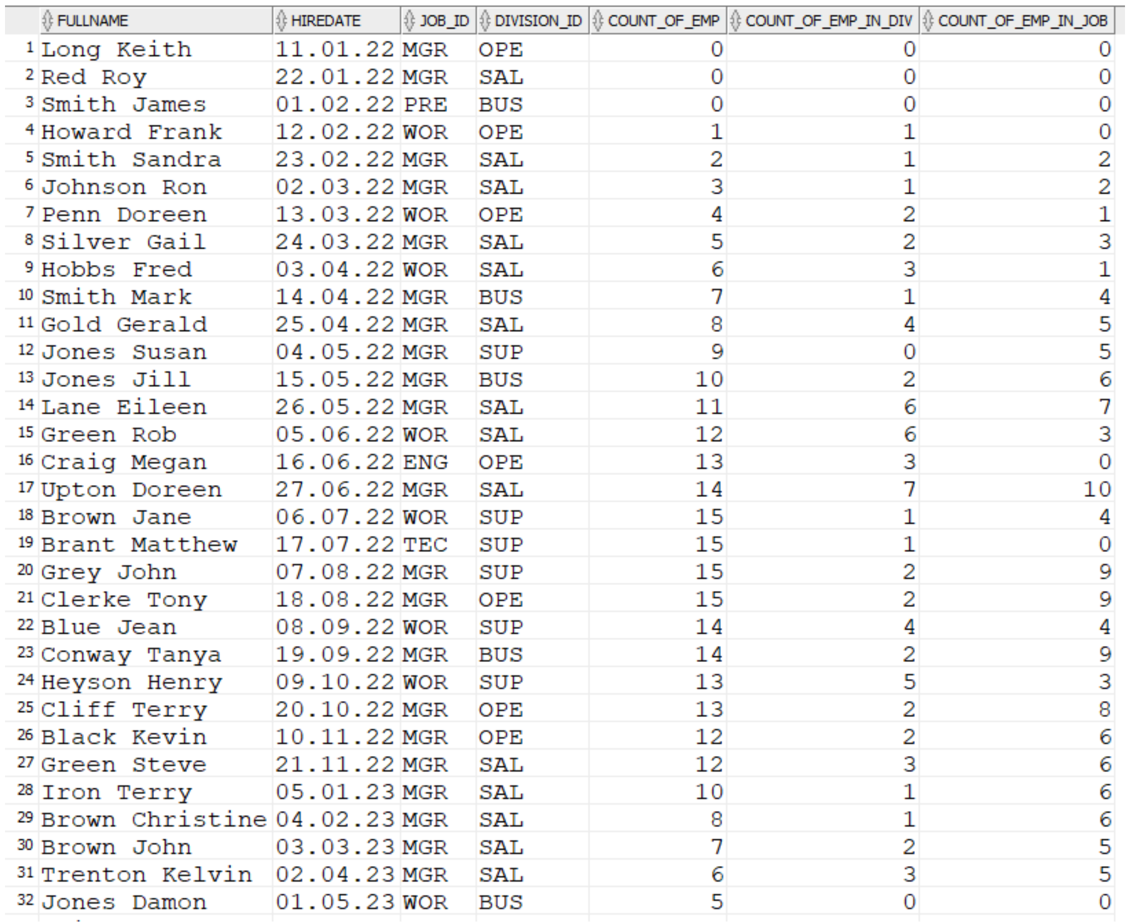
	FULLNAME	HIREDATE	JOB_ID	DIVISION_ID	SALARY	AVG_SAL	AVG_SAL_BY_JOB	AVG_SAL_BY_JOB_DIV
1	Iron Terry	05.01.23	MGR	SAL	225000	225000	225000	225000
2	Brown Christine	04.02.23	MGR	SAL	247000	236000	236000	236000
3	Brown John	03.03.23	MGR	SAL	249000	240333	240333	240333
4	Trenton Kelvin	02.04.23	MGR	SAL	255000	244000	244000	244000
5	Jones Damon	01.05.23	WOR	BUS	280000	251200	280000	280000
6	Ewing Jack	10.07.23	MGR	SAL	235000	253200	246500	246500
7	Owens Paul	09.08.23	MGR	SAL	245000	252800	246000	246000
8	York Melanie	08.09.23	MGR	SAL	255000	254000	247500	247500
9	Yellow Tracy	07.10.23	MGR	SAL	225000	248000	240000	240000
10	White Sarah	06.11.23	MGR	SAL	235000	239000	239000	239000

Сужение окна

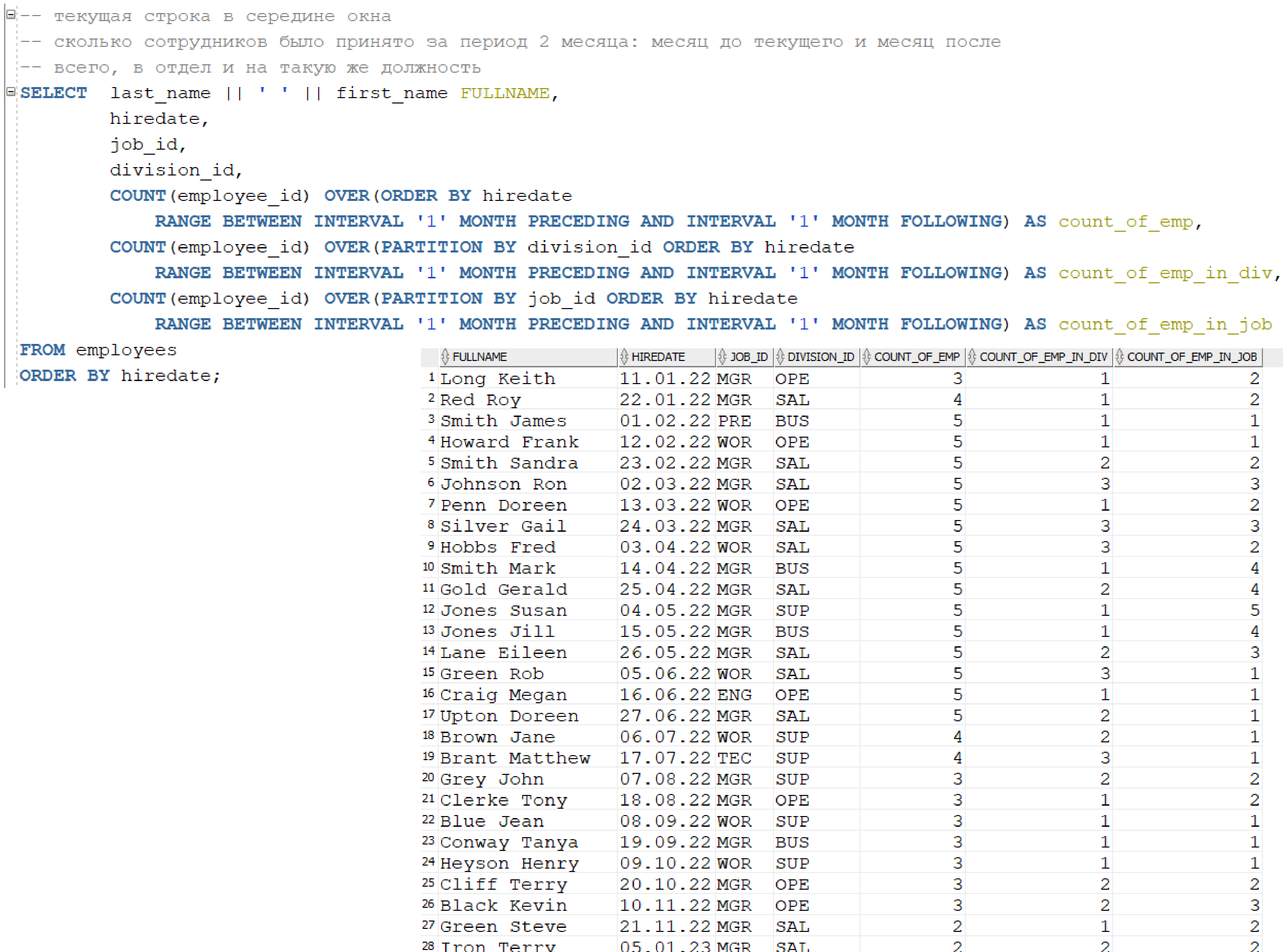
- **UNBOUNDED PRECEDING –** окно не ограничено снизу
- **CURRENT ROW –** окно начинается с текущей строки
- **N PRECEDING** – окно заканчивается текущей строкой, начинается с N строки до текущей
- **N FOLLOWING** – окно начинается текущей строкой, заканчивается N строкой от текущей
- **UNBOUNDED FOLLOWING –** окно не ограничено сверху



```sql
-- текущая строка может и не попадать в окно
-- сколько сотрудников было принято от полугода до месяца
-- всего, в отдел и на такую же должность
SELECT  last_name || ' ' || first_name FULLNAME,
        hiredate,
        job_id,
        division_id,
        COUNT(employee_id) OVER(ORDER BY hiredate
            RANGE BETWEEN INTERVAL '6' MONTH PRECEDING AND INTERVAL '1' MONTH PRECEDING) AS count_of_emp,
        COUNT(employee_id) OVER(PARTITION BY division_id ORDER BY hiredate
            RANGE BETWEEN INTERVAL '6' MONTH PRECEDING AND INTERVAL '1' MONTH PRECEDING) AS count_of_emp_in_div,
        COUNT(employee_id) OVER(PARTITION BY job_id ORDER BY hiredate
            RANGE BETWEEN INTERVAL '6' MONTH PRECEDING AND INTERVAL '1' MONTH PRECEDING) AS count_of_emp_in_job
FROM employees
ORDER BY hiredate;
```



	FULLNAME	HIREDATE	JOB_ID	DIVISION_ID	COUNT_OF_EMP	COUNT_OF_EMP_IN_DIV	COUNT_OF_EMP_IN_JOB
1	Long Keith	11.01.22	MGR	OPE	0	0	0
2	Red Roy	22.01.22	MGR	SAL	0	0	0
3	Smith James	01.02.22	PRE	BUS	0	0	0
4	Howard Frank	12.02.22	WOR	OPE	1	1	0
5	Smith Sandra	23.02.22	MGR	SAL	2	1	2
6	Johnson Ron	02.03.22	MGR	SAL	3	1	2
7	Penn Doreen	13.03.22	WOR	OPE	4	2	1
8	Silver Gail	24.03.22	MGR	SAL	5	2	3
9	Hobbs Fred	03.04.22	WOR	SAL	6	3	1
10	Smith Mark	14.04.22	MGR	BUS	7	1	4
11	Gold Gerald	25.04.22	MGR	SAL	8	4	5
12	Jones Susan	04.05.22	MGR	SUP	9	0	5
13	Jones Jill	15.05.22	MGR	BUS	10	2	6
14	Lane Eileen	26.05.22	MGR	SAL	11	6	7
15	Green Rob	05.06.22	WOR	SAL	12	6	3
16	Craig Megan	16.06.22	ENG	OPE	13	3	0
17	Upton Doreen	27.06.22	MGR	SAL	14	7	10
18	Brown Jane	06.07.22	WOR	SUP	15	1	4
19	Brant Matthew	17.07.22	TEC	SUP	15	1	0
20	Grey John	07.08.22	MGR	SUP	15	2	9
21	Clerke Tony	18.08.22	MGR	OPE	15	2	9
22	Blue Jean	08.09.22	WOR	SUP	14	4	4
23	Conway Tanya	19.09.22	MGR	BUS	14	2	9
24	Heyson Henry	09.10.22	WOR	SUP	13	5	3
25	Cliff Terry	20.10.22	MGR	OPE	13	2	8
26	Black Kevin	10.11.22	MGR	OPE	12	2	6
27	Green Steve	21.11.22	MGR	SAL	12	3	6
28	Iron Terry	05.01.23	MGR	SAL	10	1	6
29	Brown Christine	04.02.23	MGR	SAL	8	1	6
30	Brown John	03.03.23	MGR	SAL	7	2	5
31	Trenton Kelvin	02.04.23	MGR	SAL	6	3	5
32	Jones Damon	01.05.23	WOR	BUS	5	0	0



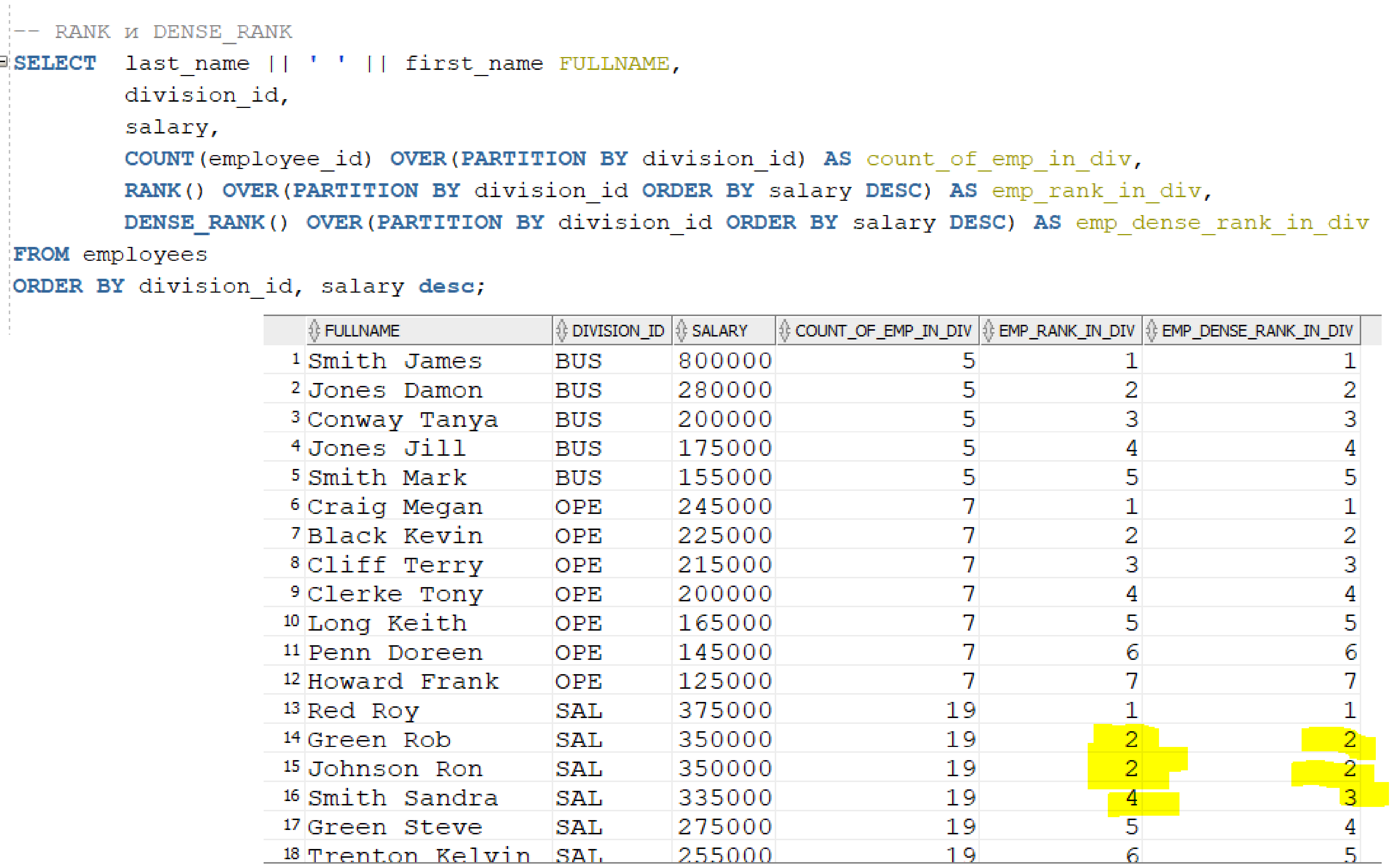
```
-- текущая строка в середине окна
-- сколько сотрудников было принято за период 2 месяца: месяц до текущего и месяц после
-- всего, в отдел и на такую же должность
SELECT  last_name || ' ' || first_name FULLNAME,
        hiredate,
        job_id,
        division_id,
        COUNT(employee_id) OVER(ORDER BY hiredate
            RANGE BETWEEN INTERVAL '1' MONTH PRECEDING AND INTERVAL '1' MONTH FOLLOWING) AS count_of_emp,
        COUNT(employee_id) OVER(PARTITION BY division_id ORDER BY hiredate
            RANGE BETWEEN INTERVAL '1' MONTH PRECEDING AND INTERVAL '1' MONTH FOLLOWING) AS count_of_emp_in_div,
        COUNT(employee_id) OVER(PARTITION BY job_id ORDER BY hiredate
            RANGE BETWEEN INTERVAL '1' MONTH PRECEDING AND INTERVAL '1' MONTH FOLLOWING) AS count_of_emp_in_job
FROM employees
ORDER BY hiredate;
```

	FULLNAME	HIREDATE	JOB_ID	DIVISION_ID	COUNT_OF_EMP	COUNT_OF_EMP_IN_DIV	COUNT_OF_EMP_IN_JOB
1	Long Keith	11.01.22	MGR	OPE	3	1	2
2	Red Roy	22.01.22	MGR	SAL	4	1	2
3	Smith James	01.02.22	PRE	BUS	5	1	1
4	Howard Frank	12.02.22	WOR	OPE	5	1	1
5	Smith Sandra	23.02.22	MGR	SAL	5	2	2
6	Johnson Ron	02.03.22	MGR	SAL	5	3	3
7	Penn Doreen	13.03.22	WOR	OPE	5	1	2
8	Silver Gail	24.03.22	MGR	SAL	5	3	3
9	Hobbs Fred	03.04.22	WOR	SAL	5	3	2
10	Smith Mark	14.04.22	MGR	BUS	5	1	4
11	Gold Gerald	25.04.22	MGR	SAL	5	2	4
12	Jones Susan	04.05.22	MGR	SUP	5	1	5
13	Jones Jill	15.05.22	MGR	BUS	5	1	4
14	Lane Eileen	26.05.22	MGR	SAL	5	2	3
15	Green Rob	05.06.22	WOR	SAL	5	3	1
16	Craig Megan	16.06.22	ENG	OPE	5	1	1
17	Upton Doreen	27.06.22	MGR	SAL	5	2	1
18	Brown Jane	06.07.22	WOR	SUP	4	2	1
19	Brant Matthew	17.07.22	TEC	SUP	4	3	1
20	Grey John	07.08.22	MGR	SUP	3	2	2
21	Clerke Tony	18.08.22	MGR	OPE	3	1	2
22	Blue Jean	08.09.22	WOR	SUP	3	1	1
23	Conway Tanya	19.09.22	MGR	BUS	3	1	1
24	Heyson Henry	09.10.22	WOR	SUP	3	1	1
25	Cliff Terry	20.10.22	MGR	OPE	3	2	2
26	Black Kevin	10.11.22	MGR	OPE	3	2	3
27	Green Steve	21.11.22	MGR	SAL	2	1	2
28	Iron Terry	05.01.23	MGR	SAL	2	2	2

ФУНКЦИИ

- RANK ( ) и DENSE_RANK ( )
- ROW_NUMBER ( )
- LEAD( ) и LAG ( )
- FIRST_VALUE( ) и LAST_VALUE ( )
- RATIO_TO_REPORT ( )
- NTILE ( )
- Конструкции KEEP FIRST и KEEP LAST

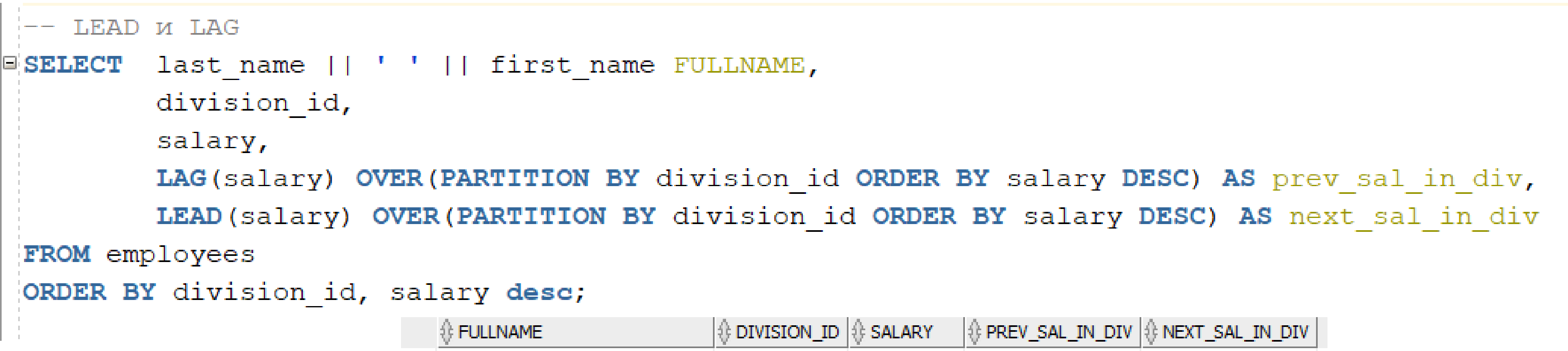
RANK и DENSE_RANK



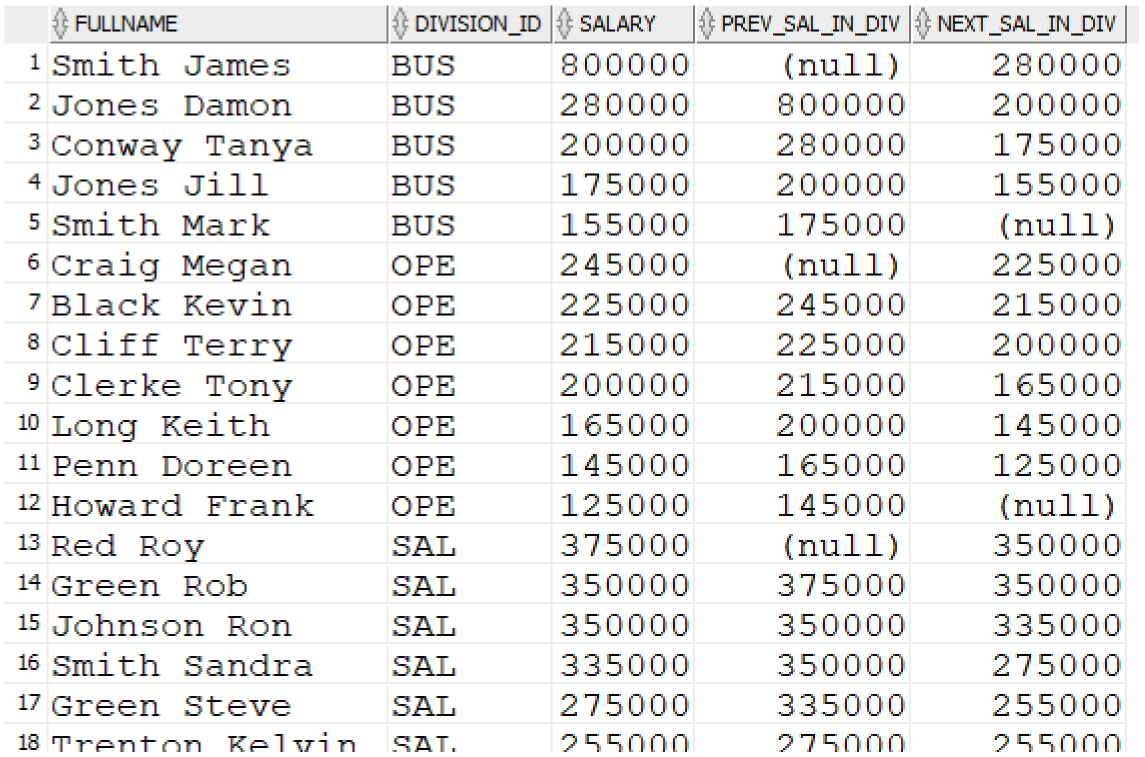
```sql
-- RANK и DENSE_RANK
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        salary,
        COUNT(employee_id) OVER(PARTITION BY division_id) AS count_of_emp_in_div,
        RANK() OVER(PARTITION BY division_id ORDER BY salary DESC) AS emp_rank_in_div,
        DENSE_RANK() OVER(PARTITION BY division_id ORDER BY salary DESC) AS emp_dense_rank_in_div
FROM employees
ORDER BY division_id, salary desc;
```

	FULLNAME	DIVISION_ID	SALARY	COUNT_OF_EMP_IN_DIV	EMP_RANK_IN_DIV	EMP_DENSE_RANK_IN_DIV
1	Smith James	BUS	800000	5	1	1
2	Jones Damon	BUS	280000	5	2	2
3	Conway Tanya	BUS	200000	5	3	3
4	Jones Jill	BUS	175000	5	4	4
5	Smith Mark	BUS	155000	5	5	5
6	Craig Megan	OPE	245000	7	1	1
7	Black Kevin	OPE	225000	7	2	2
8	Cliff Terry	OPE	215000	7	3	3
9	Clerke Tony	OPE	200000	7	4	4
10	Long Keith	OPE	165000	7	5	5
11	Penn Doreen	OPE	145000	7	6	6
12	Howard Frank	OPE	125000	7	7	7
13	Red Roy	SAL	375000	19	1	1
14	Green Rob	SAL	350000	19	2	2
15	Johnson Ron	SAL	350000	19	2	2
16	Smith Sandra	SAL	335000	19	4	3
17	Green Steve	SAL	275000	19	5	4
18	Trenton Kelvin	SAL	255000	19	6	5

LEAD и LAG

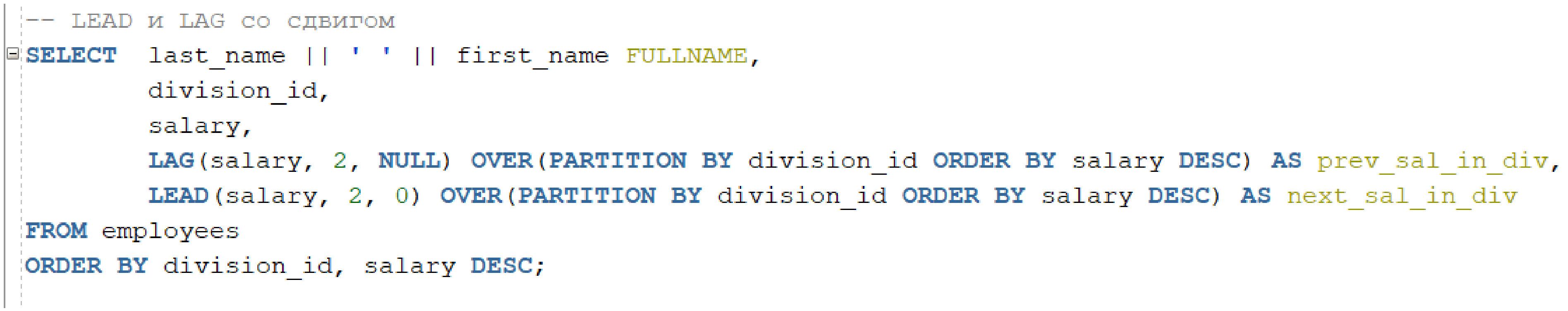


```sql
-- LEAD и LAG
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        salary,
        LAG(salary) OVER(PARTITION BY division_id ORDER BY salary DESC) AS prev_sal_in_div,
        LEAD(salary) OVER(PARTITION BY division_id ORDER BY salary DESC) AS next_sal_in_div
FROM employees
ORDER BY division_id, salary desc;
```

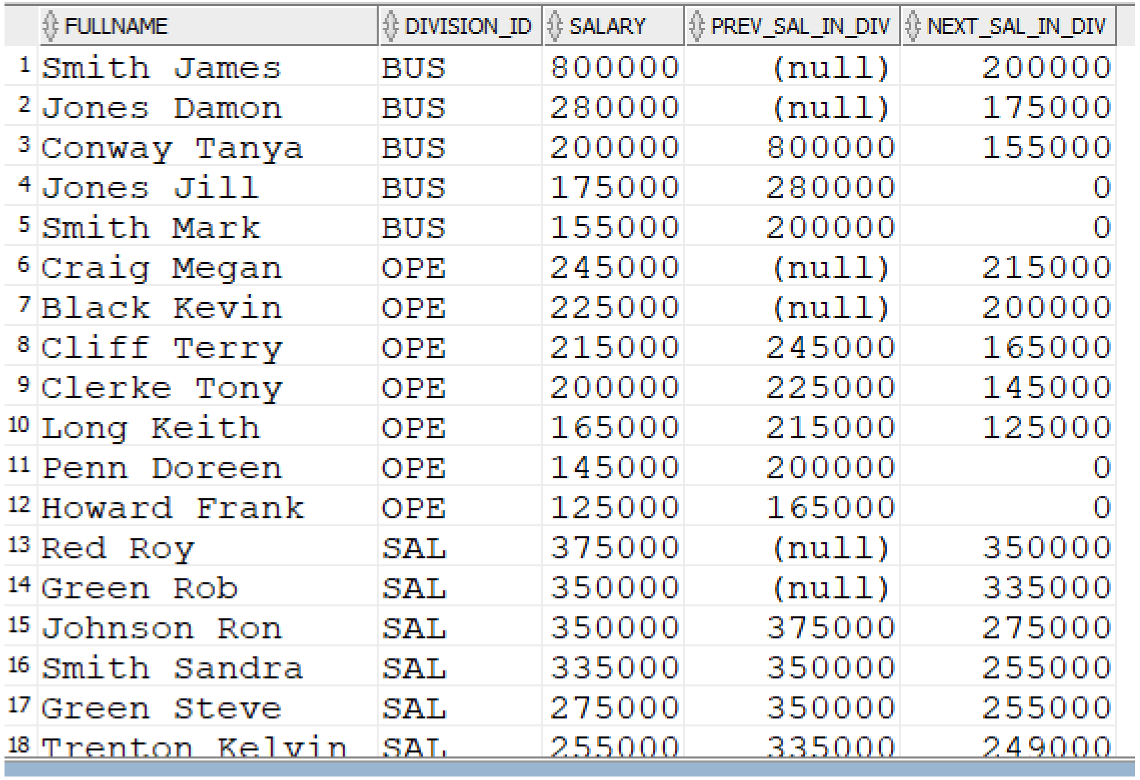


	FULLNAME	DIVISION_ID	SALARY	PREV_SAL_IN_DIV	NEXT_SAL_IN_DIV
1	Smith James	BUS	800000	(null)	280000
2	Jones Damon	BUS	280000	800000	200000
3	Conway Tanya	BUS	200000	280000	175000
4	Jones Jill	BUS	175000	200000	155000
5	Smith Mark	BUS	155000	175000	(null)
6	Craig Megan	OPE	245000	(null)	225000
7	Black Kevin	OPE	225000	245000	215000
8	Cliff Terry	OPE	215000	225000	200000
9	Clerke Tony	OPE	200000	215000	165000
10	Long Keith	OPE	165000	200000	145000
11	Penn Doreen	OPE	145000	165000	125000
12	Howard Frank	OPE	125000	145000	(null)
13	Red Roy	SAL	375000	(null)	350000
14	Green Rob	SAL	350000	375000	350000
15	Johnson Ron	SAL	350000	350000	335000
16	Smith Sandra	SAL	335000	350000	275000
17	Green Steve	SAL	275000	335000	255000
18	Trenton Kelvin	SAL	255000	275000	255000

LEAD и LAG

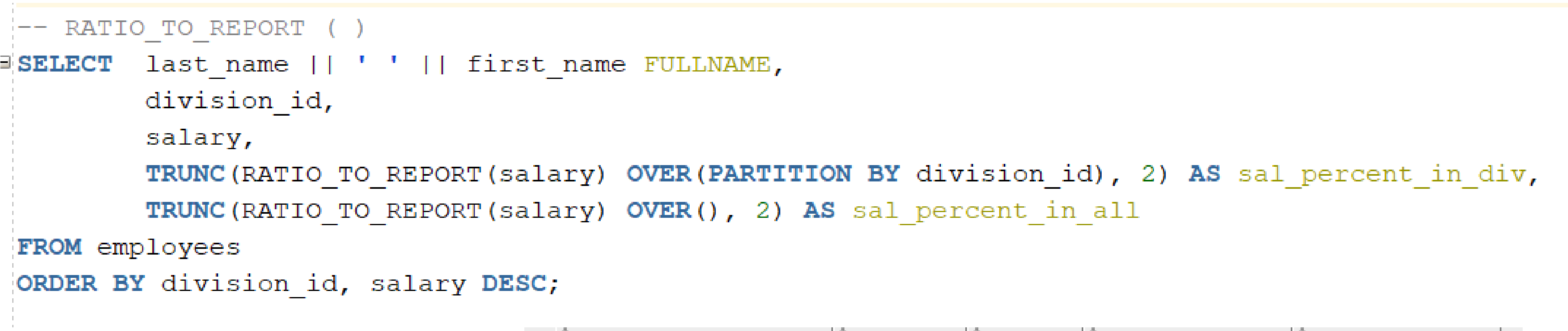


```sql
-- LEAD и LAG со сдвигом
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        salary,
        LAG(salary, 2, NULL) OVER(PARTITION BY division_id ORDER BY salary DESC) AS prev_sal_in_div,
        LEAD(salary, 2, 0) OVER(PARTITION BY division_id ORDER BY salary DESC) AS next_sal_in_div
FROM employees
ORDER BY division_id, salary DESC;
```

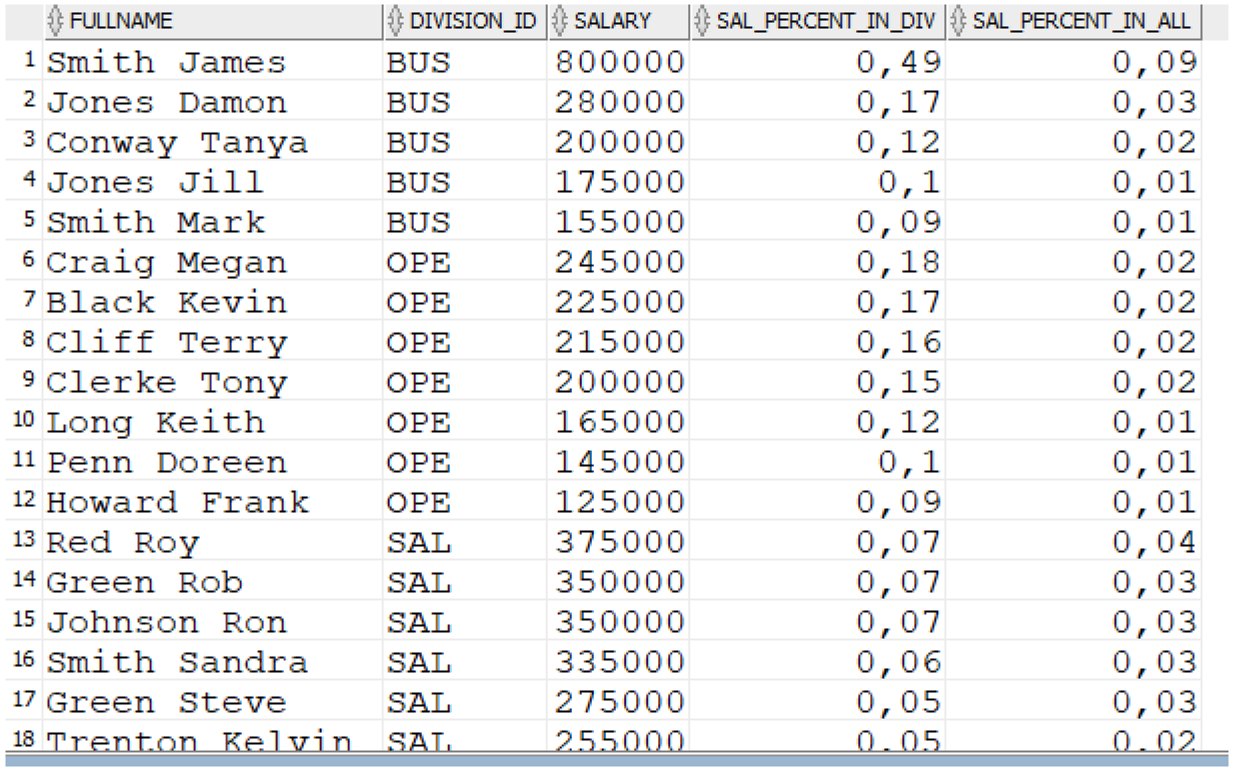


	FULLNAME	DIVISION_ID	SALARY	PREV_SAL_IN_DIV	NEXT_SAL_IN_DIV
1	Smith James	BUS	800000	(null)	200000
2	Jones Damon	BUS	280000	(null)	175000
3	Conway Tanya	BUS	200000	800000	155000
4	Jones Jill	BUS	175000	280000	0
5	Smith Mark	BUS	155000	200000	0
6	Craig Megan	OPE	245000	(null)	215000
7	Black Kevin	OPE	225000	(null)	200000
8	Cliff Terry	OPE	215000	245000	165000
9	Clerke Tony	OPE	200000	225000	145000
10	Long Keith	OPE	165000	215000	125000
11	Penn Doreen	OPE	145000	200000	0
12	Howard Frank	OPE	125000	165000	0
13	Red Roy	SAL	375000	(null)	350000
14	Green Rob	SAL	350000	(null)	335000
15	Johnson Ron	SAL	350000	375000	275000
16	Smith Sandra	SAL	335000	350000	255000
17	Green Steve	SAL	275000	350000	255000
18	Trenton Kelvin	SAL	255000	335000	249000

RATIO_TO_REPORT

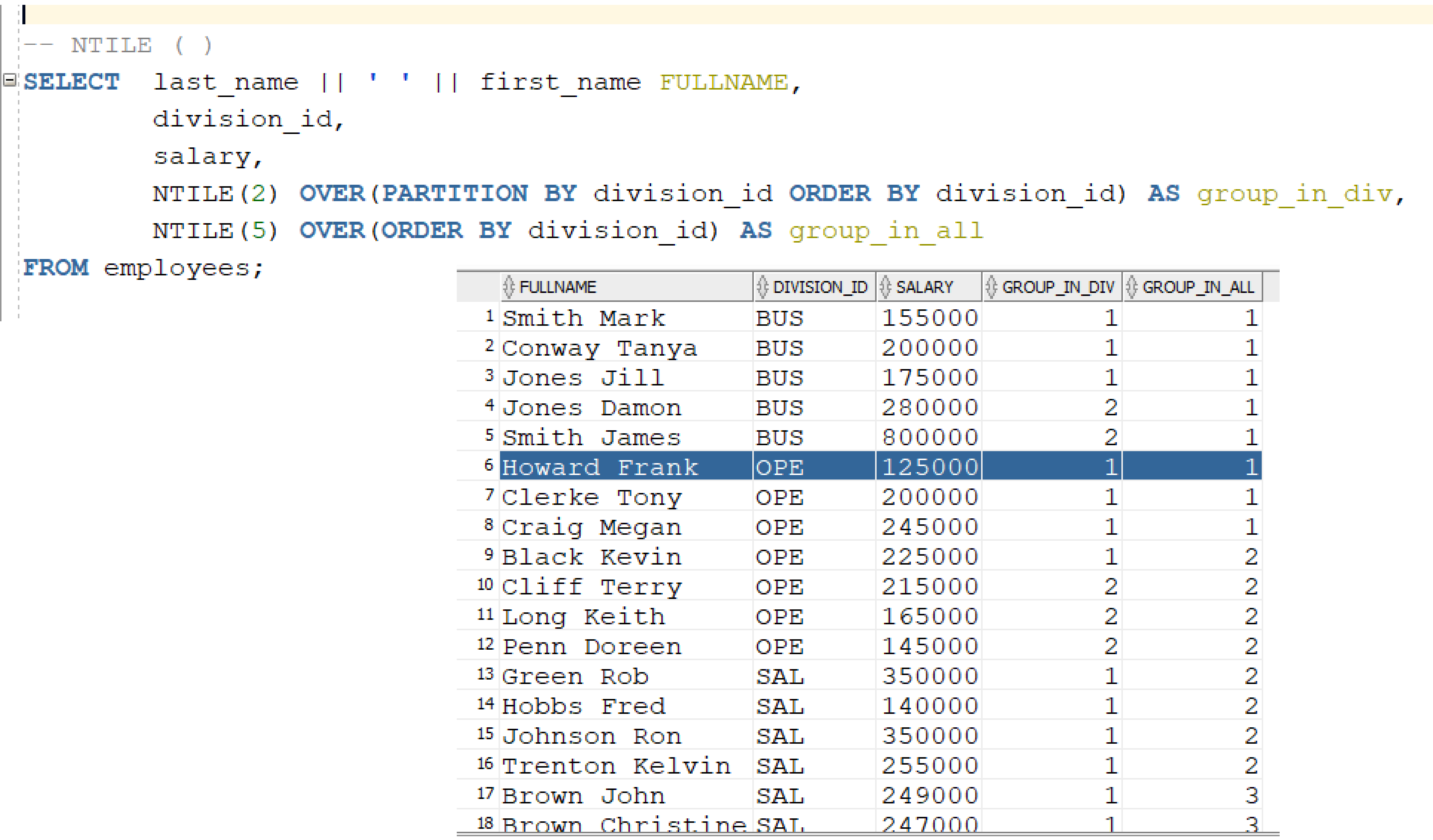


```
-- RATIO_TO_REPORT ( )
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        salary,
        TRUNC(RATIO_TO_REPORT(salary) OVER(PARTITION BY division_id), 2) AS sal_percent_in_div,
        TRUNC(RATIO_TO_REPORT(salary) OVER(), 2) AS sal_percent_in_all
FROM employees
ORDER BY division_id, salary DESC;
```



	FULLNAME	DIVISION_ID	SALARY	SAL_PERCENT_IN_DIV	SAL_PERCENT_IN_ALL
1	Smith James	BUS	800000	0,49	0,09
2	Jones Damon	BUS	280000	0,17	0,03
3	Conway Tanya	BUS	200000	0,12	0,02
4	Jones Jill	BUS	175000	0,1	0,01
5	Smith Mark	BUS	155000	0,09	0,01
6	Craig Megan	OPE	245000	0,18	0,02
7	Black Kevin	OPE	225000	0,17	0,02
8	Cliff Terry	OPE	215000	0,16	0,02
9	Clerke Tony	OPE	200000	0,15	0,02
10	Long Keith	OPE	165000	0,12	0,01
11	Penn Doreen	OPE	145000	0,1	0,01
12	Howard Frank	OPE	125000	0,09	0,01
13	Red Roy	SAL	375000	0,07	0,04
14	Green Rob	SAL	350000	0,07	0,03
15	Johnson Ron	SAL	350000	0,07	0,03
16	Smith Sandra	SAL	335000	0,06	0,03
17	Green Steve	SAL	275000	0,05	0,03
18	Trenton Kelvin	SAL	255000	0,05	0,02

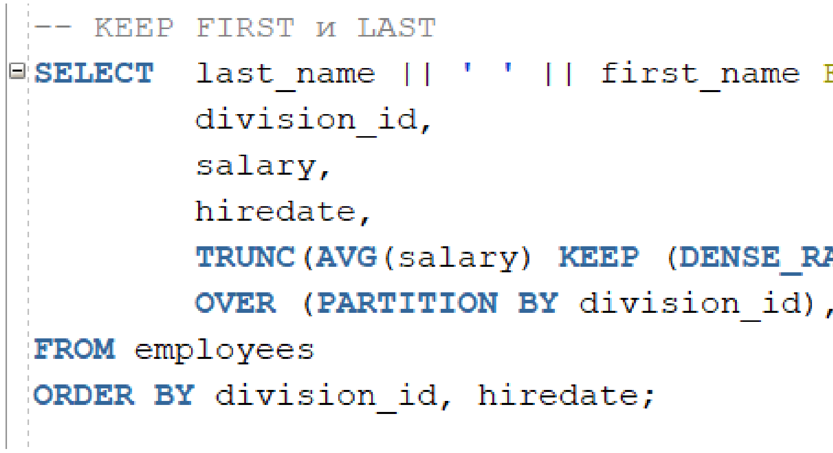
NTILE



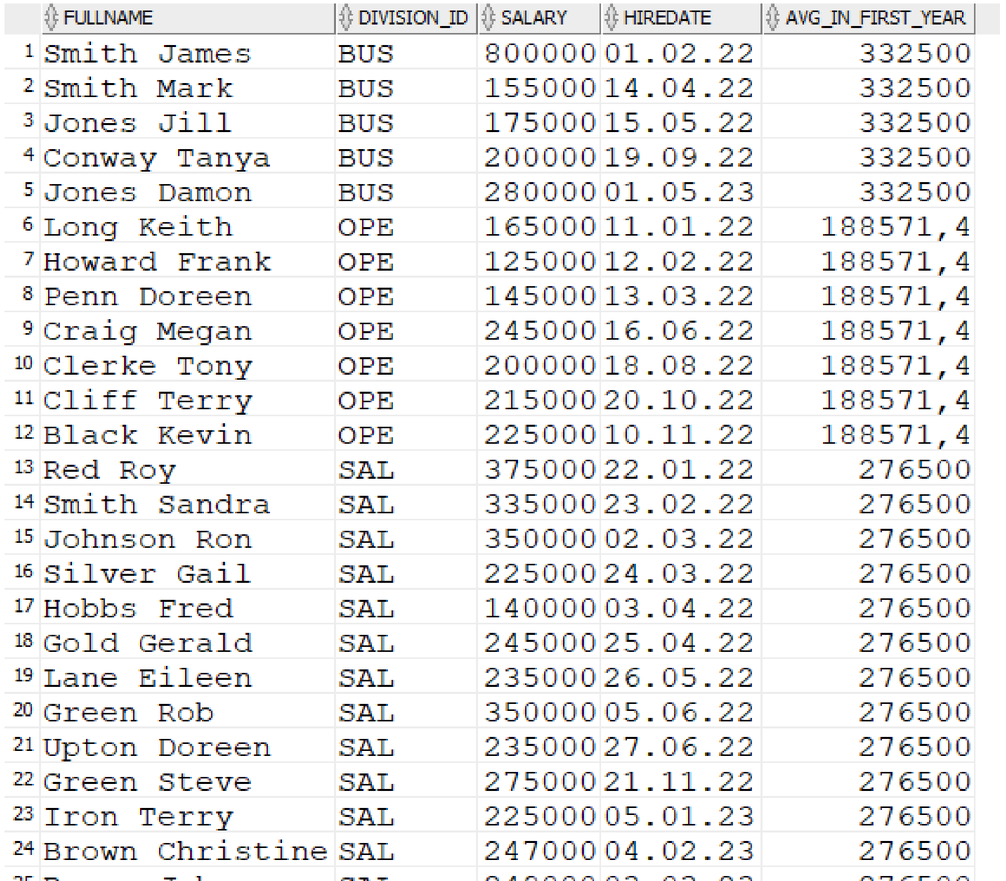
```sql
-- NTILE ( )
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        salary,
        NTILE(2) OVER(PARTITION BY division_id ORDER BY division_id) AS group_in_div,
        NTILE(5) OVER(ORDER BY division_id) AS group_in_all
FROM employees;
```

	FULLNAME	DIVISION_ID	SALARY	GROUP_IN_DIV	GROUP_IN_ALL
1	Smith Mark	BUS	155000	1	1
2	Conway Tanya	BUS	200000	1	1
3	Jones Jill	BUS	175000	1	1
4	Jones Damon	BUS	280000	2	1
5	Smith James	BUS	800000	2	1
6	Howard Frank	OPE	125000	1	1
7	Clerke Tony	OPE	200000	1	1
8	Craig Megan	OPE	245000	1	1
9	Black Kevin	OPE	225000	1	2
10	Cliff Terry	OPE	215000	2	2
11	Long Keith	OPE	165000	2	2
12	Penn Doreen	OPE	145000	2	2
13	Green Rob	SAL	350000	1	2
14	Hobbs Fred	SAL	140000	1	2
15	Johnson Ron	SAL	350000	1	2
16	Trenton Kelvin	SAL	255000	1	2
17	Brown John	SAL	249000	1	3
18	Brown Christine	SAL	247000	1	3

KEEP FIRST и LAST

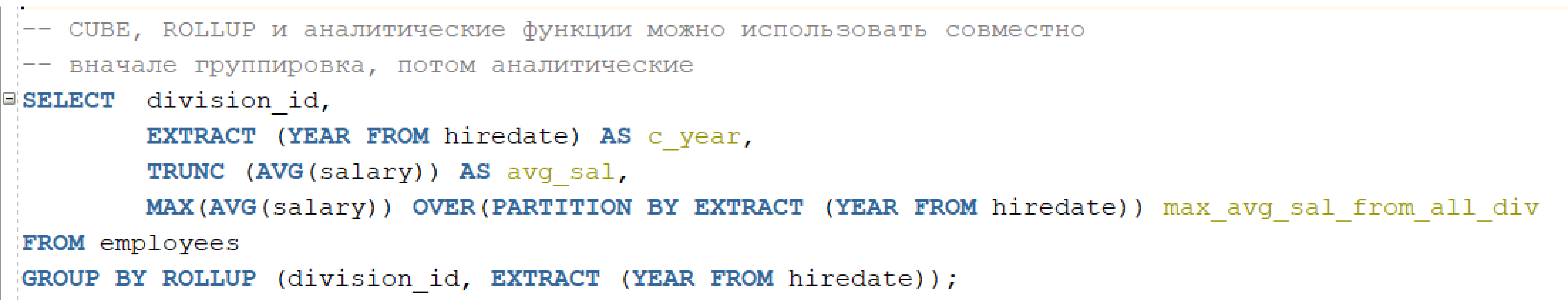


```
-- KEEP FIRST и LAST
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        salary,
        hiredate,
        TRUNC(AVG(salary) KEEP (DENSE_RANK FIRST ORDER BY TRUNC(hiredate,'YYYY'))
        OVER (PARTITION BY division_id), 1) as avg_in_first_year
FROM employees
ORDER BY division_id, hiredate;
```

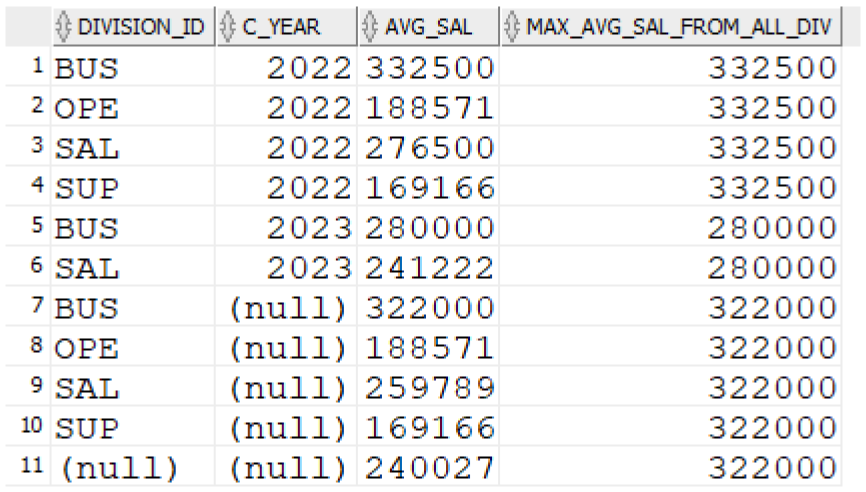


	FULLNAME	DIVISION_ID	SALARY	HIREDATE	AVG_IN_FIRST_YEAR
1	Smith James	BUS	800000	01.02.22	332500
2	Smith Mark	BUS	155000	14.04.22	332500
3	Jones Jill	BUS	175000	15.05.22	332500
4	Conway Tanya	BUS	200000	19.09.22	332500
5	Jones Damon	BUS	280000	01.05.23	332500
6	Long Keith	OPE	165000	11.01.22	188571,4
7	Howard Frank	OPE	125000	12.02.22	188571,4
8	Penn Doreen	OPE	145000	13.03.22	188571,4
9	Craig Megan	OPE	245000	16.06.22	188571,4
10	Clerke Tony	OPE	200000	18.08.22	188571,4
11	Cliff Terry	OPE	215000	20.10.22	188571,4
12	Black Kevin	OPE	225000	10.11.22	188571,4
13	Red Roy	SAL	375000	22.01.22	276500
14	Smith Sandra	SAL	335000	23.02.22	276500
15	Johnson Ron	SAL	350000	02.03.22	276500
16	Silver Gail	SAL	225000	24.03.22	276500
17	Hobbs Fred	SAL	140000	03.04.22	276500
18	Gold Gerald	SAL	245000	25.04.22	276500
19	Lane Eileen	SAL	235000	26.05.22	276500
20	Green Rob	SAL	350000	05.06.22	276500
21	Upton Doreen	SAL	235000	27.06.22	276500
22	Green Steve	SAL	275000	21.11.22	276500
23	Iron Terry	SAL	225000	05.01.23	276500
24	Brown Christine	SAL	247000	04.02.23	276500

Группировки и аналитические функции

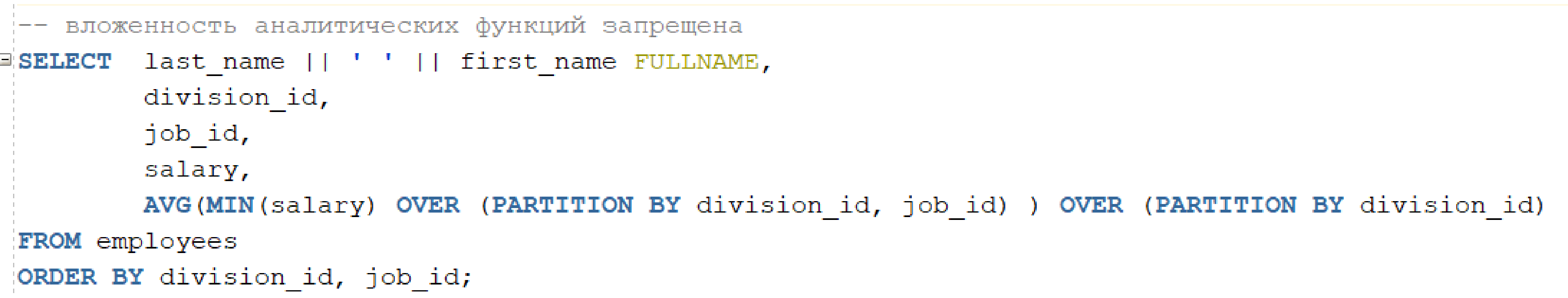


```sql
-- CUBE, ROLLUP и аналитические функции можно использовать совместно
-- вначале группировка, потом аналитические
SELECT  division_id,
        EXTRACT (YEAR FROM hiredate) AS c_year,
        TRUNC (AVG(salary)) AS avg_sal,
        MAX(AVG(salary)) OVER(PARTITION BY EXTRACT (YEAR FROM hiredate)) max_avg_sal_from_all_div
FROM employees
GROUP BY ROLLUP (division_id, EXTRACT (YEAR FROM hiredate));
```

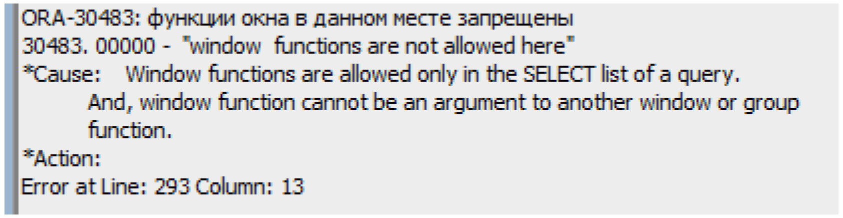


	DIVISION_ID	C_YEAR	AVG_SAL	MAX_AVG_SAL_FROM_ALL_DIV
1	BUS	2022	332500	332500
2	OPE	2022	188571	332500
3	SAL	2022	276500	332500
4	SUP	2022	169166	332500
5	BUS	2023	280000	280000
6	SAL	2023	241222	280000
7	BUS	(null)	322000	322000
8	OPE	(null)	188571	322000
9	SAL	(null)	259789	322000
10	SUP	(null)	169166	322000
11	(null)	(null)	240027	322000

Вложенность аналитических функций

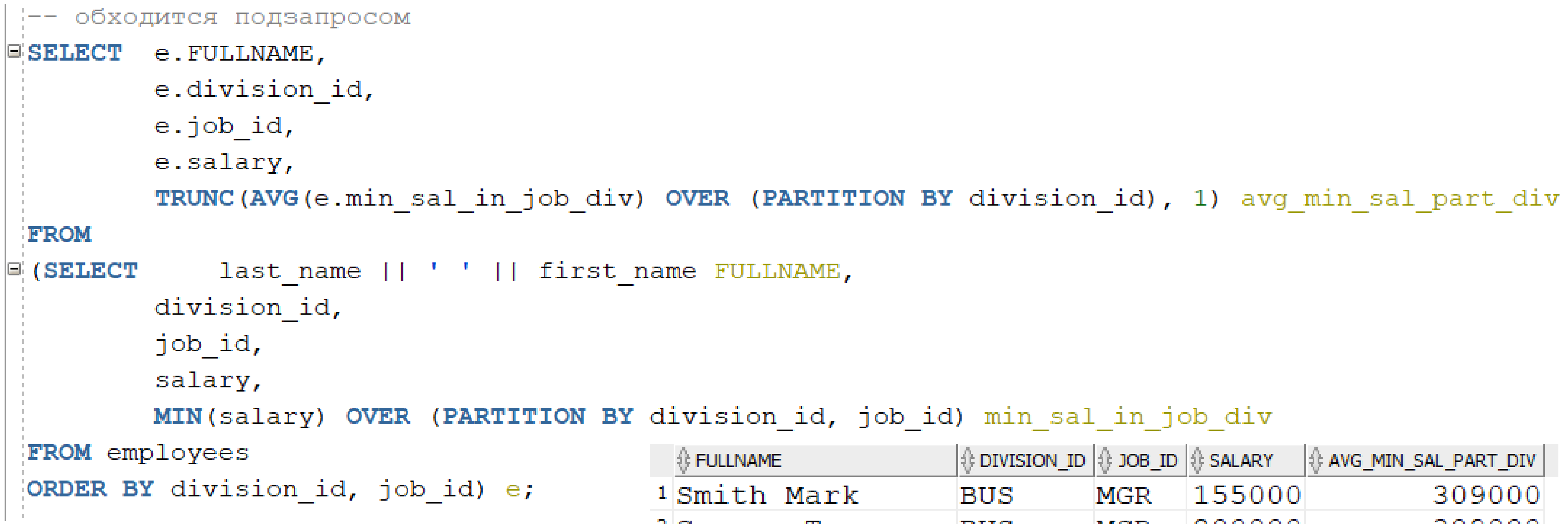


```
-- вложенность аналитических функций запрещена
SELECT  last_name || ' ' || first_name FULLNAME,
        division_id,
        job_id,
        salary,
        AVG(MIN(salary) OVER (PARTITION BY division_id, job_id) ) OVER (PARTITION BY division_id)
FROM employees
ORDER BY division_id, job_id;
```

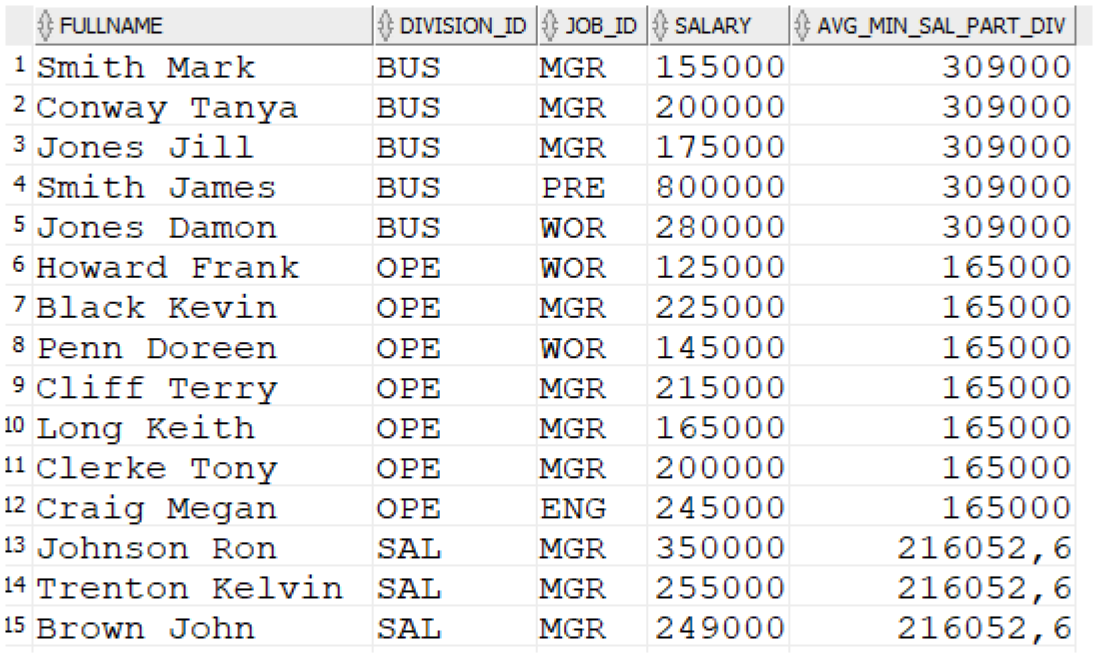


```
ORA-30483: функции окна в данном месте запрещены
30483. 00000 -  "window  functions are not allowed here"
*Cause:    Window functions are allowed only in the SELECT list of a query.
           And, window function cannot be an argument to another window or group
           function.
*Action:
Error at Line: 293 Column: 13
```

Вложенность аналитических функций

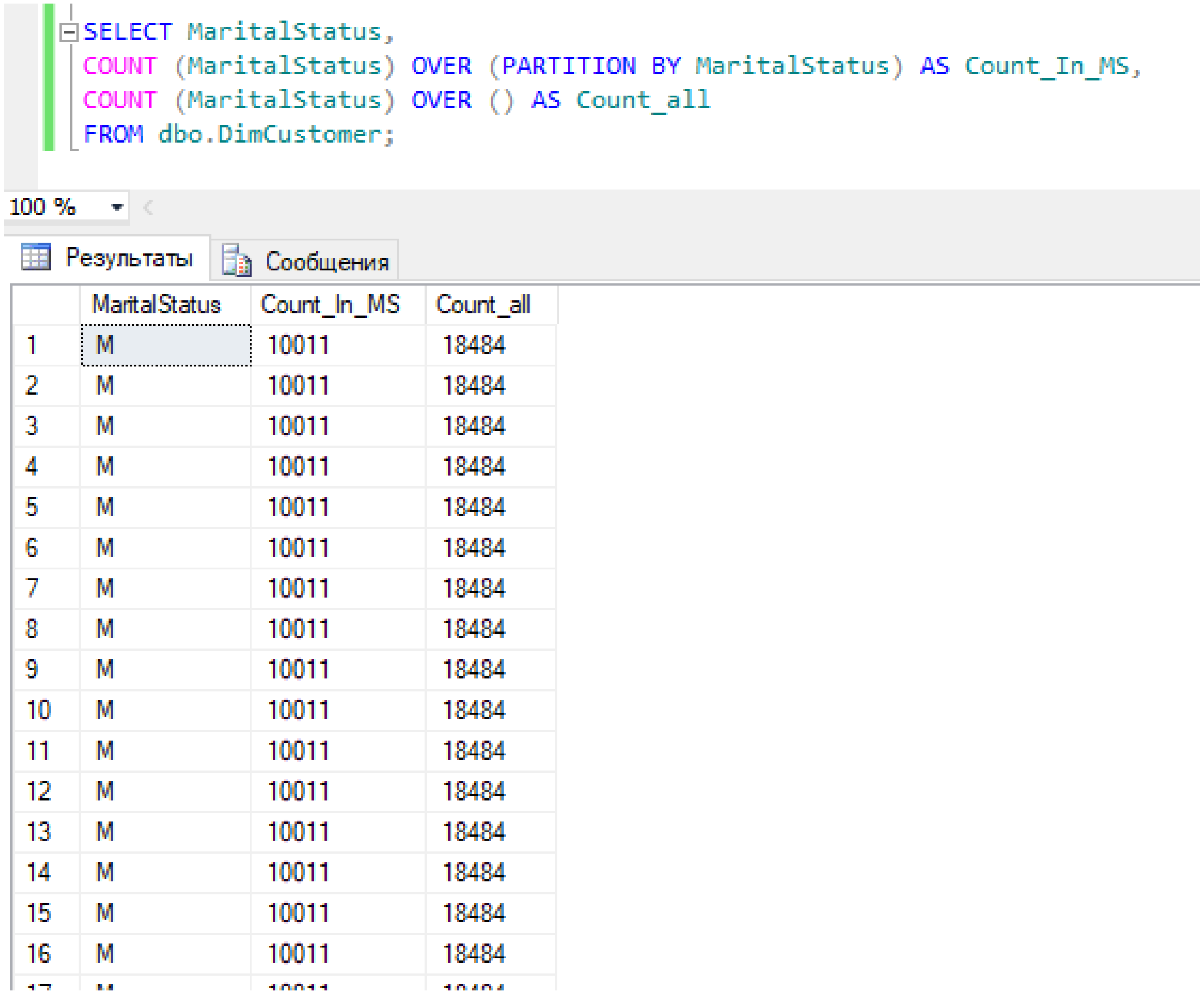


```
-- обходится подзапросом
SELECT  e.FULLNAME,
        e.division_id,
        e.job_id,
        e.salary,
        TRUNC(AVG(e.min_sal_in_job_div) OVER (PARTITION BY division_id), 1) avg_min_sal_part_div
FROM
(SELECT     last_name || ' ' || first_name FULLNAME,
        division_id,
        job_id,
        salary,
        MIN(salary) OVER (PARTITION BY division_id, job_id) min_sal_in_job_div
FROM employees
ORDER BY division_id, job_id) e;
```



	FULLNAME	DIVISION_ID	JOB_ID	SALARY	AVG_MIN_SAL_PART_DIV
1	Smith Mark	BUS	MGR	155000	309000
2	Conway Tanya	BUS	MGR	200000	309000
3	Jones Jill	BUS	MGR	175000	309000
4	Smith James	BUS	PRE	800000	309000
5	Jones Damon	BUS	WOR	280000	309000
6	Howard Frank	OPE	WOR	125000	165000
7	Black Kevin	OPE	MGR	225000	165000
8	Penn Doreen	OPE	WOR	145000	165000
9	Cliff Terry	OPE	MGR	215000	165000
10	Long Keith	OPE	MGR	165000	165000
11	Clerke Tony	OPE	MGR	200000	165000
12	Craig Megan	OPE	ENG	245000	165000
13	Johnson Ron	SAL	MGR	350000	216052,6
14	Trenton Kelvin	SAL	MGR	255000	216052,6
15	Brown John	SAL	MGR	249000	216052,6

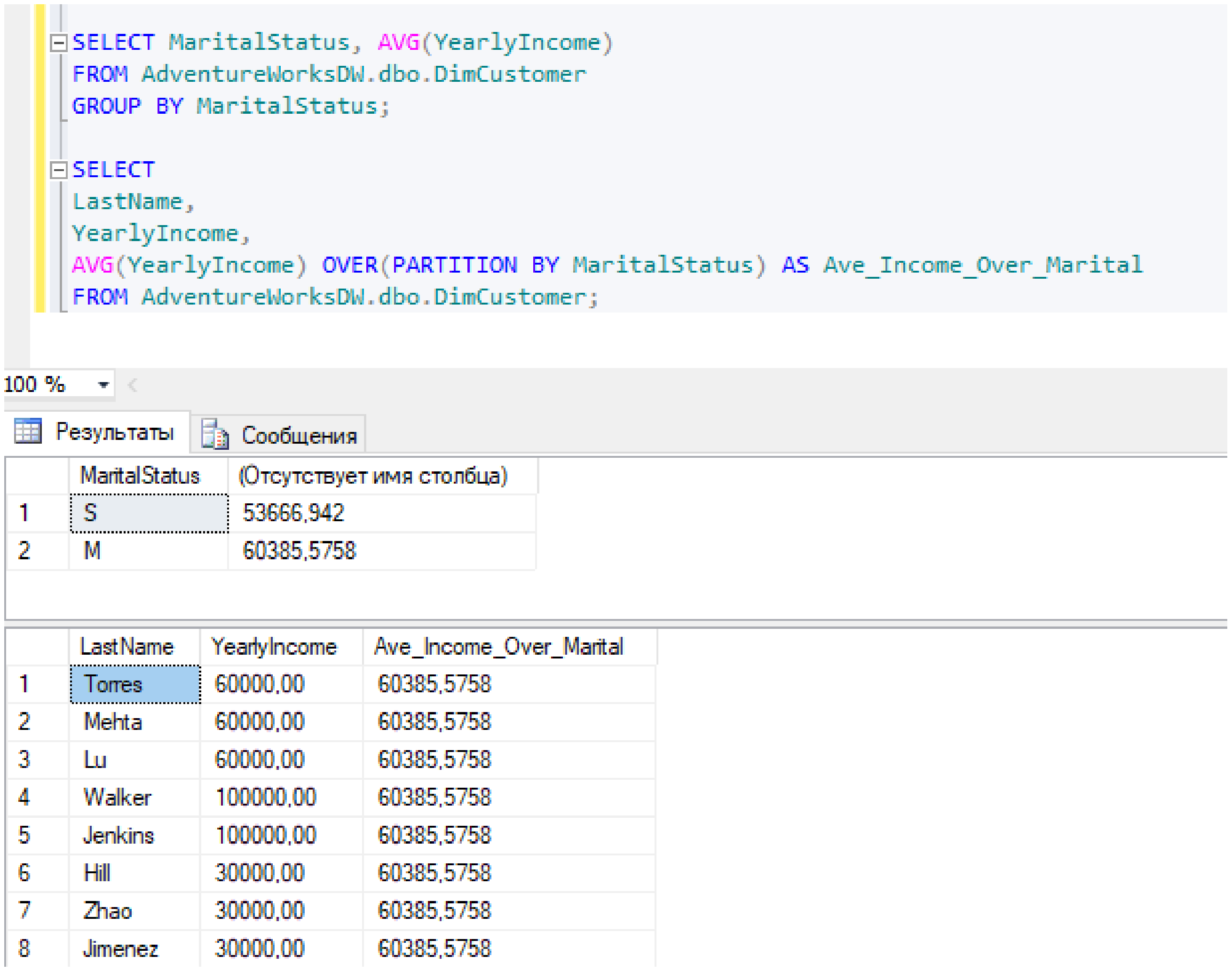
OVER



```sql
SELECT MaritalStatus,
COUNT (MaritalStatus) OVER (PARTITION BY MaritalStatus) AS Count_In_MS,
COUNT (MaritalStatus) OVER () AS Count_all
FROM dbo.DimCustomer;
```

	MaritalStatus	Count_In_MS	Count_all
1	M	10011	18484
2	M	10011	18484
3	M	10011	18484
4	M	10011	18484
5	M	10011	18484
6	M	10011	18484
7	M	10011	18484
8	M	10011	18484
9	M	10011	18484
10	M	10011	18484
11	M	10011	18484
12	M	10011	18484
13	M	10011	18484
14	M	10011	18484
15	M	10011	18484
16	M	10011	18484

OVER



```sql
SELECT MaritalStatus, AVG(YearlyIncome)
FROM AdventureWorksDW.dbo.DimCustomer
GROUP BY MaritalStatus;

SELECT
LastName,
YearlyIncome,
AVG(YearlyIncome) OVER(PARTITION BY MaritalStatus) AS Ave_Income_Over_Marital
FROM AdventureWorksDW.dbo.DimCustomer;
```

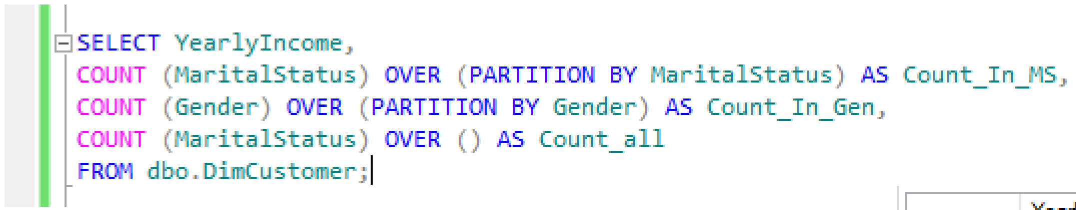
100 %

Результаты | Сообщения

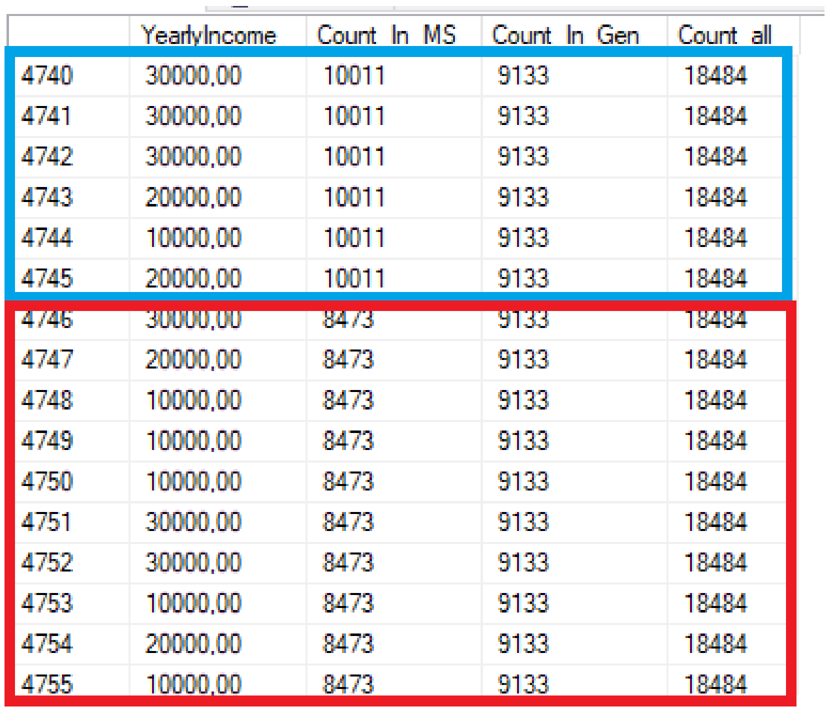
	MaritalStatus	(Отсутствует имя столбца)
1	S	53666,942
2	M	60385,5758

	LastName	YearlyIncome	Ave_Income_Over_Marital
1	Torres	60000,00	60385,5758
2	Mehta	60000,00	60385,5758
3	Lu	60000,00	60385,5758
4	Walker	100000,00	60385,5758
5	Jenkins	100000,00	60385,5758
6	Hill	30000,00	60385,5758
7	Zhao	30000,00	60385,5758
8	Jimenez	30000,00	60385,5758

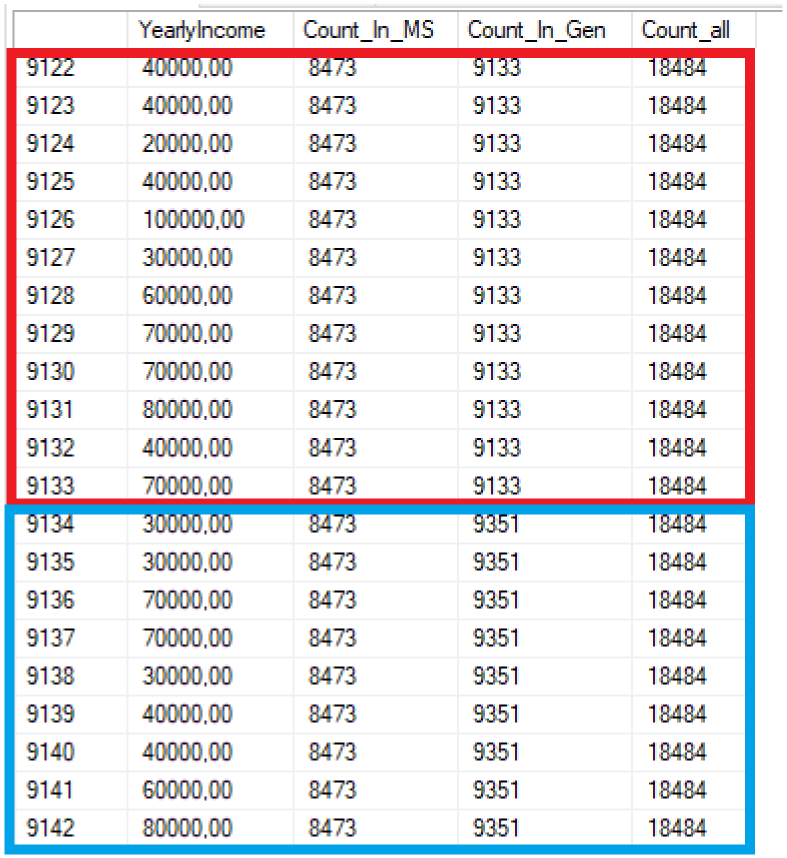
Понятие окна



```
SELECT YearlyIncome,
COUNT (MaritalStatus) OVER (PARTITION BY MaritalStatus) AS Count_In_MS,
COUNT (Gender) OVER (PARTITION BY Gender) AS Count_In_Gen,
COUNT (MaritalStatus) OVER () AS Count_all
FROM dbo.DimCustomer;
```



	YearlyIncome	Count_In_MS	Count_In_Gen	Count_all
4740	30000,00	10011	9133	18484
4741	30000,00	10011	9133	18484
4742	30000,00	10011	9133	18484
4743	20000,00	10011	9133	18484
4744	10000,00	10011	9133	18484
4745	20000,00	10011	9133	18484
4746	30000,00	8473	9133	18484
4747	20000,00	8473	9133	18484
4748	10000,00	8473	9133	18484
4749	10000,00	8473	9133	18484
4750	10000,00	8473	9133	18484
4751	30000,00	8473	9133	18484
4752	30000,00	8473	9133	18484
4753	10000,00	8473	9133	18484
4754	20000,00	8473	9133	18484
4755	10000,00	8473	9133	18484



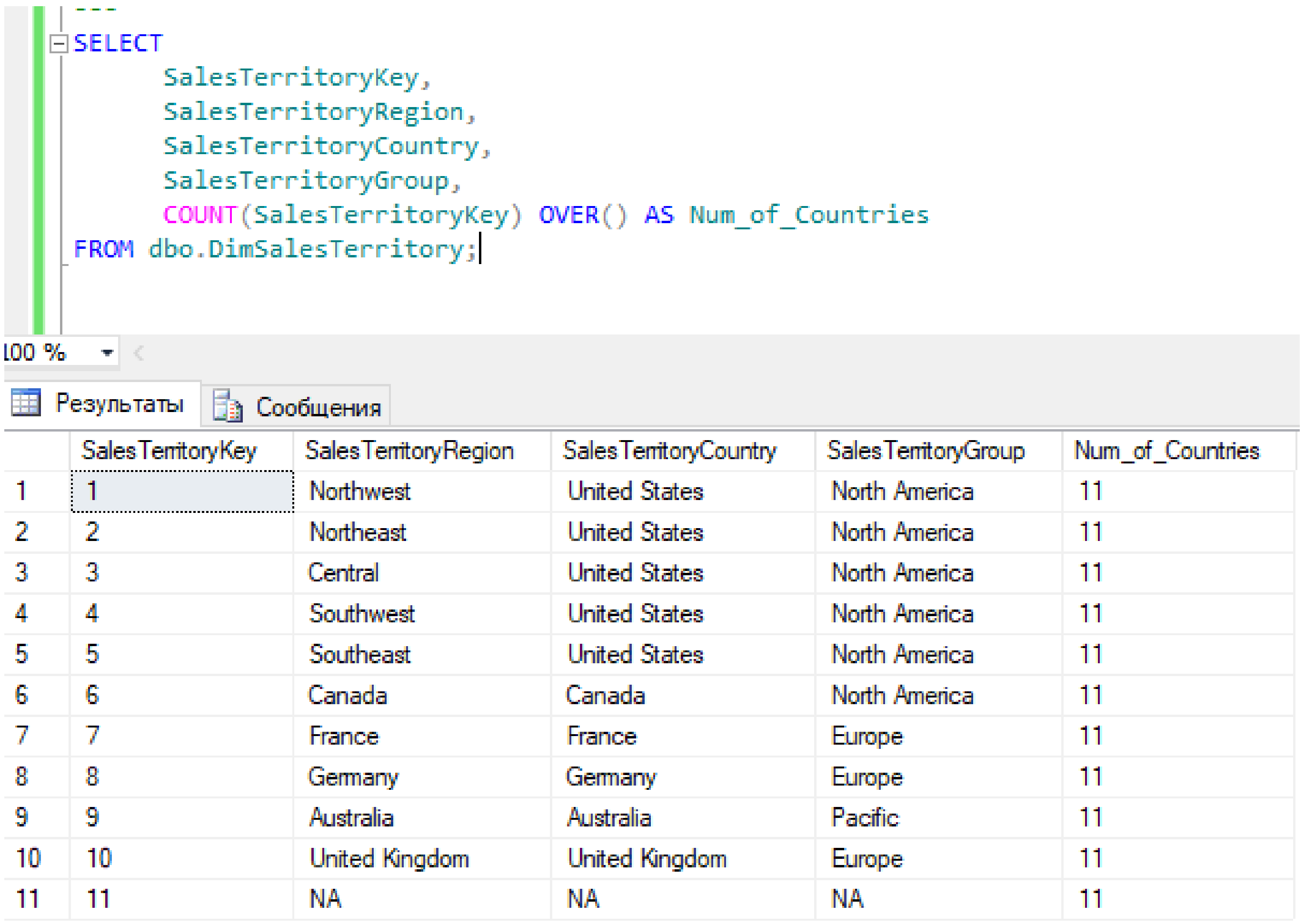
	YearlyIncome	Count_In_MS	Count_In_Gen	Count_all
9122	40000,00	8473	9133	18484
9123	40000,00	8473	9133	18484
9124	20000,00	8473	9133	18484
9125	40000,00	8473	9133	18484
9126	100000,00	8473	9133	18484
9127	30000,00	8473	9133	18484
9128	60000,00	8473	9133	18484
9129	70000,00	8473	9133	18484
9130	70000,00	8473	9133	18484
9131	80000,00	8473	9133	18484
9132	40000,00	8473	9133	18484
9133	70000,00	8473	9133	18484
9134	30000,00	8473	9351	18484
9135	30000,00	8473	9351	18484
9136	70000,00	8473	9351	18484
9137	70000,00	8473	9351	18484
9138	30000,00	8473	9351	18484
9139	40000,00	8473	9351	18484
9140	40000,00	8473	9351	18484
9141	60000,00	8473	9351	18484
9142	80000,00	8473	9351	18484

PARTITION BY - деление на окна

Виды оконных функций

- Агрегатные функции - возвращают значение, полученное путем арифметических вычислений:
  - SUM(), MAX(), MIN(), AVG(), COUNT()
- Функции ранжирования - позволяют получить порядковые номера записей в окне:
  - RANK(), DENSE_RANK(), ROW_NUMBER(), NTILE()
- Функции сдвига - возвращают значение из другой строки окна:
  - LAG(), LEAD(), FIRST_VALUE(), LAST_VALUE()
- Аналитические функции - предоставляют информацию о распределении данных:
  - PERCENT_RANK, CUME_DIST, PERCENTILE_CONT, PERCENTILE_DISC

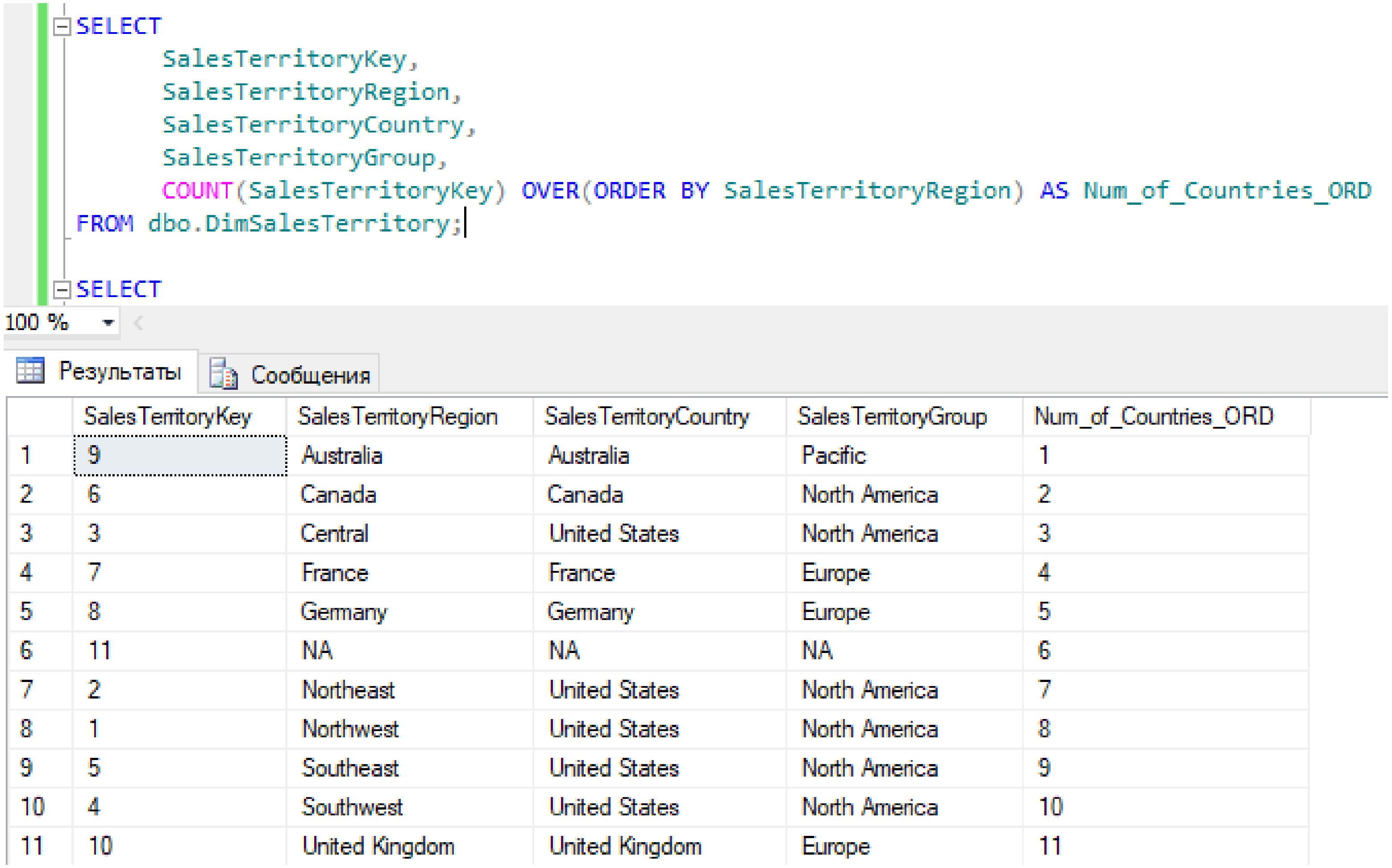
Агрегатные функции OVER()



```
---
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        COUNT(SalesTerritoryKey) OVER() AS Num_of_Countries
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries
1	1	Northwest	United States	North America	11
2	2	Northeast	United States	North America	11
3	3	Central	United States	North America	11
4	4	Southwest	United States	North America	11
5	5	Southeast	United States	North America	11
6	6	Canada	Canada	North America	11
7	7	France	France	Europe	11
8	8	Germany	Germany	Europe	11
9	9	Australia	Australia	Pacific	11
10	10	United Kingdom	United Kingdom	Europe	11
11	11	NA	NA	NA	11

OVER(ORDER BY) с нарастающим итогом



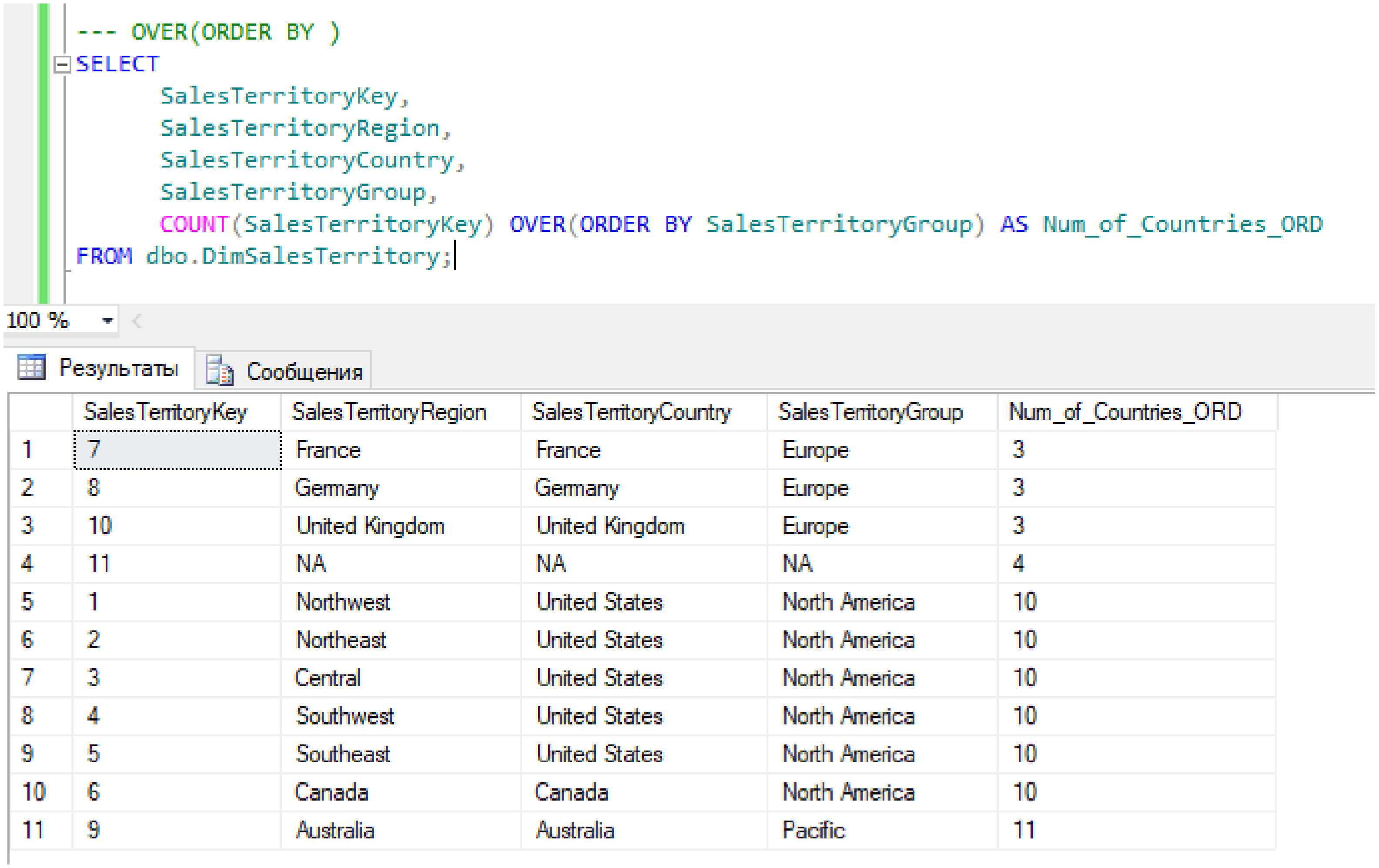
```
SELECT
       SalesTerritoryKey,
       SalesTerritoryRegion,
       SalesTerritoryCountry,
       SalesTerritoryGroup,
       COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryRegion) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;

SELECT
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	9	Australia	Australia	Pacific	1
2	6	Canada	Canada	North America	2
3	3	Central	United States	North America	3
4	7	France	France	Europe	4
5	8	Germany	Germany	Europe	5
6	11	NA	NA	NA	6
7	2	Northeast	United States	North America	7
8	1	Northwest	United States	North America	8
9	5	Southeast	United States	North America	9
10	4	Southwest	United States	North America	10
11	10	United Kingdom	United Kingdom	Europe	11

OVER(ORDER BY)
с нарастающим итогом

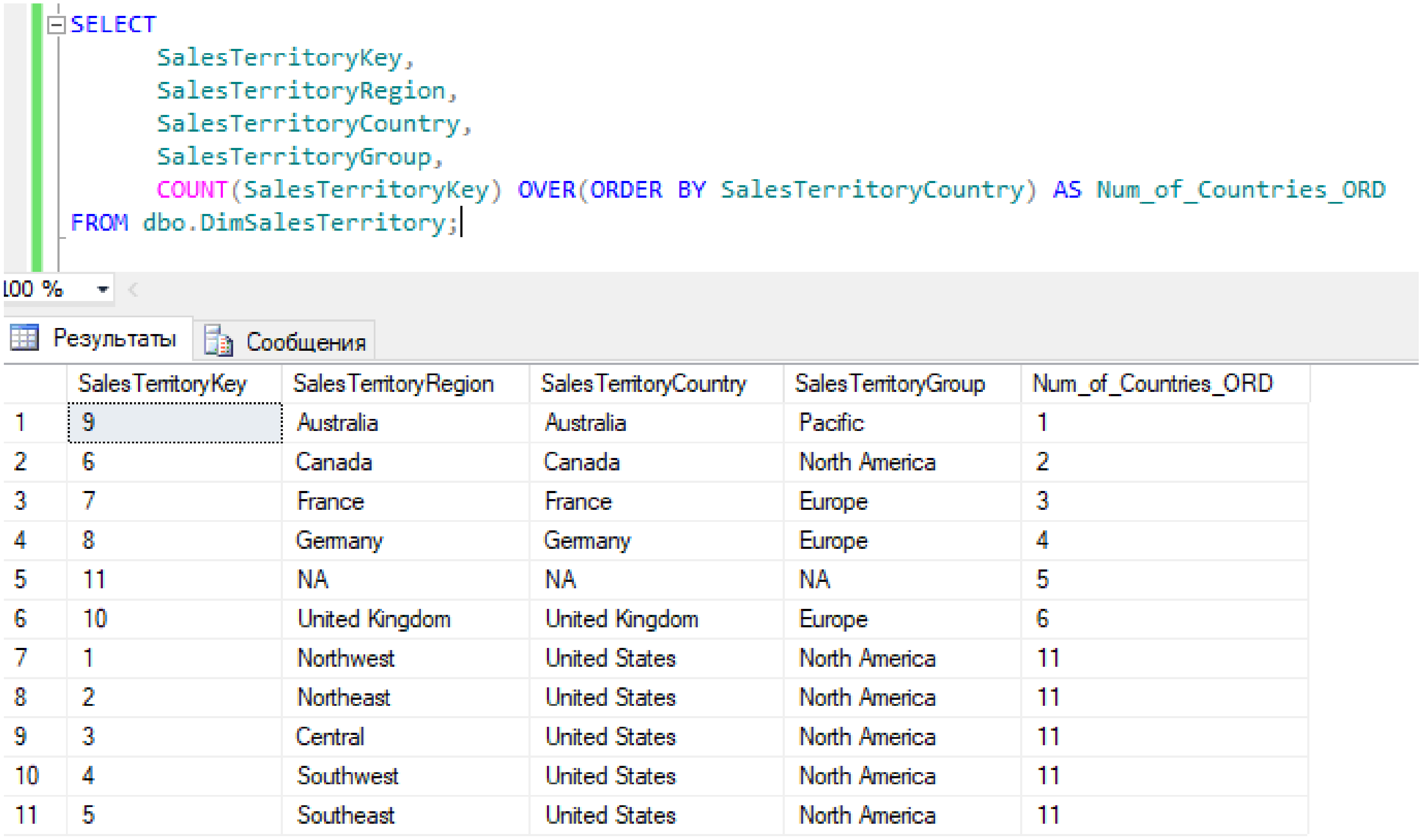


```
--- OVER(ORDER BY )
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

100 %

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	3
2	8	Germany	Germany	Europe	3
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	4
5	1	Northwest	United States	North America	10
6	2	Northeast	United States	North America	10
7	3	Central	United States	North America	10
8	4	Southwest	United States	North America	10
9	5	Southeast	United States	North America	10
10	6	Canada	Canada	North America	10
11	9	Australia	Australia	Pacific	11

Агрегатные функции OVER (ORDER BY)



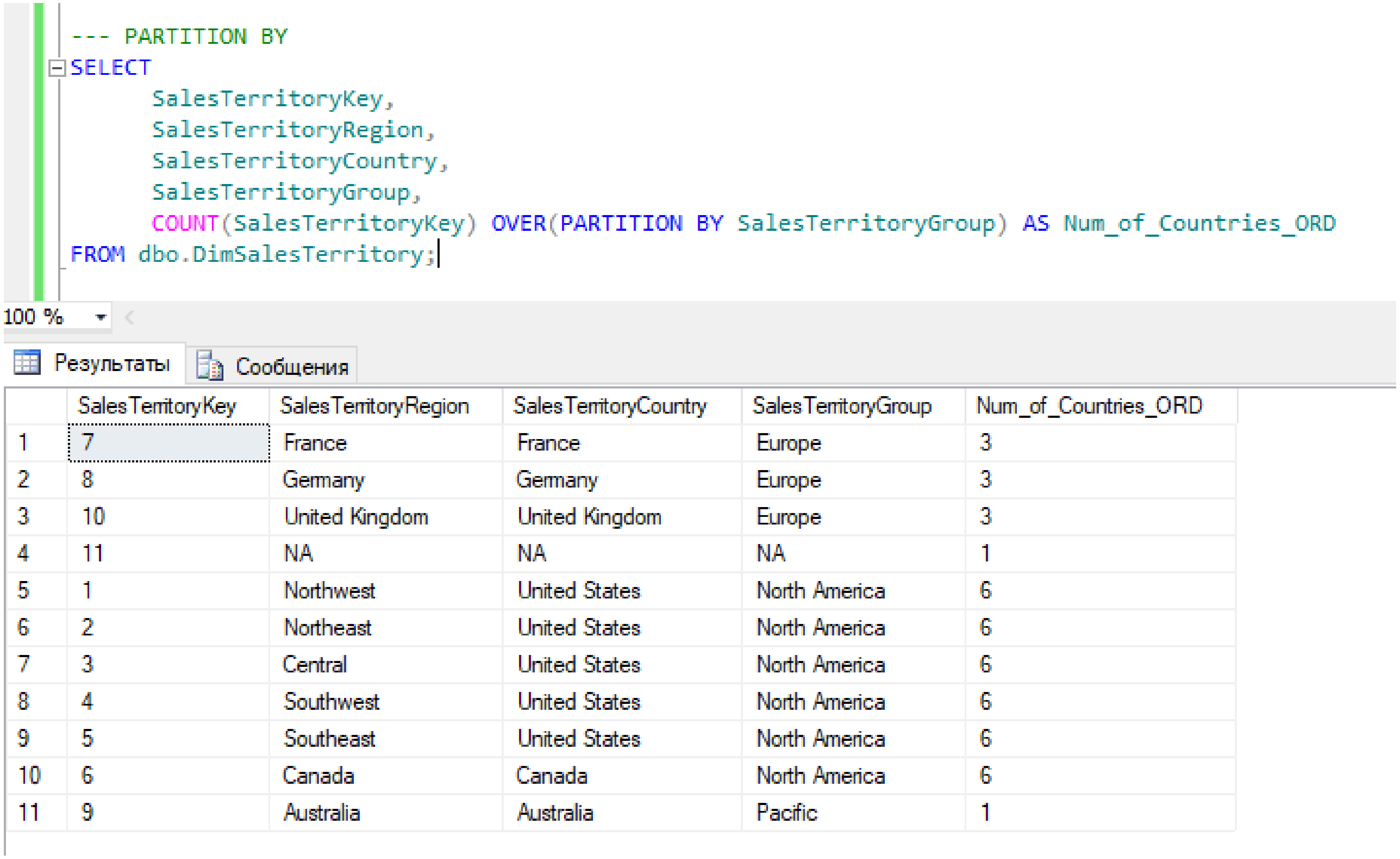
```
SELECT
       SalesTerritoryKey,
       SalesTerritoryRegion,
       SalesTerritoryCountry,
       SalesTerritoryGroup,
       COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryCountry) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	9	Australia	Australia	Pacific	1
2	6	Canada	Canada	North America	2
3	7	France	France	Europe	3
4	8	Germany	Germany	Europe	4
5	11	NA	NA	NA	5
6	10	United Kingdom	United Kingdom	Europe	6
7	1	Northwest	United States	North America	11
8	2	Northeast	United States	North America	11
9	3	Central	United States	North America	11
10	4	Southwest	United States	North America	11
11	5	Southeast	United States	North America	11

Агрегатные функции OVER (PARTITION BY)

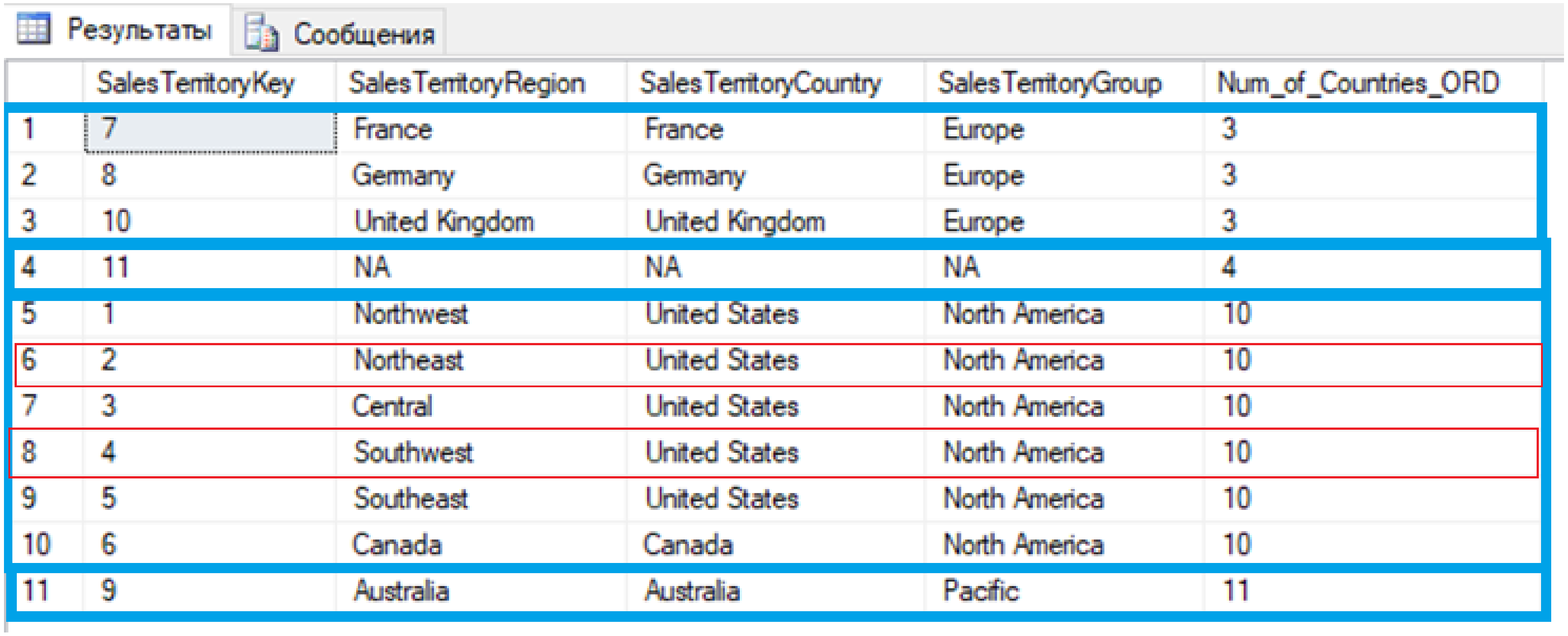


```
--- PARTITION BY
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        COUNT(SalesTerritoryKey) OVER(PARTITION BY SalesTerritoryGroup) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	3
2	8	Germany	Germany	Europe	3
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	1
5	1	Northwest	United States	North America	6
6	2	Northeast	United States	North America	6
7	3	Central	United States	North America	6
8	4	Southwest	United States	North America	6
9	5	Southeast	United States	North America	6
10	6	Canada	Canada	North America	6
11	9	Australia	Australia	Pacific	1

Строки и диапазоны

- ROWS – строки результирующего набора
- RANGE – диапазоны результирующего набора

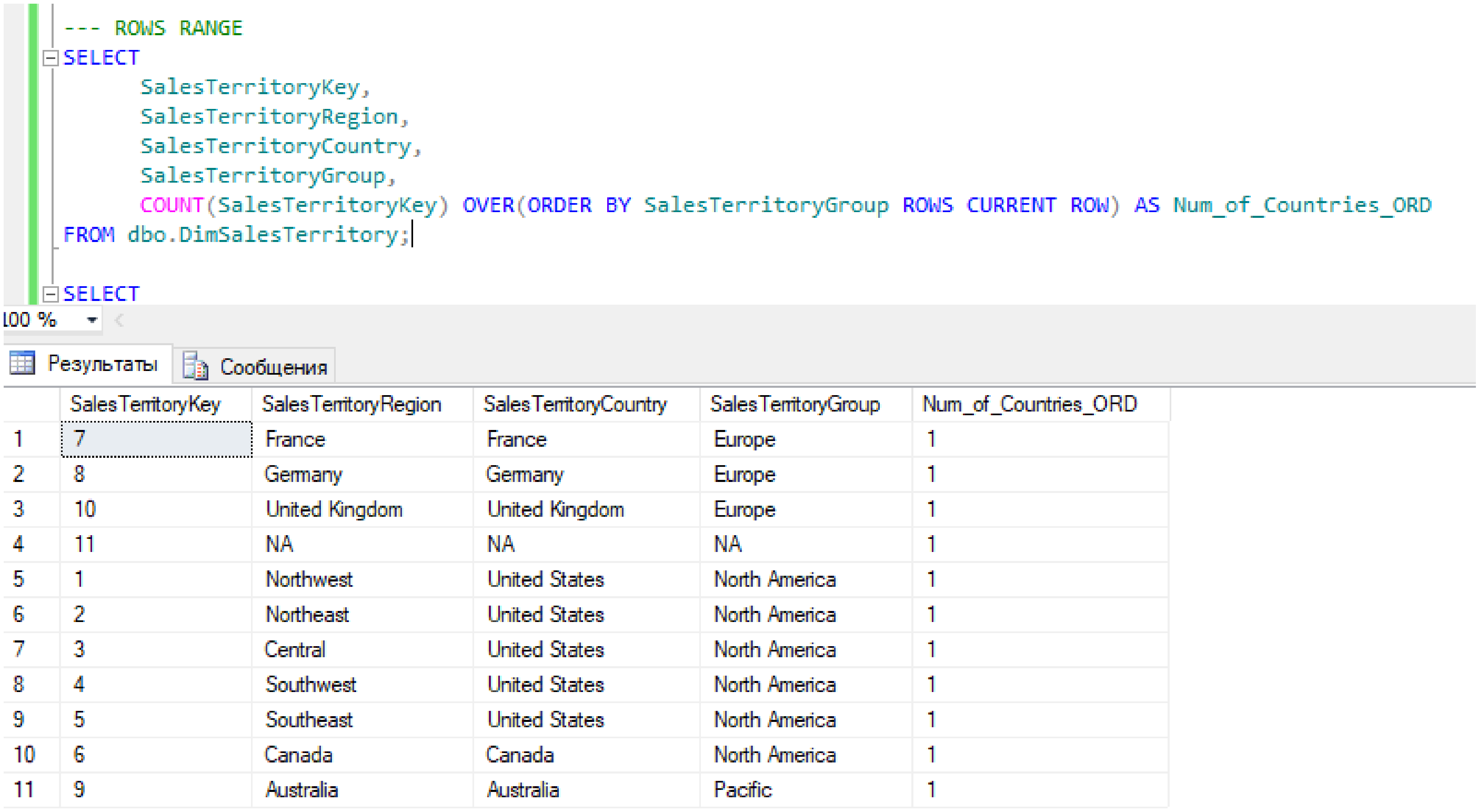


	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	3
2	8	Germany	Germany	Europe	3
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	4
5	1	Northwest	United States	North America	10
6	2	Northeast	United States	North America	10
7	3	Central	United States	North America	10
8	4	Southwest	United States	North America	10
9	5	Southeast	United States	North America	10
10	6	Canada	Canada	North America	10
11	9	Australia	Australia	Pacific	11

Строки и диапазоны

- ROWS ограничивает строки в окне **фиксированным количеством строк**
- ROWS и RANGE используются вместе с ORDER BY
- Методы (можно комбинировать):
  - CURRENT ROW –текущая строка
  - UNBOUNDED FOLLOWING – все записи после текущей
  - UNBOUNDED PRECEDING – все предыдущие записи
  - <N> PRECEDING – заданное число предыдущих строк
  - <N> FOLLOWING – заданное число последующих строк

OVER(ORDER BY ROWS)



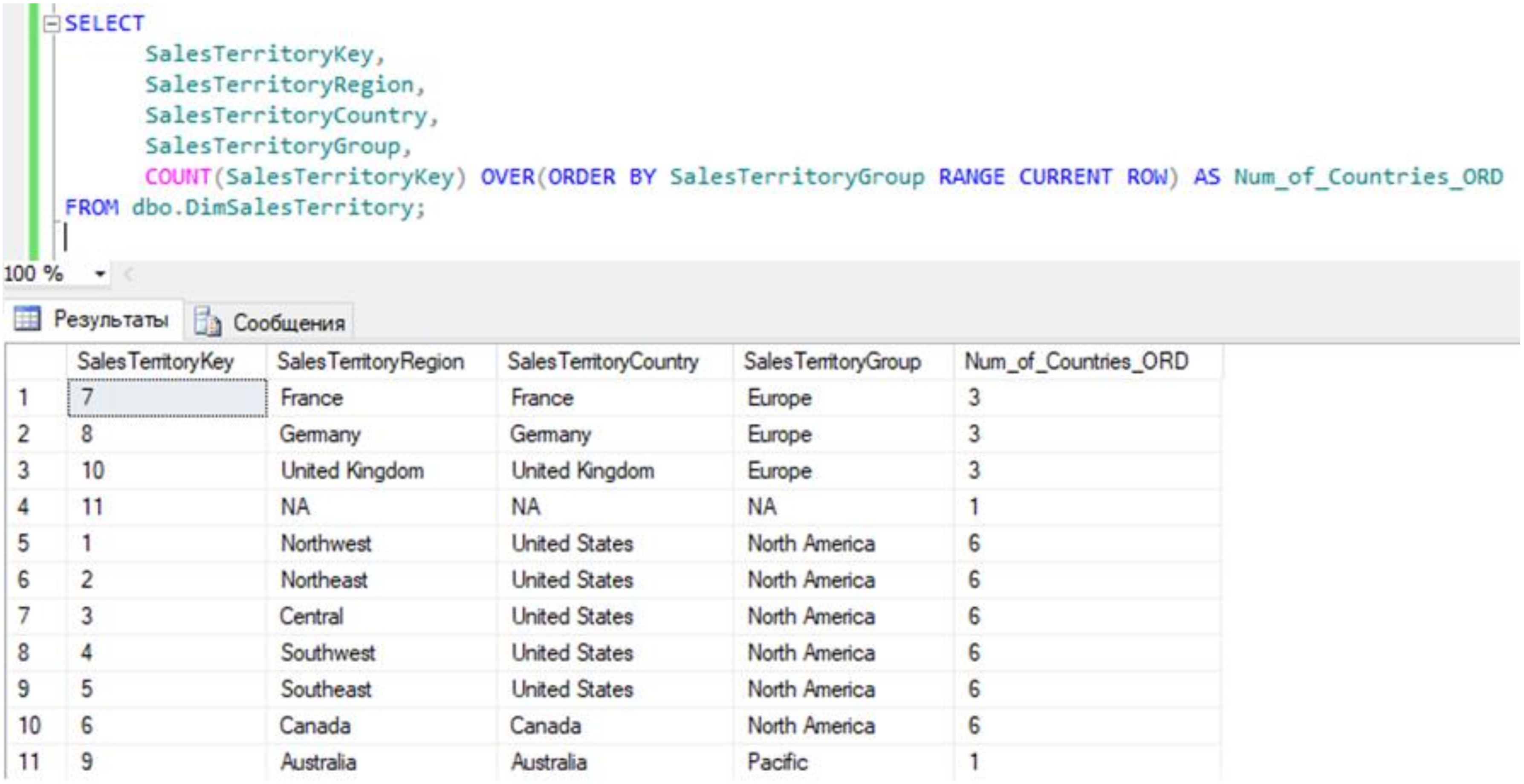
```
--- ROWS RANGE
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup ROWS CURRENT ROW) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;

SELECT
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	1
2	8	Germany	Germany	Europe	1
3	10	United Kingdom	United Kingdom	Europe	1
4	11	NA	NA	NA	1
5	1	Northwest	United States	North America	1
6	2	Northeast	United States	North America	1
7	3	Central	United States	North America	1
8	4	Southwest	United States	North America	1
9	5	Southeast	United States	North America	1
10	6	Canada	Canada	North America	1
11	9	Australia	Australia	Pacific	1

OVER(ORDER BY RANGE)

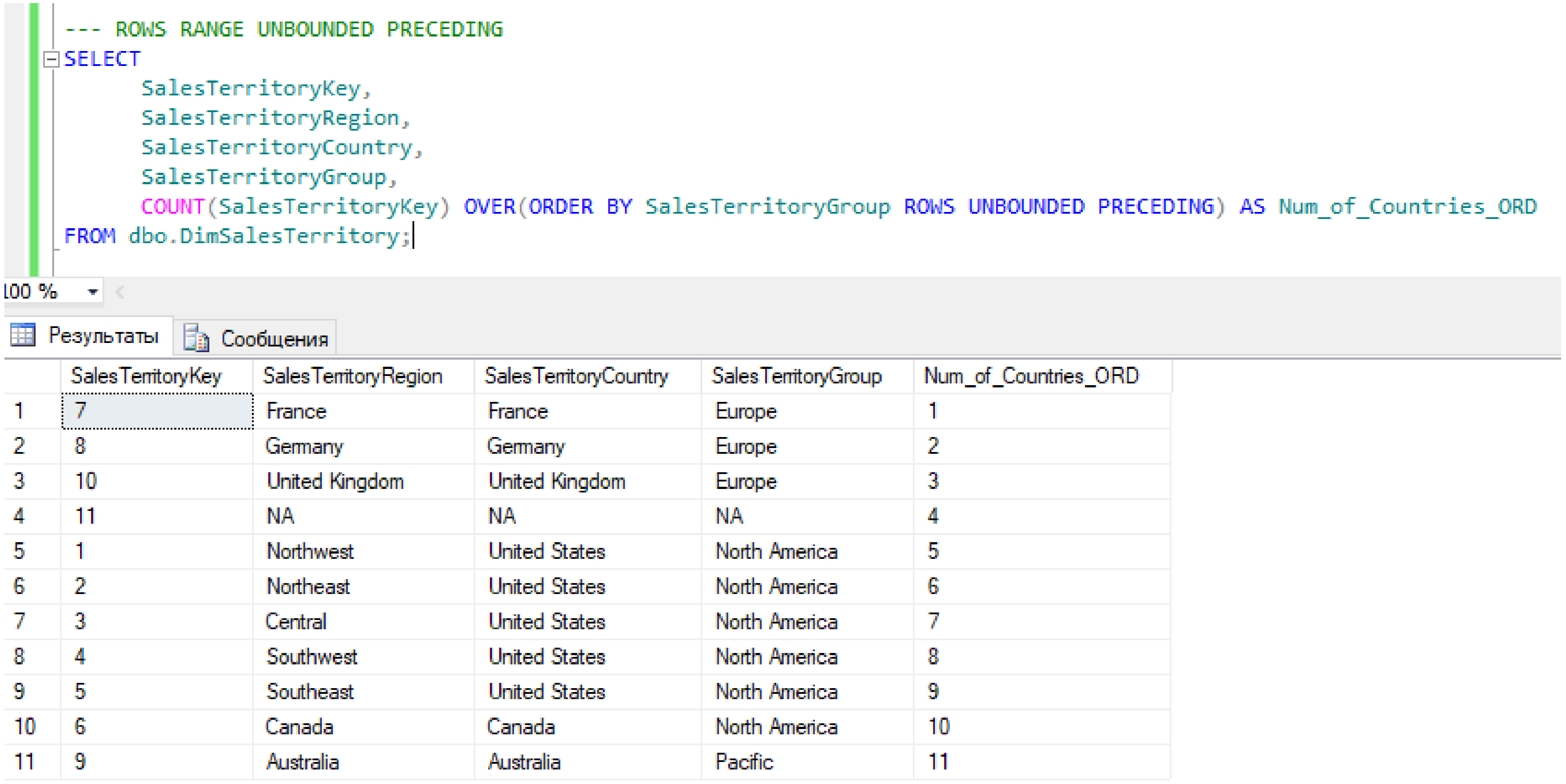


```
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup RANGE CURRENT ROW) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	3
2	8	Germany	Germany	Europe	3
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	1
5	1	Northwest	United States	North America	6
6	2	Northeast	United States	North America	6
7	3	Central	United States	North America	6
8	4	Southwest	United States	North America	6
9	5	Southeast	United States	North America	6
10	6	Canada	Canada	North America	6
11	9	Australia	Australia	Pacific	1

UNBOUNDED PRECEDING

- UNBOUNDED PRECEDING – учитываются предыдущие строки до текущей включительно



```
--- ROWS RANGE UNBOUNDED PRECEDING
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup ROWS UNBOUNDED PRECEDING) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

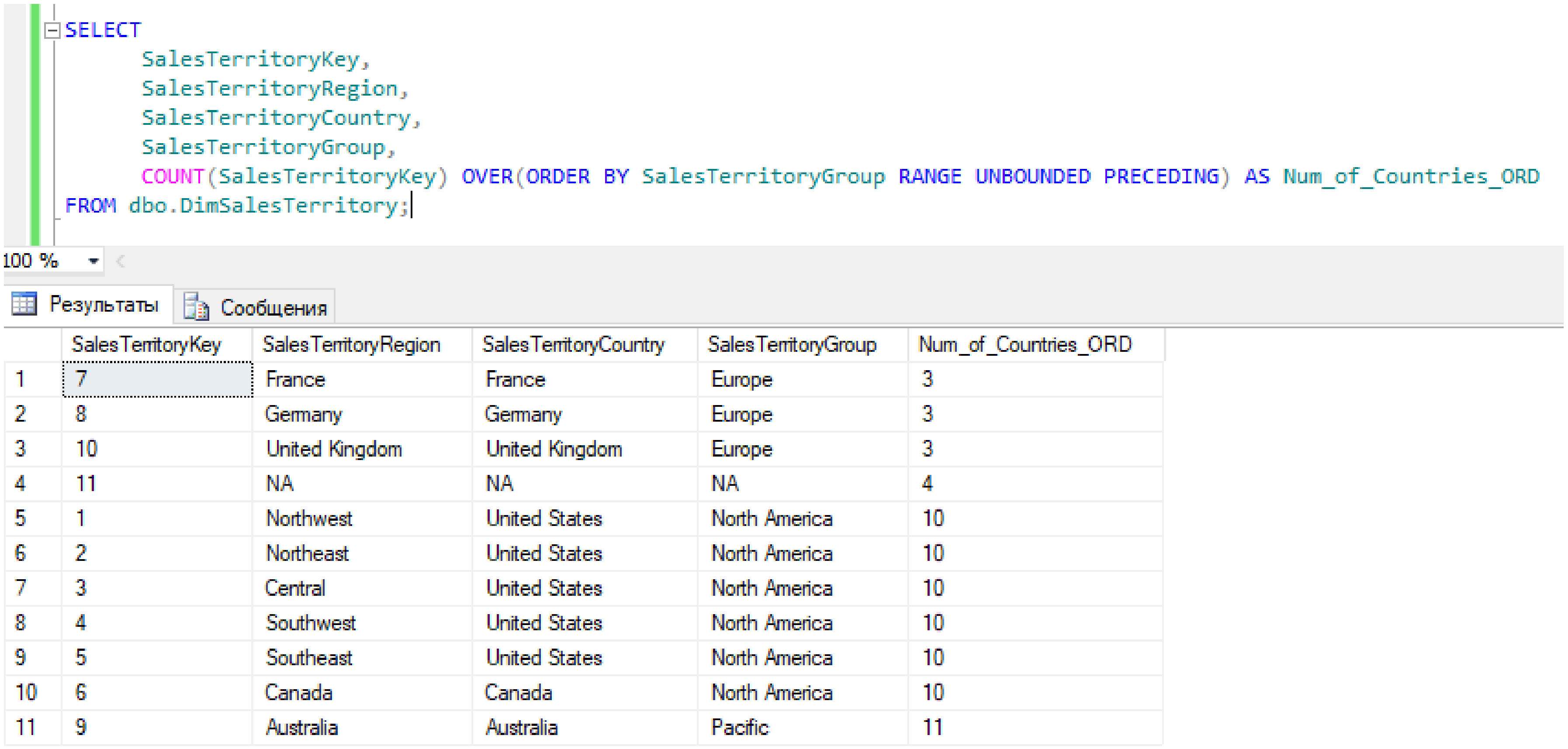
100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	1
2	8	Germany	Germany	Europe	2
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	4
5	1	Northwest	United States	North America	5
6	2	Northeast	United States	North America	6
7	3	Central	United States	North America	7
8	4	Southwest	United States	North America	8
9	5	Southeast	United States	North America	9
10	6	Canada	Canada	North America	10
11	9	Australia	Australia	Pacific	11

UNBOUNDED PRECEDING

- UNBOUNDED PRECEDING – учитываются предыдущие строки до текущей включительно



```
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup RANGE UNBOUNDED PRECEDING) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

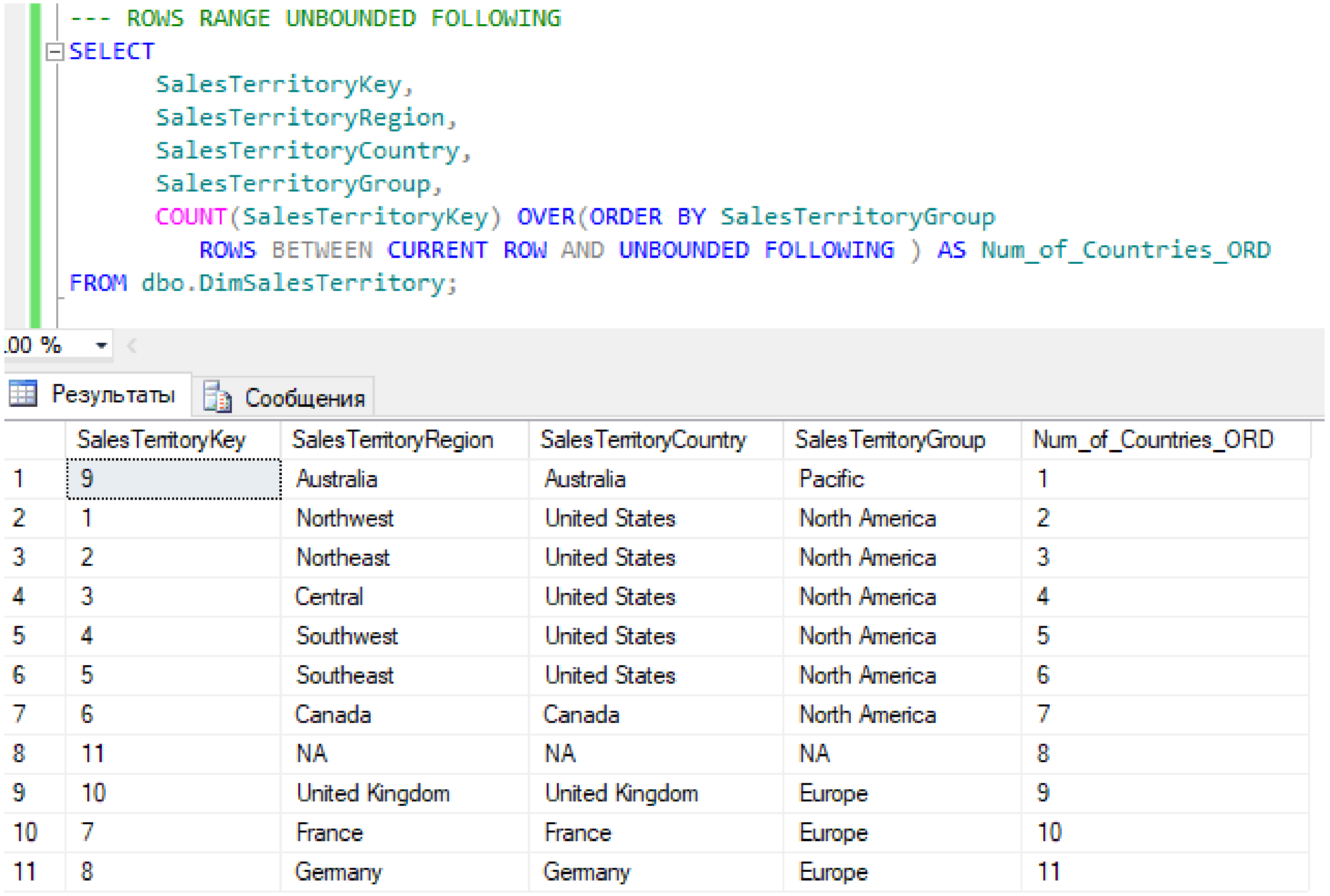
100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	3
2	8	Germany	Germany	Europe	3
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	4
5	1	Northwest	United States	North America	10
6	2	Northeast	United States	North America	10
7	3	Central	United States	North America	10
8	4	Southwest	United States	North America	10
9	5	Southeast	United States	North America	10
10	6	Canada	Canada	North America	10
11	9	Australia	Australia	Pacific	11

UNBOUNDED FOLLOWING

- UNBOUNDED FOLLOWING – учитываются последующие строки от текущей включительно



```
--- ROWS RANGE UNBOUNDED FOLLOWING
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup
         ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING ) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

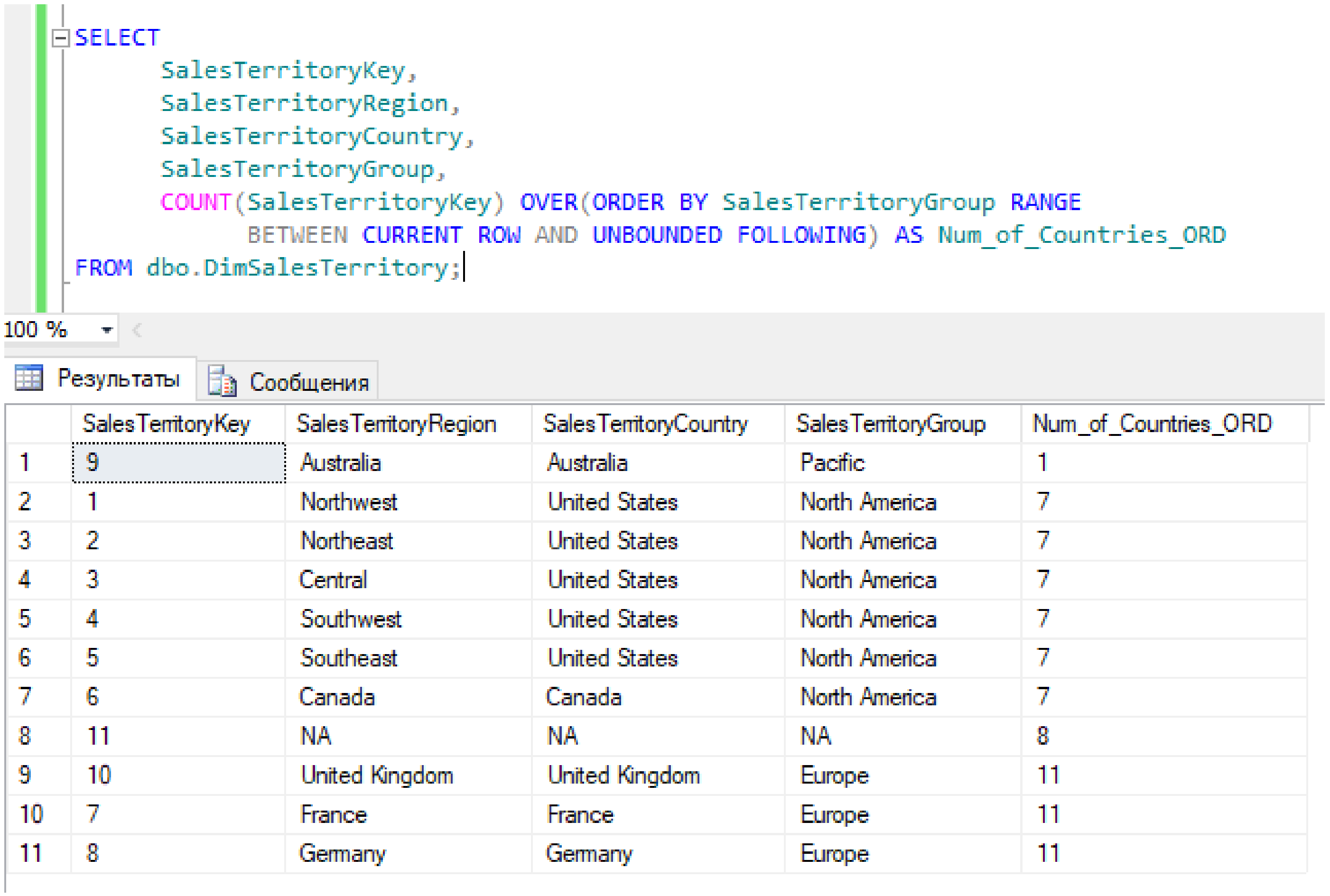
.00 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	9	Australia	Australia	Pacific	1
2	1	Northwest	United States	North America	2
3	2	Northeast	United States	North America	3
4	3	Central	United States	North America	4
5	4	Southwest	United States	North America	5
6	5	Southeast	United States	North America	6
7	6	Canada	Canada	North America	7
8	11	NA	NA	NA	8
9	10	United Kingdom	United Kingdom	Europe	9
10	7	France	France	Europe	10
11	8	Germany	Germany	Europe	11

UNBOUNDED FOLLOWING

- UNBOUNDED FOLLOWING – учитываются последующие строки от текущей включительно



```
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup RANGE
            BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

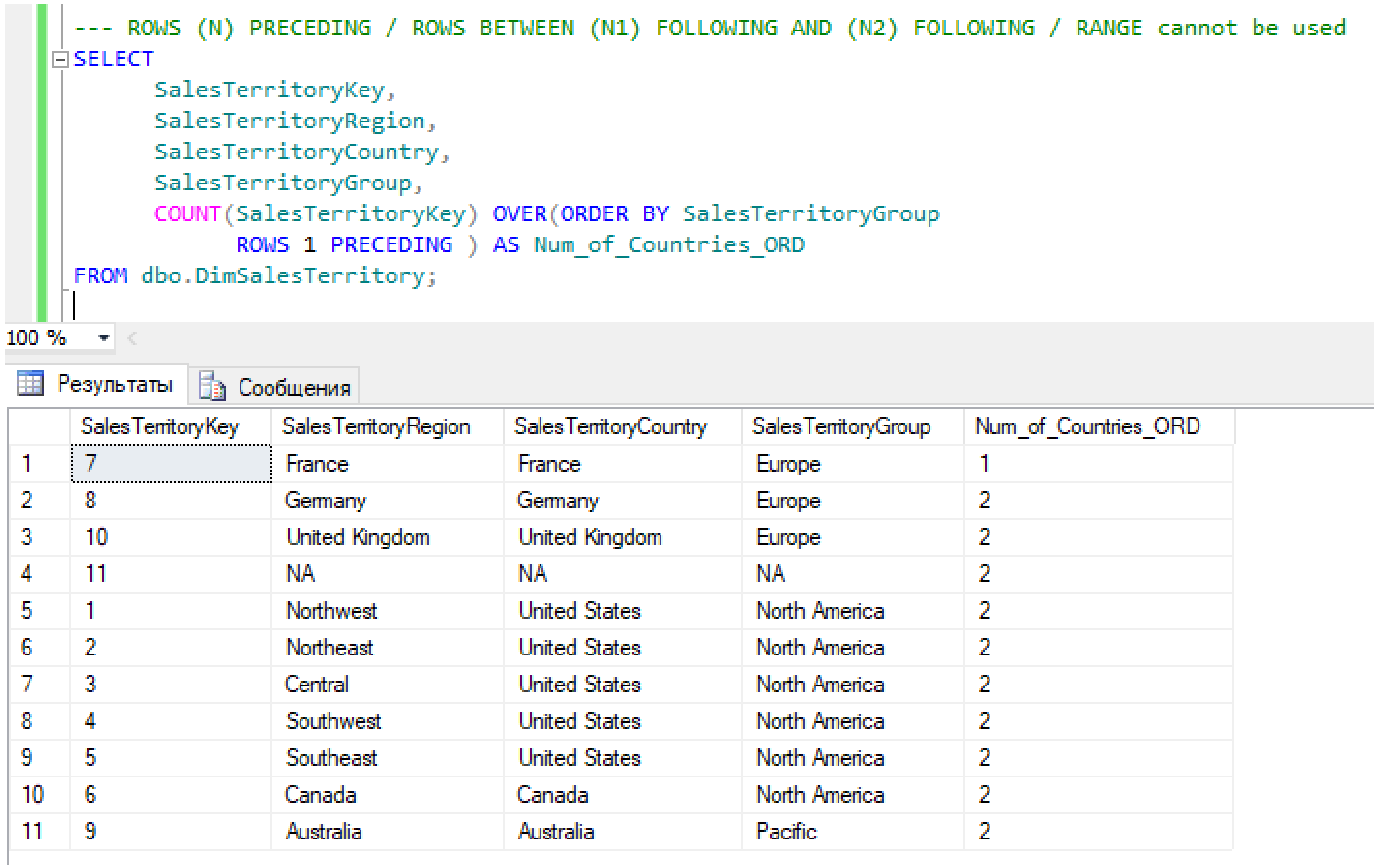
100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	9	Australia	Australia	Pacific	1
2	1	Northwest	United States	North America	7
3	2	Northeast	United States	North America	7
4	3	Central	United States	North America	7
5	4	Southwest	United States	North America	7
6	5	Southeast	United States	North America	7
7	6	Canada	Canada	North America	7
8	11	NA	NA	NA	8
9	10	United Kingdom	United Kingdom	Europe	11
10	7	France	France	Europe	11
11	8	Germany	Germany	Europe	11

ROWS (N) PRECEDING

- ROWS (N) PRECEDING – учитывается N последующих строк от текущей включительно



```
--- ROWS (N) PRECEDING / ROWS BETWEEN (N1) FOLLOWING AND (N2) FOLLOWING / RANGE cannot be used
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup
            ROWS 1 PRECEDING ) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

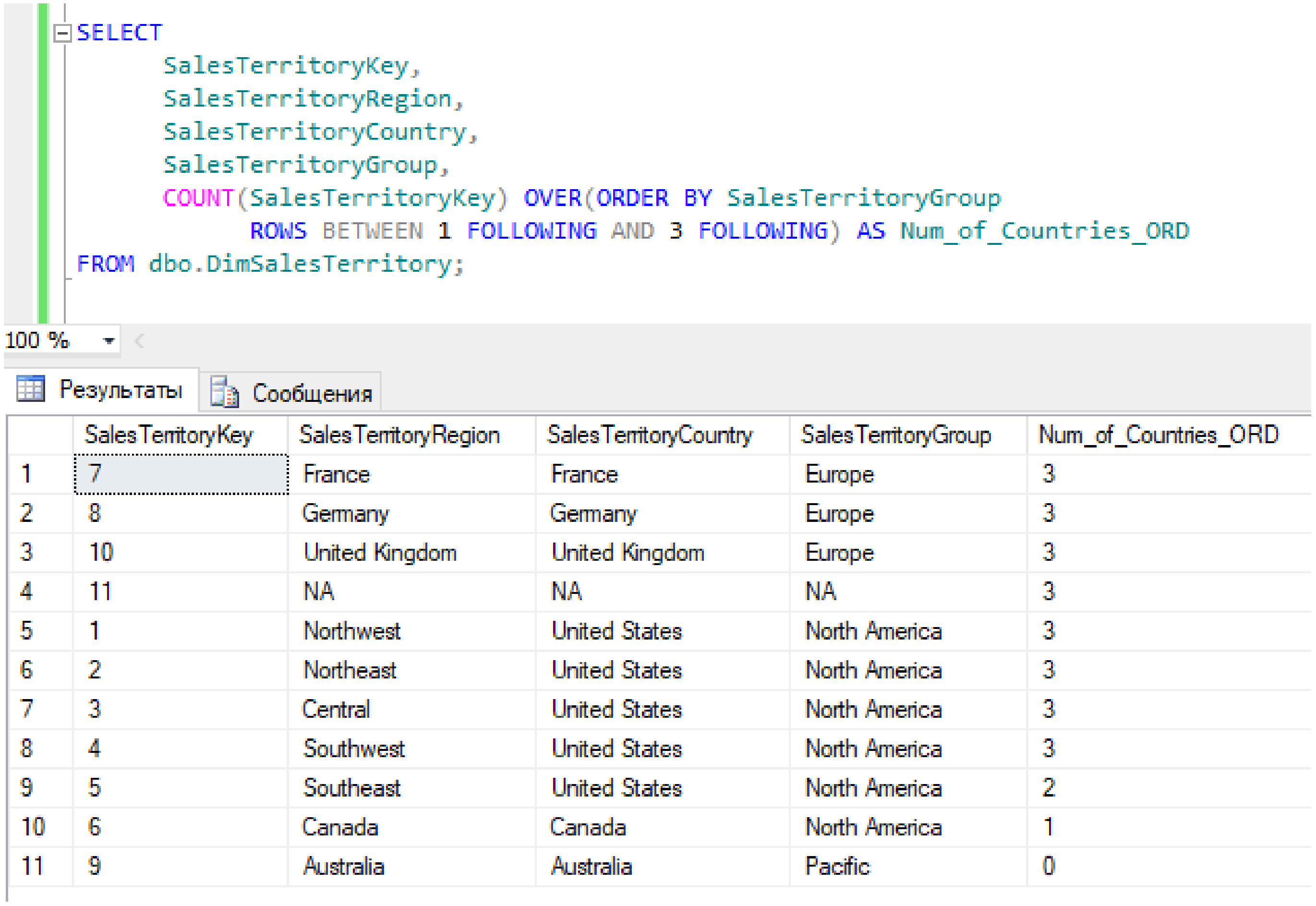
100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	1
2	8	Germany	Germany	Europe	2
3	10	United Kingdom	United Kingdom	Europe	2
4	11	NA	NA	NA	2
5	1	Northwest	United States	North America	2
6	2	Northeast	United States	North America	2
7	3	Central	United States	North America	2
8	4	Southwest	United States	North America	2
9	5	Southeast	United States	North America	2
10	6	Canada	Canada	North America	2
11	9	Australia	Australia	Pacific	2

ROWS FOLLOWING

- ROWS BETWEEN (N1) FOLLOWING AND (N2) FOLLOWING – учитываются строки с N1 до N2



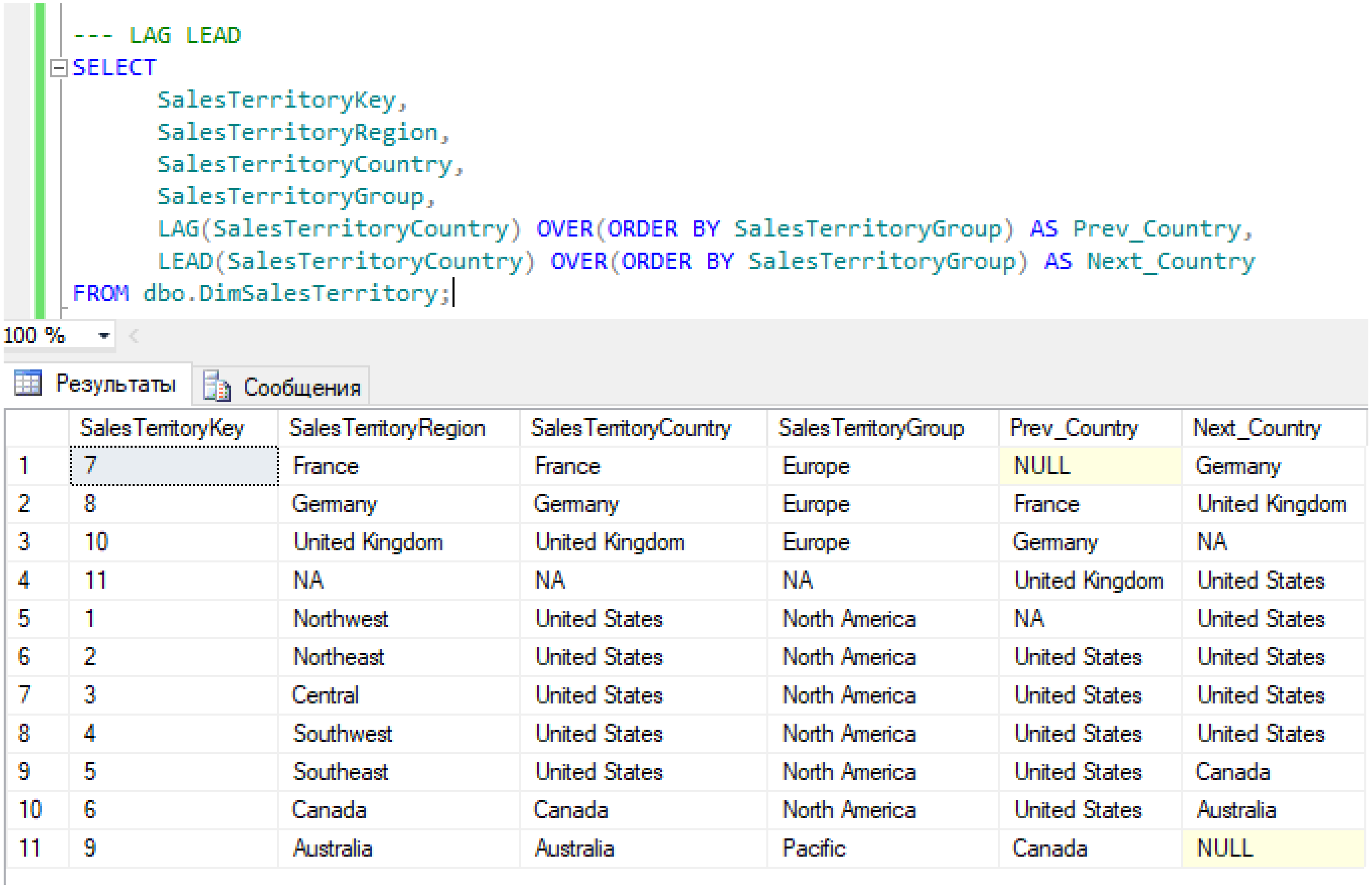
```
SELECT
       SalesTerritoryKey,
       SalesTerritoryRegion,
       SalesTerritoryCountry,
       SalesTerritoryGroup,
       COUNT(SalesTerritoryKey) OVER(ORDER BY SalesTerritoryGroup
             ROWS BETWEEN 1 FOLLOWING AND 3 FOLLOWING) AS Num_of_Countries_ORD
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Num_of_Countries_ORD
1	7	France	France	Europe	3
2	8	Germany	Germany	Europe	3
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	3
5	1	Northwest	United States	North America	3
6	2	Northeast	United States	North America	3
7	3	Central	United States	North America	3
8	4	Southwest	United States	North America	3
9	5	Southeast	United States	North America	2
10	6	Canada	Canada	North America	1
11	9	Australia	Australia	Pacific	0

LAG() и LEAD()



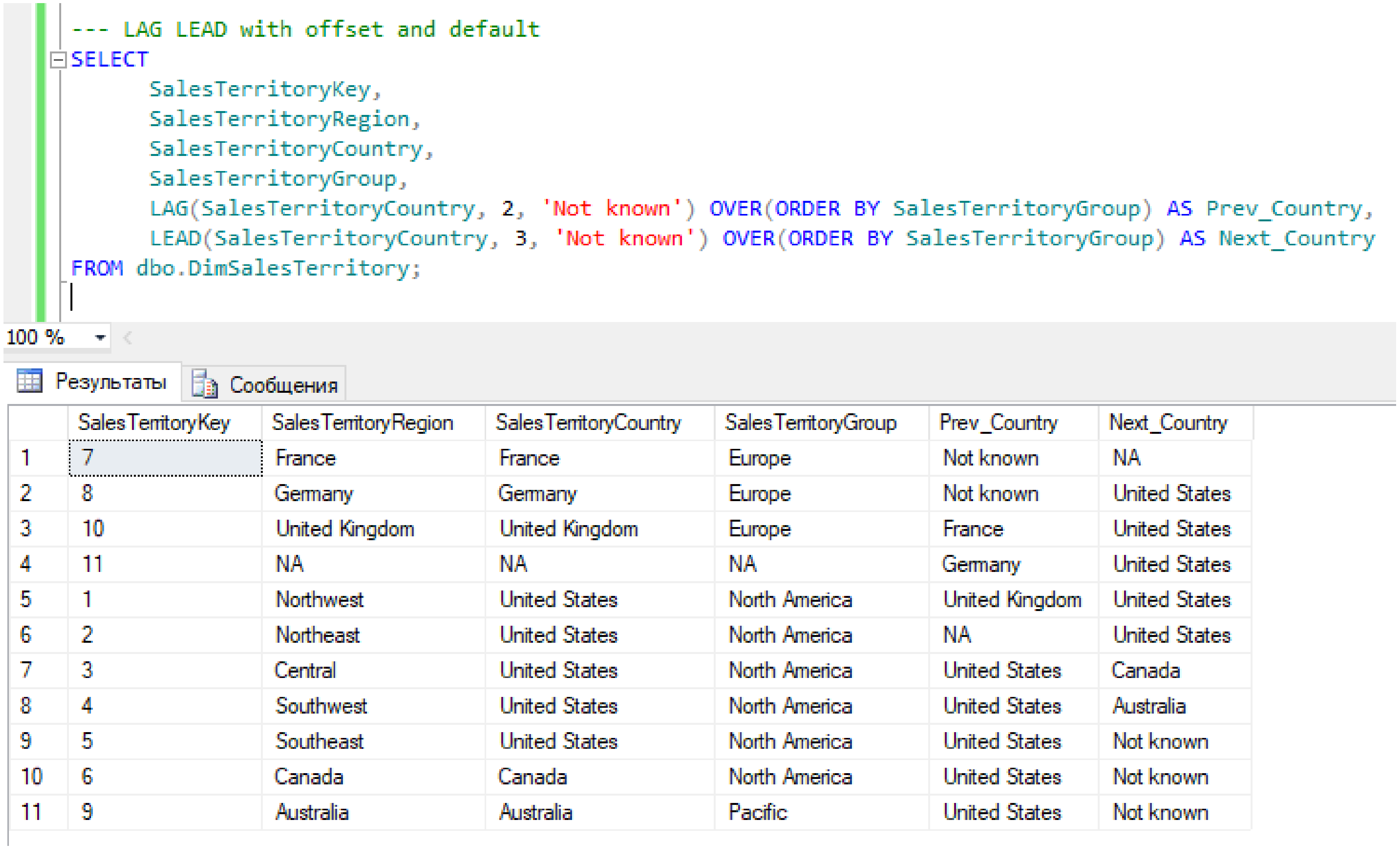
```
--- LAG LEAD
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      LAG(SalesTerritoryCountry) OVER(ORDER BY SalesTerritoryGroup) AS Prev_Country,
      LEAD(SalesTerritoryCountry) OVER(ORDER BY SalesTerritoryGroup) AS Next_Country
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Prev_Country	Next_Country
1	7	France	France	Europe	NULL	Germany
2	8	Germany	Germany	Europe	France	United Kingdom
3	10	United Kingdom	United Kingdom	Europe	Germany	NA
4	11	NA	NA	NA	United Kingdom	United States
5	1	Northwest	United States	North America	NA	United States
6	2	Northeast	United States	North America	United States	United States
7	3	Central	United States	North America	United States	United States
8	4	Southwest	United States	North America	United States	United States
9	5	Southeast	United States	North America	United States	Canada
10	6	Canada	Canada	North America	United States	Australia
11	9	Australia	Australia	Pacific	Canada	NULL

LAG() и LEAD()



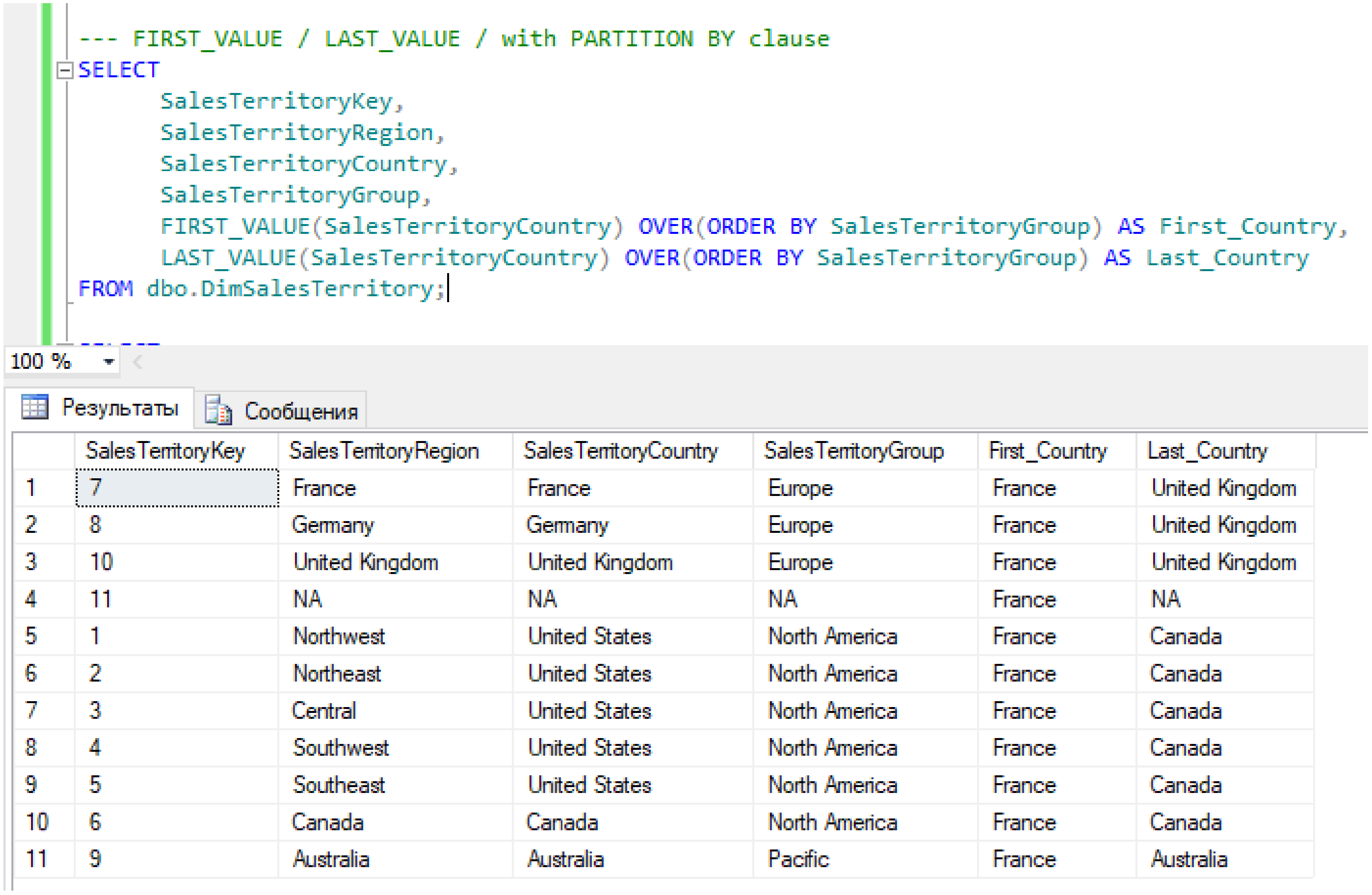
```
--- LAG LEAD with offset and default
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        LAG(SalesTerritoryCountry, 2, 'Not known') OVER(ORDER BY SalesTerritoryGroup) AS Prev_Country,
        LEAD(SalesTerritoryCountry, 3, 'Not known') OVER(ORDER BY SalesTerritoryGroup) AS Next_Country
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Prev_Country	Next_Country
1	7	France	France	Europe	Not known	NA
2	8	Germany	Germany	Europe	Not known	United States
3	10	United Kingdom	United Kingdom	Europe	France	United States
4	11	NA	NA	NA	Germany	United States
5	1	Northwest	United States	North America	United Kingdom	United States
6	2	Northeast	United States	North America	NA	United States
7	3	Central	United States	North America	United States	Canada
8	4	Southwest	United States	North America	United States	Australia
9	5	Southeast	United States	North America	United States	Not known
10	6	Canada	Canada	North America	United States	Not known
11	9	Australia	Australia	Pacific	United States	Not known

FIRST_VALUE() и LAST_VALUE()



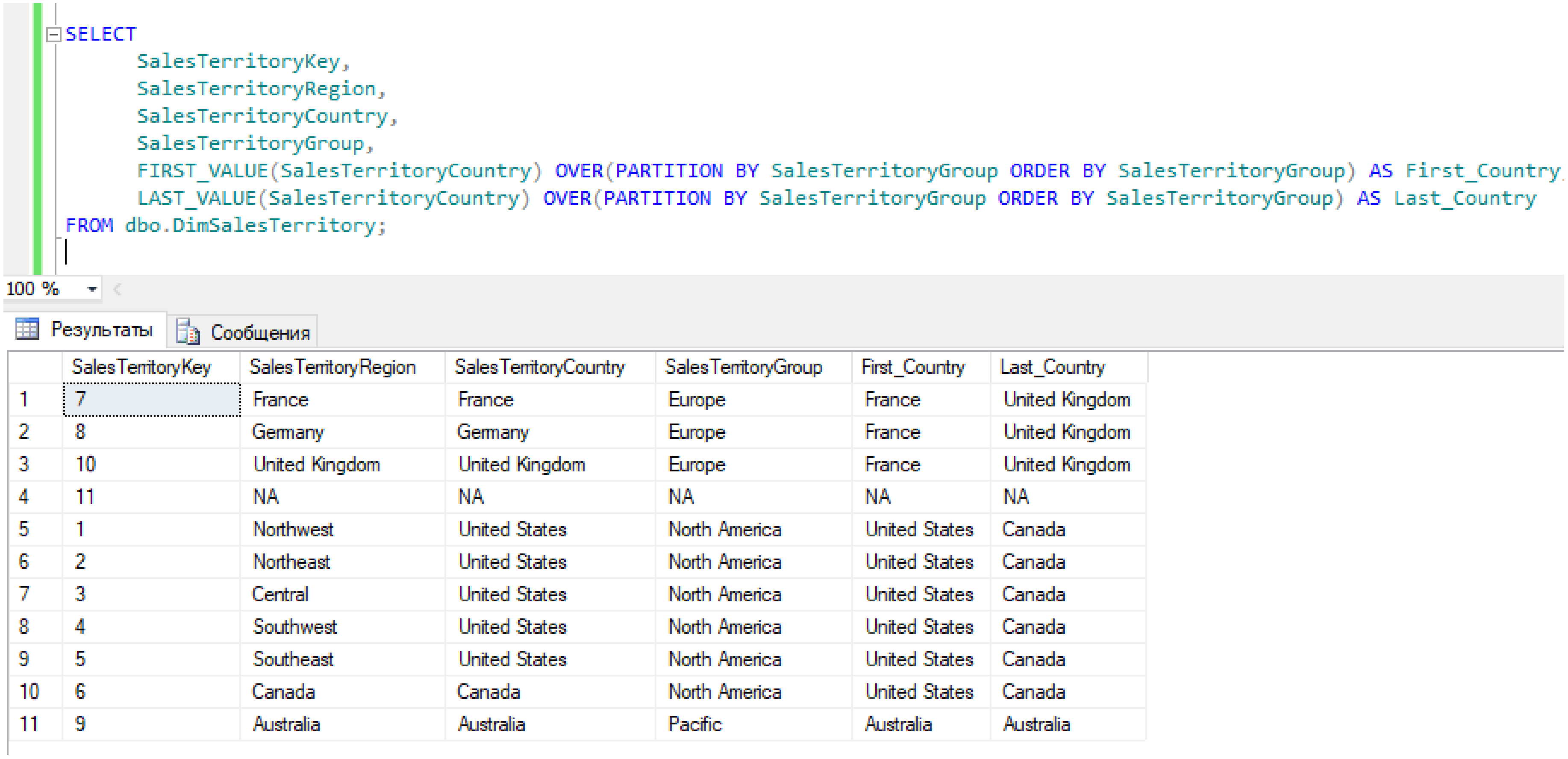
```
--- FIRST_VALUE / LAST_VALUE / with PARTITION BY clause
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      FIRST_VALUE(SalesTerritoryCountry) OVER(ORDER BY SalesTerritoryGroup) AS First_Country,
      LAST_VALUE(SalesTerritoryCountry) OVER(ORDER BY SalesTerritoryGroup) AS Last_Country
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	First_Country	Last_Country
1	7	France	France	Europe	France	United Kingdom
2	8	Germany	Germany	Europe	France	United Kingdom
3	10	United Kingdom	United Kingdom	Europe	France	United Kingdom
4	11	NA	NA	NA	France	NA
5	1	Northwest	United States	North America	France	Canada
6	2	Northeast	United States	North America	France	Canada
7	3	Central	United States	North America	France	Canada
8	4	Southwest	United States	North America	France	Canada
9	5	Southeast	United States	North America	France	Canada
10	6	Canada	Canada	North America	France	Canada
11	9	Australia	Australia	Pacific	France	Australia

FIRST_VALUE() и LAST_VALUE()



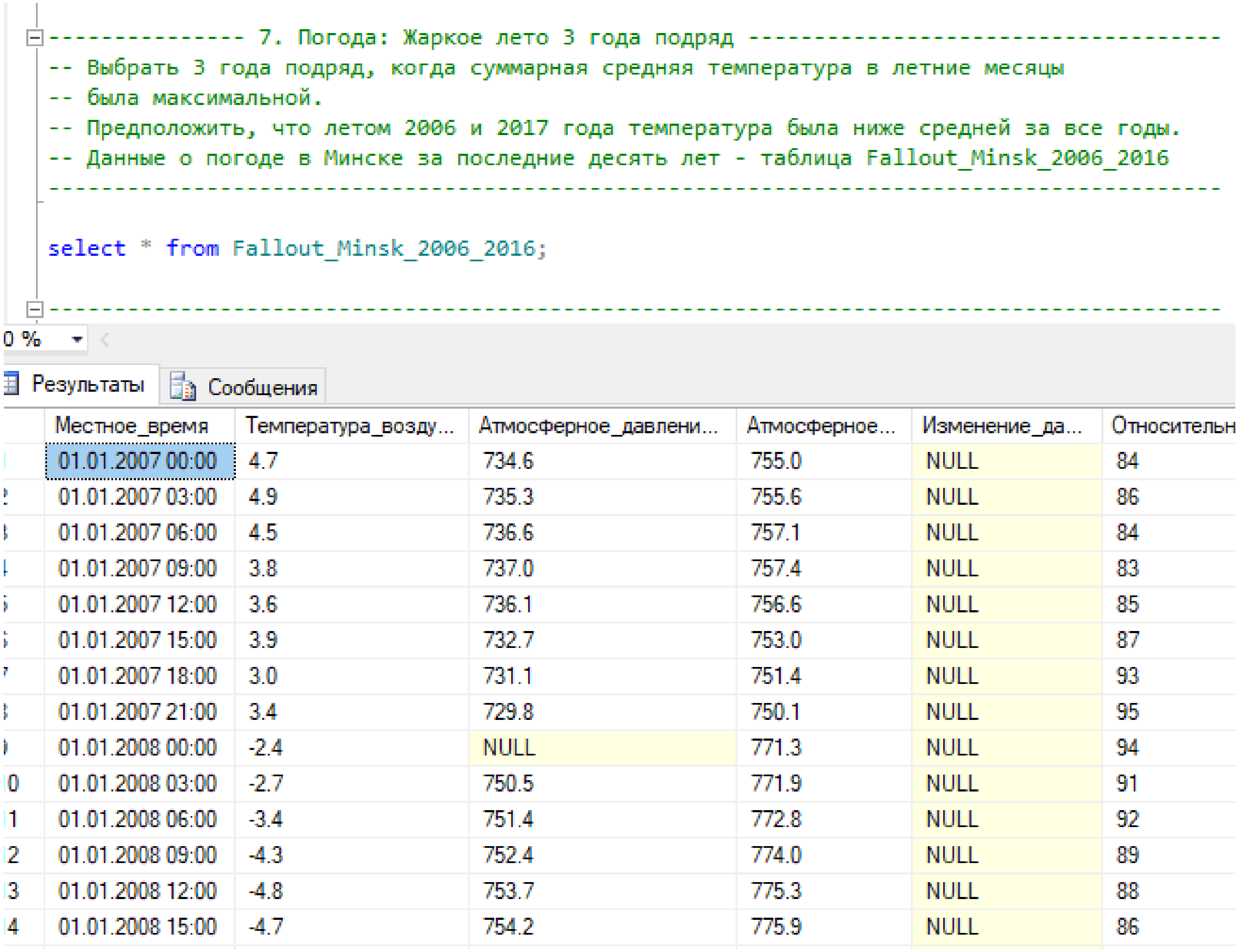
```
SELECT
	SalesTerritoryKey,
	SalesTerritoryRegion,
	SalesTerritoryCountry,
	SalesTerritoryGroup,
	FIRST_VALUE(SalesTerritoryCountry) OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS First_Country,
	LAST_VALUE(SalesTerritoryCountry) OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Last_Country
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	First_Country	Last_Country
1	7	France	France	Europe	France	United Kingdom
2	8	Germany	Germany	Europe	France	United Kingdom
3	10	United Kingdom	United Kingdom	Europe	France	United Kingdom
4	11	NA	NA	NA	NA	NA
5	1	Northwest	United States	North America	United States	Canada
6	2	Northeast	United States	North America	United States	Canada
7	3	Central	United States	North America	United States	Canada
8	4	Southwest	United States	North America	United States	Canada
9	5	Southeast	United States	North America	United States	Canada
10	6	Canada	Canada	North America	United States	Canada
11	9	Australia	Australia	Pacific	Australia	Australia

Пример



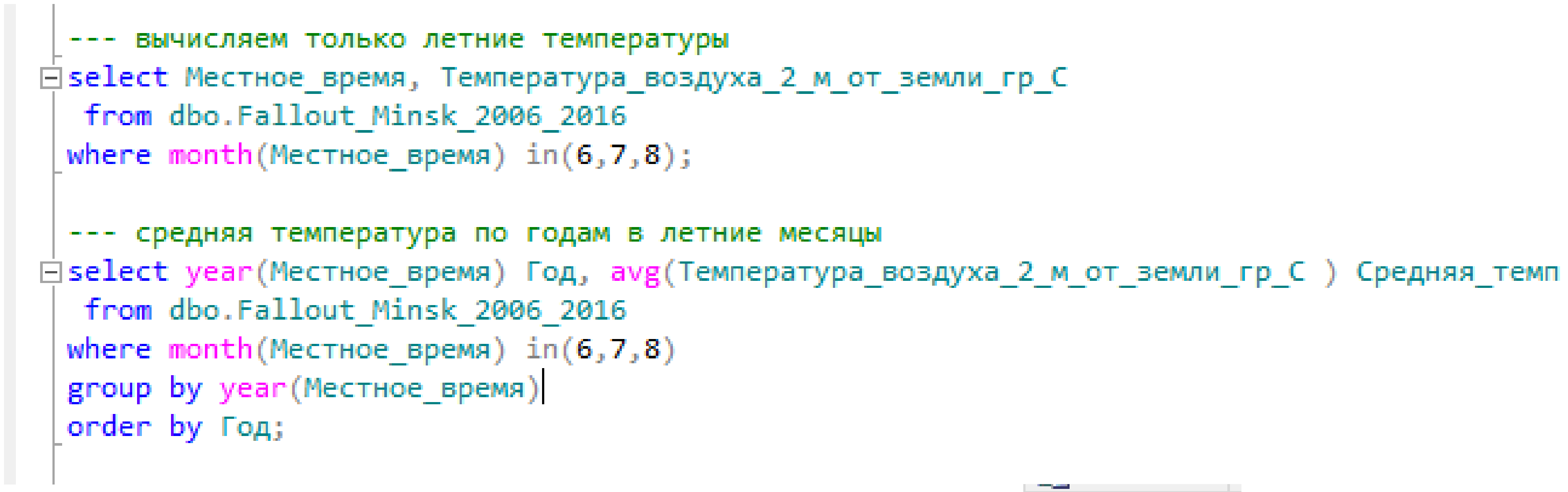
```
--------------- 7. Погода: Жаркое лето 3 года подряд -----------------------------------
-- Выбрать 3 года подряд, когда суммарная средняя температура в летние месяцы
-- была максимальной.
-- Предположить, что летом 2006 и 2017 года температура была ниже средней за все годы.
-- Данные о погоде в Минске за последние десять лет - таблица Fallout_Minsk_2006_2016
-----------------------------------------------------------------------------------------

select * from Fallout_Minsk_2006_2016;

-----------------------------------------------------------------------------------------
```

0 %

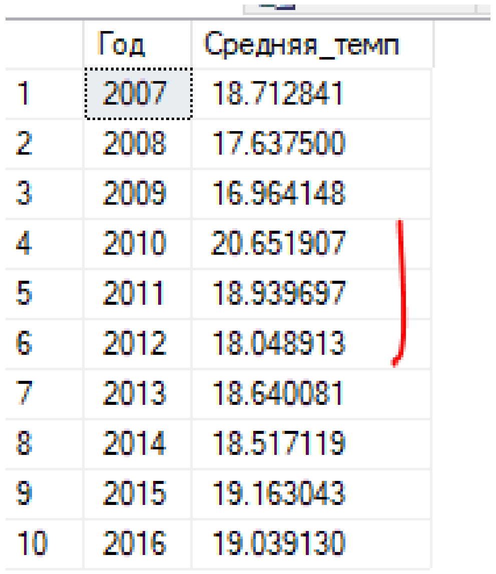
Результаты | Сообщения

	Местное_время	Температура_возду...	Атмосферное_давлени...	Атмосферное...	Изменение_да...	Относительн
	01.01.2007 00:00	4.7	734.6	755.0	NULL	84
	01.01.2007 03:00	4.9	735.3	755.6	NULL	86
	01.01.2007 06:00	4.5	736.6	757.1	NULL	84
	01.01.2007 09:00	3.8	737.0	757.4	NULL	83
	01.01.2007 12:00	3.6	736.1	756.6	NULL	85
	01.01.2007 15:00	3.9	732.7	753.0	NULL	87
	01.01.2007 18:00	3.0	731.1	751.4	NULL	93
	01.01.2007 21:00	3.4	729.8	750.1	NULL	95
	01.01.2008 00:00	-2.4	NULL	771.3	NULL	94
0	01.01.2008 03:00	-2.7	750.5	771.9	NULL	91
1	01.01.2008 06:00	-3.4	751.4	772.8	NULL	92
2	01.01.2008 09:00	-4.3	752.4	774.0	NULL	89
3	01.01.2008 12:00	-4.8	753.7	775.3	NULL	88
4	01.01.2008 15:00	-4.7	754.2	775.9	NULL	86

Пример

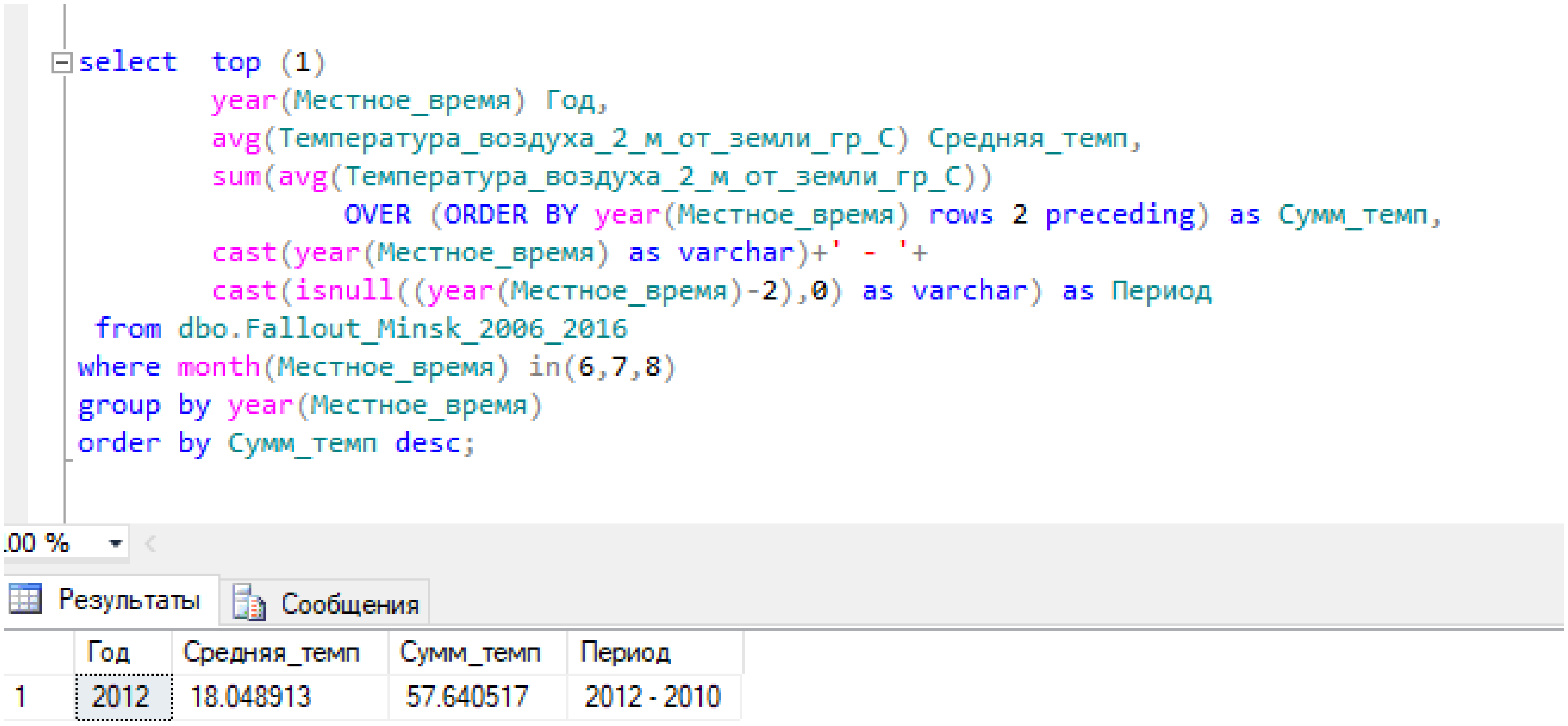


```
--- вычисляем только летние температуры
select Местное_время, Температура_воздуха_2_м_от_земли_гр_С
 from dbo.Fallout_Minsk_2006_2016
where month(Местное_время) in(6,7,8);

--- средняя температура по годам в летние месяцы
select year(Местное_время) Год, avg(Температура_воздуха_2_м_от_земли_гр_С ) Средняя_темп
 from dbo.Fallout_Minsk_2006_2016
where month(Местное_время) in(6,7,8)
group by year(Местное_время)
order by Год;
```



	Год	Средняя_темп
1	2007	18.712841
2	2008	17.637500
3	2009	16.964148
4	2010	20.651907
5	2011	18.939697
6	2012	18.048913
7	2013	18.640081
8	2014	18.517119
9	2015	19.163043
10	2016	19.039130

Пример



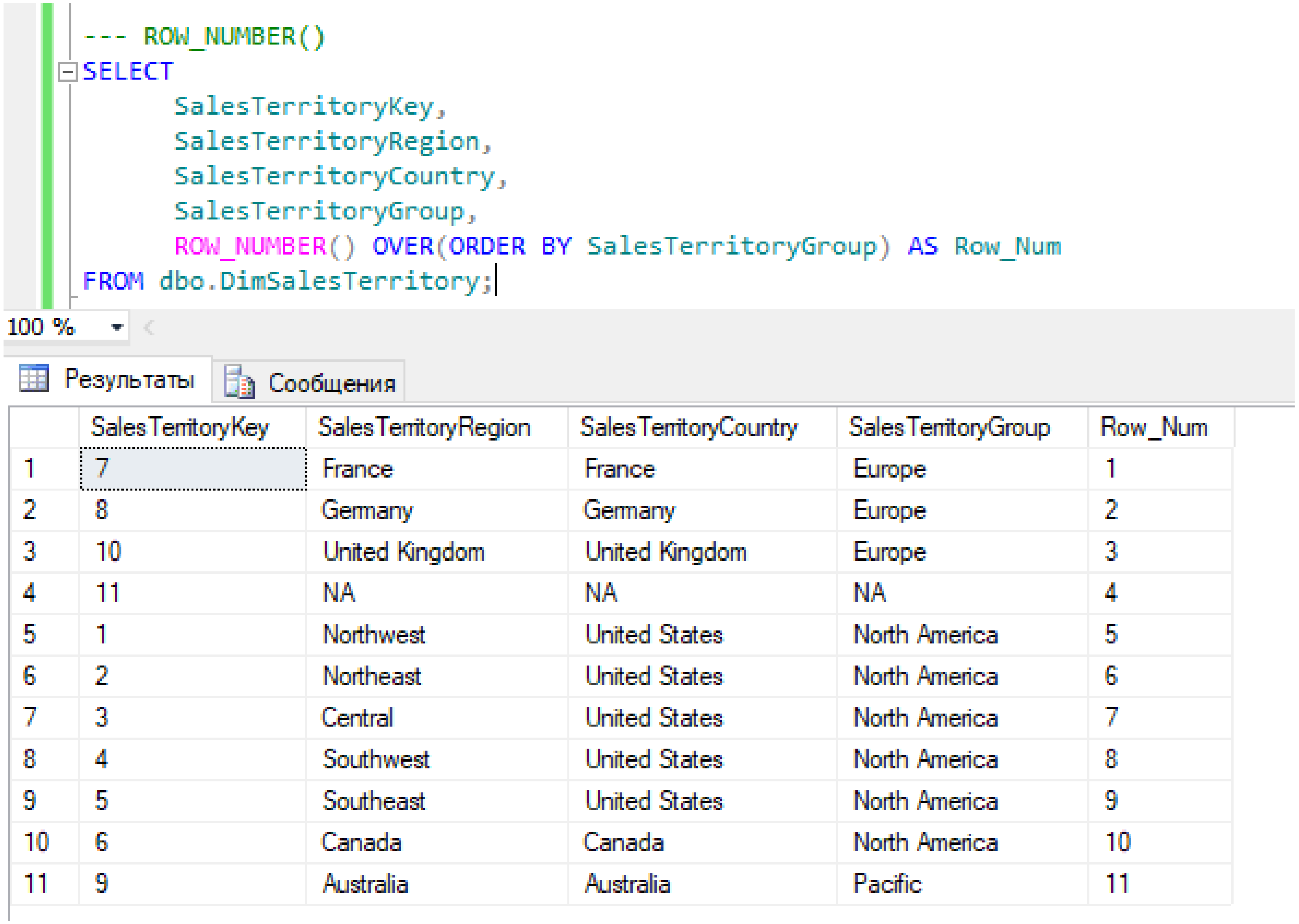
```
select  top (1)
        year(Местное_время) Год,
        avg(Температура_воздуха_2_м_от_земли_гр_C) Средняя_темп,
        sum(avg(Температура_воздуха_2_м_от_земли_гр_C))
                OVER (ORDER BY year(Местное_время) rows 2 preceding) as Сумм_темп,
        cast(year(Местное_время) as varchar)+' - '+
        cast(isnull((year(Местное_время)-2),0) as varchar) as Период
 from dbo.Fallout_Minsk_2006_2016
where month(Местное_время) in(6,7,8)
group by year(Местное_время)
order by Сумм_темп desc;
```

	Год	Средняя_темп	Сумм_темп	Период
1	2012	18.048913	57.640517	2012 - 2010

Функции ранжирования

- ROW_NUMBER()
- RANK()
- DENSE_RANK()
- NTILE()

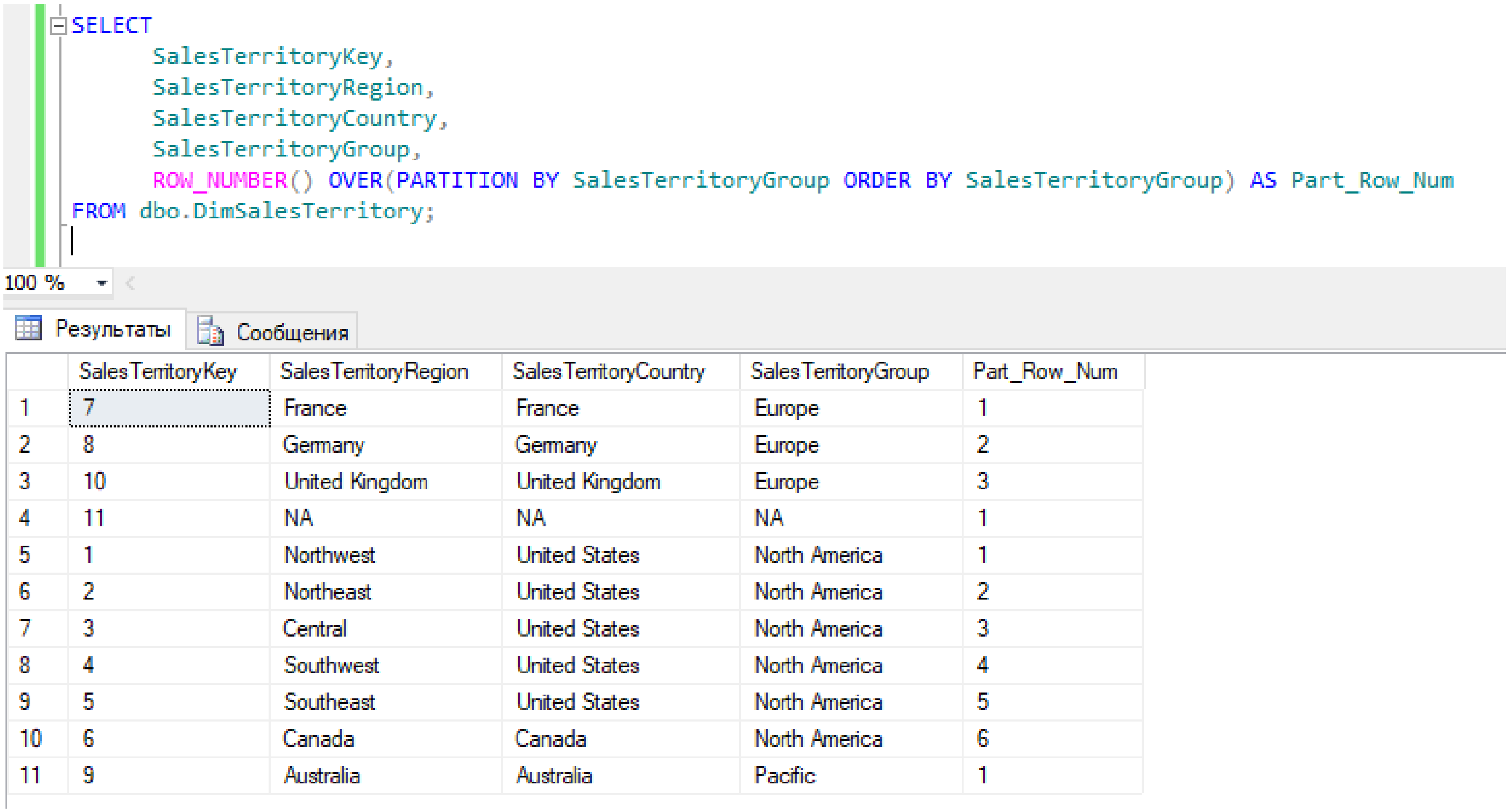
ROW_NUMBER()



```
--- ROW_NUMBER()
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        ROW_NUMBER() OVER(ORDER BY SalesTerritoryGroup) AS Row_Num
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Row_Num
1	7	France	France	Europe	1
2	8	Germany	Germany	Europe	2
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	4
5	1	Northwest	United States	North America	5
6	2	Northeast	United States	North America	6
7	3	Central	United States	North America	7
8	4	Southwest	United States	North America	8
9	5	Southeast	United States	North America	9
10	6	Canada	Canada	North America	10
11	9	Australia	Australia	Pacific	11

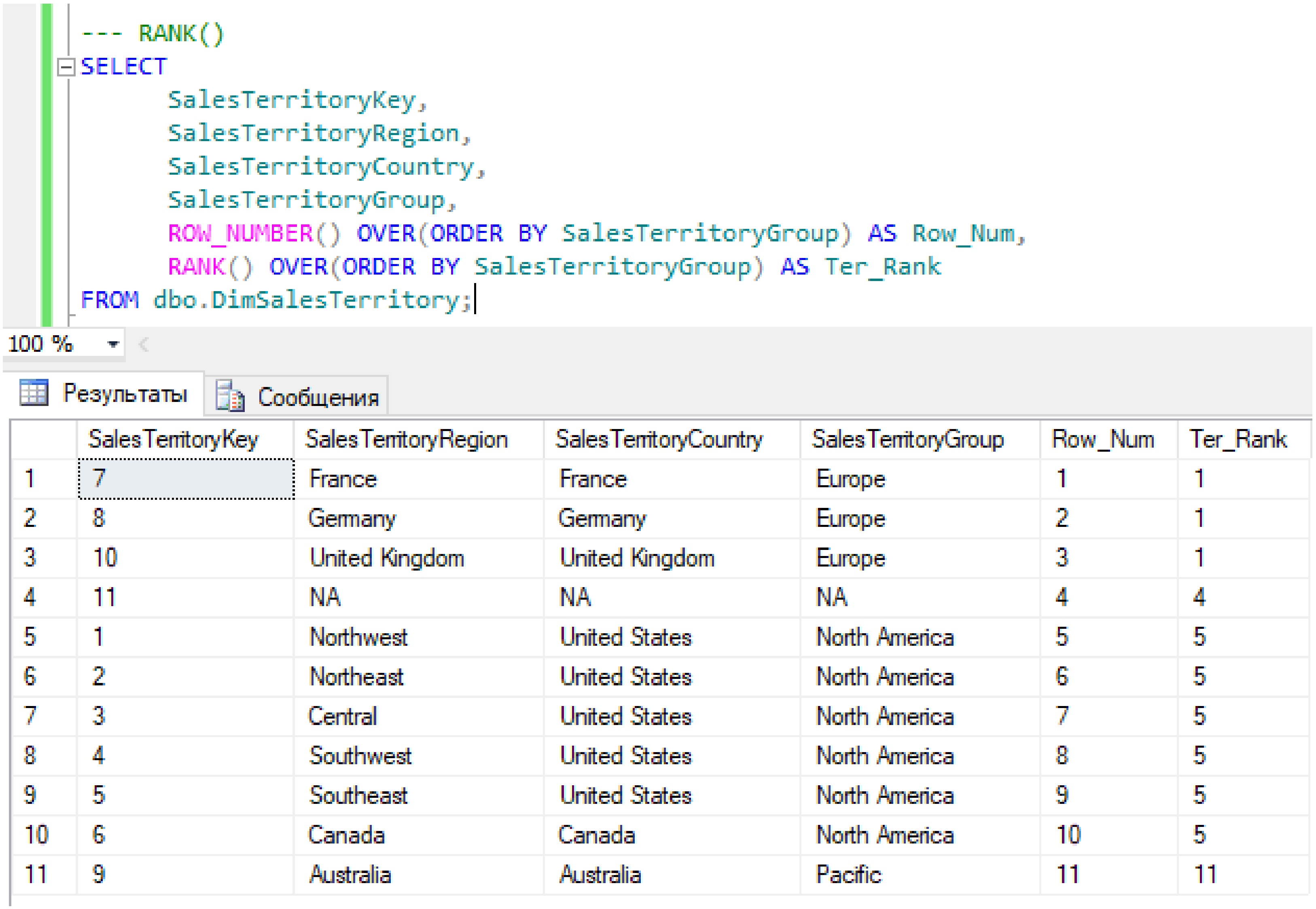
ROW_NUMBER()



```
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        ROW_NUMBER() OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Part_Row_Num
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Part_Row_Num
1	7	France	France	Europe	1
2	8	Germany	Germany	Europe	2
3	10	United Kingdom	United Kingdom	Europe	3
4	11	NA	NA	NA	1
5	1	Northwest	United States	North America	1
6	2	Northeast	United States	North America	2
7	3	Central	United States	North America	3
8	4	Southwest	United States	North America	4
9	5	Southeast	United States	North America	5
10	6	Canada	Canada	North America	6
11	9	Australia	Australia	Pacific	1

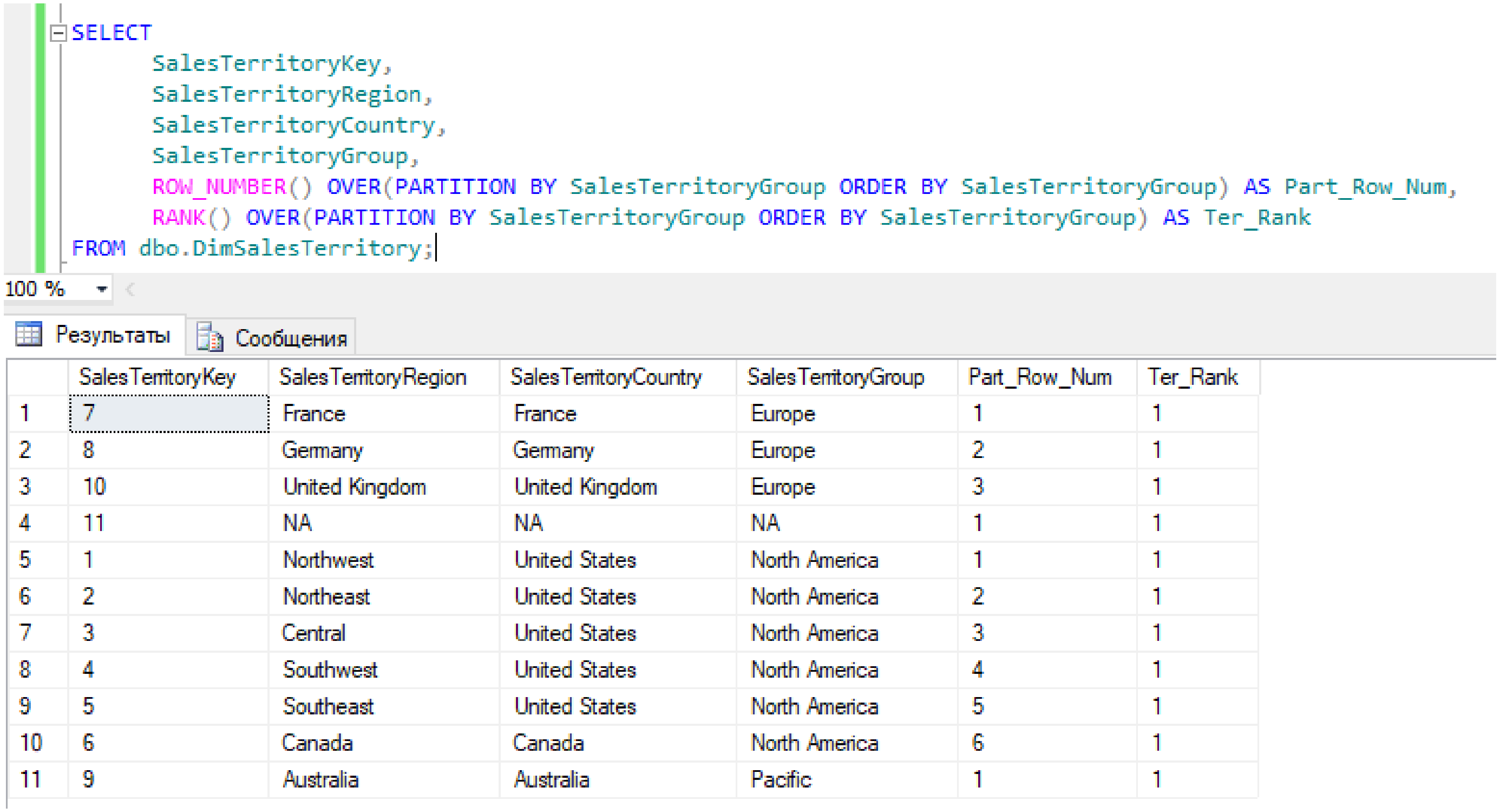
RANK()



```
--- RANK()
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        ROW_NUMBER() OVER(ORDER BY SalesTerritoryGroup) AS Row_Num,
        RANK() OVER(ORDER BY SalesTerritoryGroup) AS Ter_Rank
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Row_Num	Ter_Rank
1	7	France	France	Europe	1	1
2	8	Germany	Germany	Europe	2	1
3	10	United Kingdom	United Kingdom	Europe	3	1
4	11	NA	NA	NA	4	4
5	1	Northwest	United States	North America	5	5
6	2	Northeast	United States	North America	6	5
7	3	Central	United States	North America	7	5
8	4	Southwest	United States	North America	8	5
9	5	Southeast	United States	North America	9	5
10	6	Canada	Canada	North America	10	5
11	9	Australia	Australia	Pacific	11	11

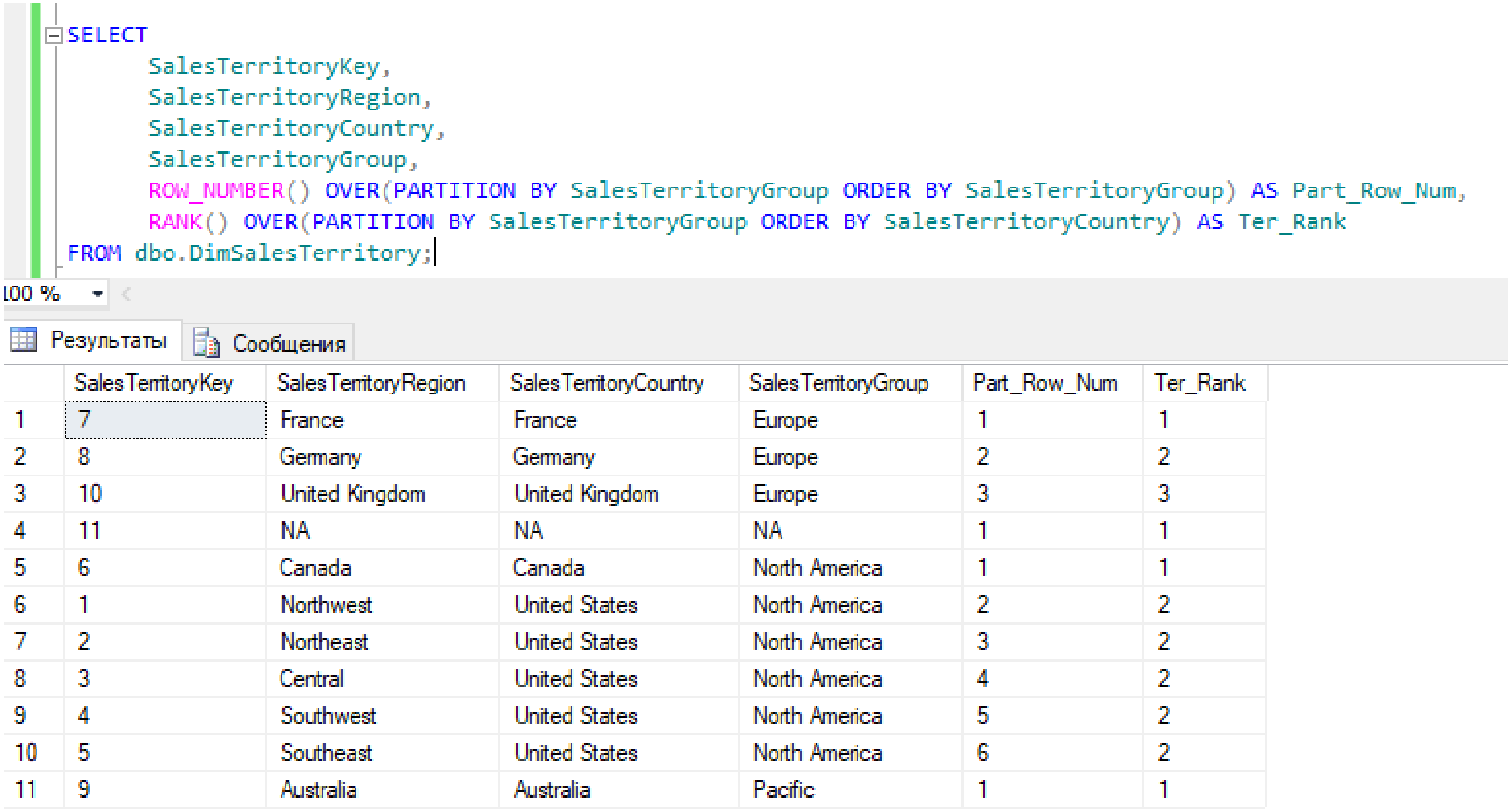
RANK()



```
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      ROW_NUMBER() OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Part_Row_Num,
      RANK() OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Ter_Rank
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Part_Row_Num	Ter_Rank
1	7	France	France	Europe	1	1
2	8	Germany	Germany	Europe	2	1
3	10	United Kingdom	United Kingdom	Europe	3	1
4	11	NA	NA	NA	1	1
5	1	Northwest	United States	North America	1	1
6	2	Northeast	United States	North America	2	1
7	3	Central	United States	North America	3	1
8	4	Southwest	United States	North America	4	1
9	5	Southeast	United States	North America	5	1
10	6	Canada	Canada	North America	6	1
11	9	Australia	Australia	Pacific	1	1

RANK()



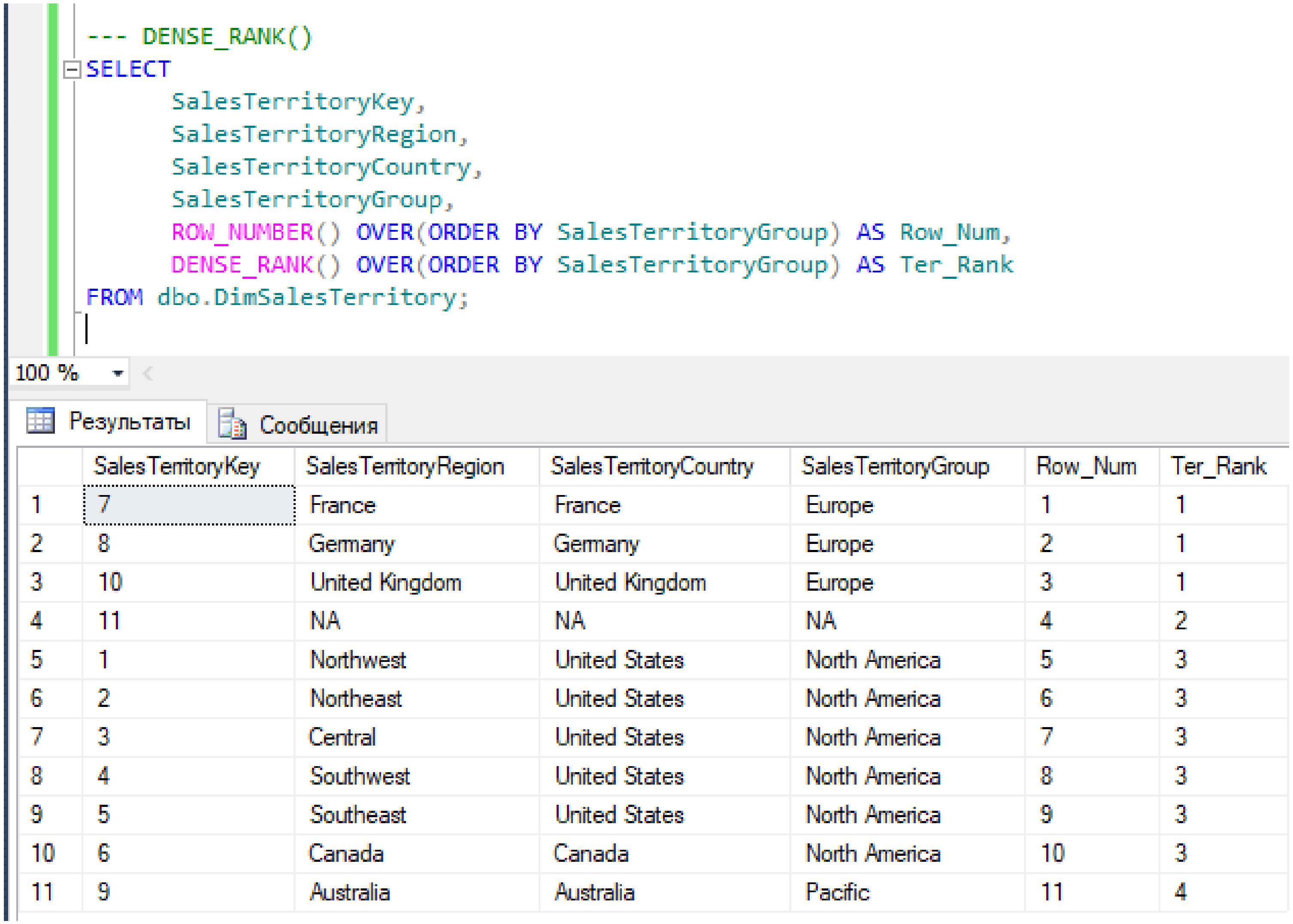
```
SELECT
        SalesTerritoryKey,
        SalesTerritoryRegion,
        SalesTerritoryCountry,
        SalesTerritoryGroup,
        ROW_NUMBER() OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Part_Row_Num,
        RANK() OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryCountry) AS Ter_Rank
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Part_Row_Num	Ter_Rank
1	7	France	France	Europe	1	1
2	8	Germany	Germany	Europe	2	2
3	10	United Kingdom	United Kingdom	Europe	3	3
4	11	NA	NA	NA	1	1
5	6	Canada	Canada	North America	1	1
6	1	Northwest	United States	North America	2	2
7	2	Northeast	United States	North America	3	2
8	3	Central	United States	North America	4	2
9	4	Southwest	United States	North America	5	2
10	5	Southeast	United States	North America	6	2
11	9	Australia	Australia	Pacific	1	1

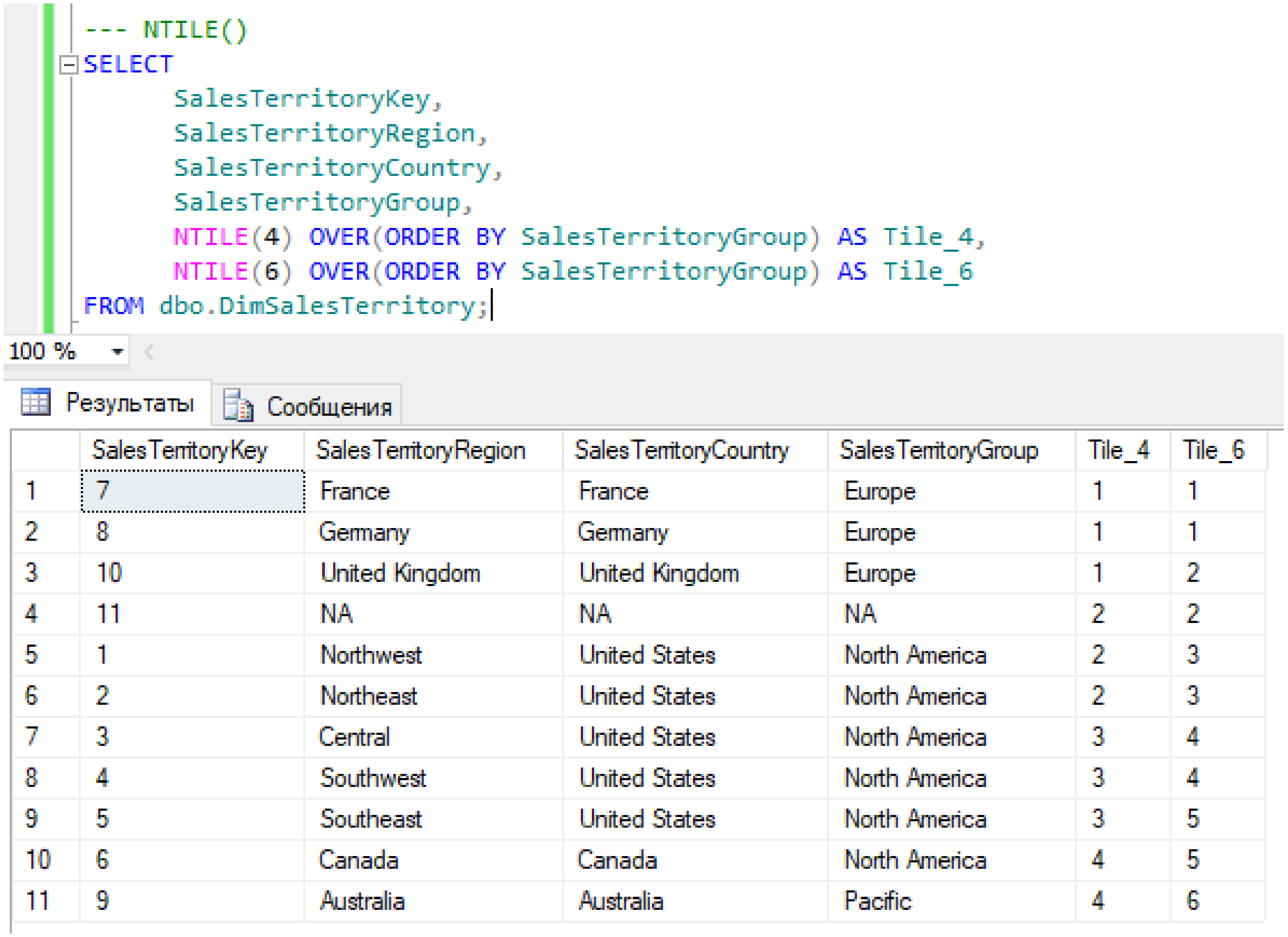
DENSE_RANK()



```sql
--- DENSE_RANK()
SELECT
       SalesTerritoryKey,
       SalesTerritoryRegion,
       SalesTerritoryCountry,
       SalesTerritoryGroup,
       ROW_NUMBER() OVER(ORDER BY SalesTerritoryGroup) AS Row_Num,
       DENSE_RANK() OVER(ORDER BY SalesTerritoryGroup) AS Ter_Rank
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Row_Num	Ter_Rank
1	7	France	France	Europe	1	1
2	8	Germany	Germany	Europe	2	1
3	10	United Kingdom	United Kingdom	Europe	3	1
4	11	NA	NA	NA	4	2
5	1	Northwest	United States	North America	5	3
6	2	Northeast	United States	North America	6	3
7	3	Central	United States	North America	7	3
8	4	Southwest	United States	North America	8	3
9	5	Southeast	United States	North America	9	3
10	6	Canada	Canada	North America	10	3
11	9	Australia	Australia	Pacific	11	4

NTILE()



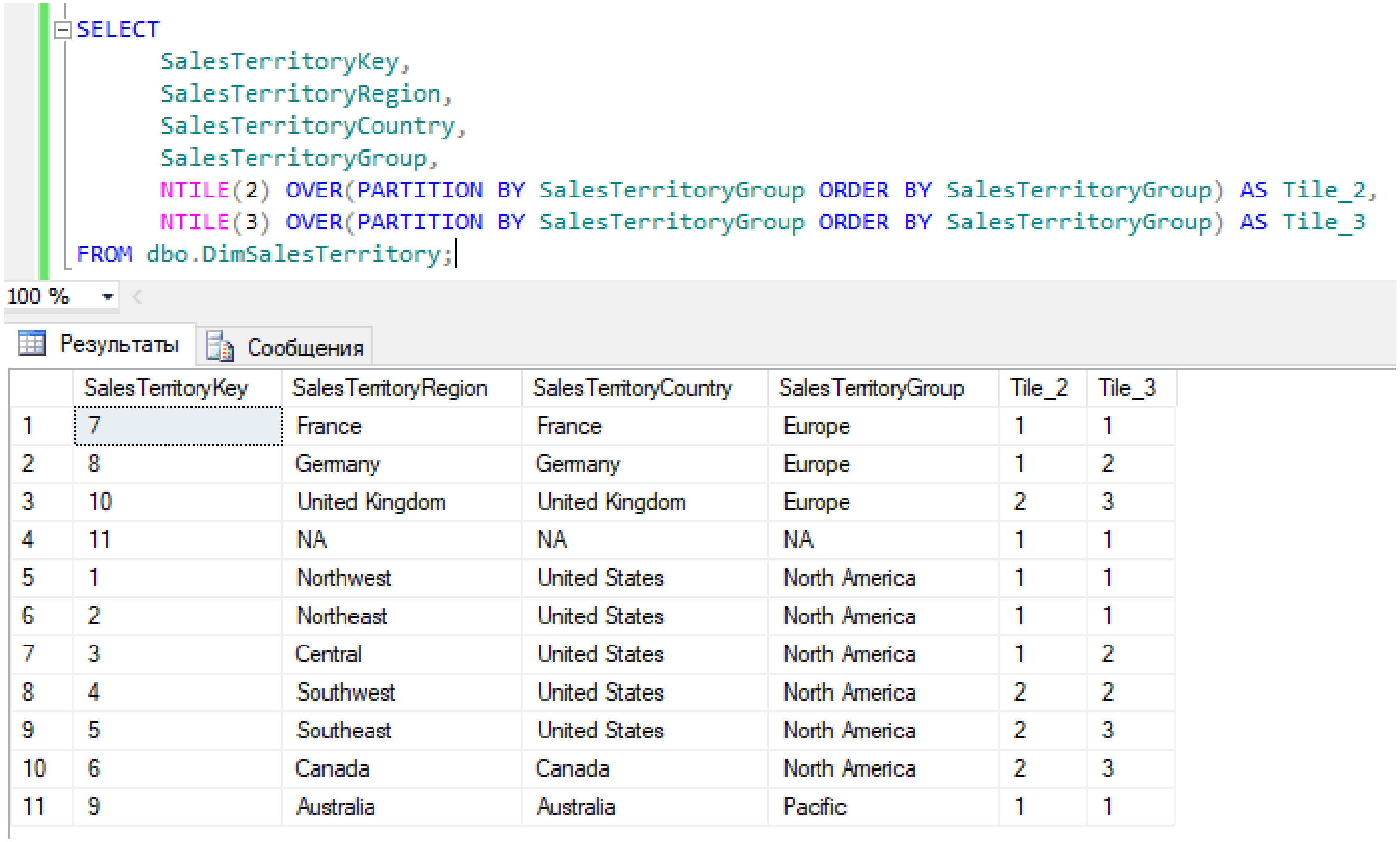
```sql
--- NTILE()
SELECT
       SalesTerritoryKey,
       SalesTerritoryRegion,
       SalesTerritoryCountry,
       SalesTerritoryGroup,
       NTILE(4) OVER(ORDER BY SalesTerritoryGroup) AS Tile_4,
       NTILE(6) OVER(ORDER BY SalesTerritoryGroup) AS Tile_6
FROM dbo.DimSalesTerritory;
```

100 %

Результаты | Сообщения

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Tile_4	Tile_6
1	7	France	France	Europe	1	1
2	8	Germany	Germany	Europe	1	1
3	10	United Kingdom	United Kingdom	Europe	1	2
4	11	NA	NA	NA	2	2
5	1	Northwest	United States	North America	2	3
6	2	Northeast	United States	North America	2	3
7	3	Central	United States	North America	3	4
8	4	Southwest	United States	North America	3	4
9	5	Southeast	United States	North America	3	5
10	6	Canada	Canada	North America	4	5
11	9	Australia	Australia	Pacific	4	6

NTILE()



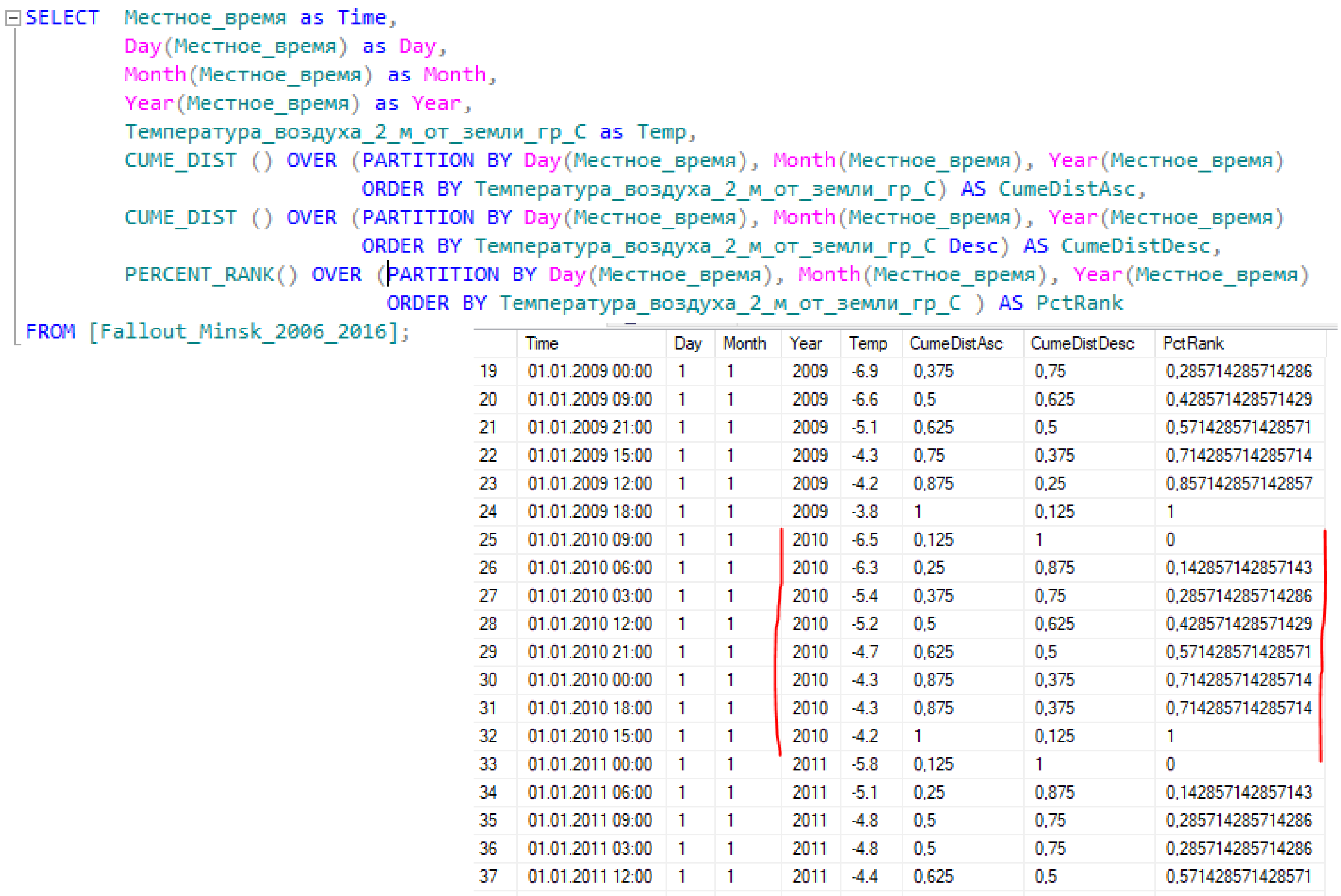
```
SELECT
      SalesTerritoryKey,
      SalesTerritoryRegion,
      SalesTerritoryCountry,
      SalesTerritoryGroup,
      NTILE(2) OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Tile_2,
      NTILE(3) OVER(PARTITION BY SalesTerritoryGroup ORDER BY SalesTerritoryGroup) AS Tile_3
FROM dbo.DimSalesTerritory;
```

	SalesTerritoryKey	SalesTerritoryRegion	SalesTerritoryCountry	SalesTerritoryGroup	Tile_2	Tile_3
1	7	France	France	Europe	1	1
2	8	Germany	Germany	Europe	1	2
3	10	United Kingdom	United Kingdom	Europe	2	3
4	11	NA	NA	NA	1	1
5	1	Northwest	United States	North America	1	1
6	2	Northeast	United States	North America	1	1
7	3	Central	United States	North America	1	2
8	4	Southwest	United States	North America	2	2
9	5	Southeast	United States	North America	2	3
10	6	Canada	Canada	North America	2	3
11	9	Australia	Australia	Pacific	1	1

Аналитические функции

- PERCENT_RANK
- CUME_DIST
- PERCENTILE_CONT
- PERCENTILE_DISC

PERCENT_RANK

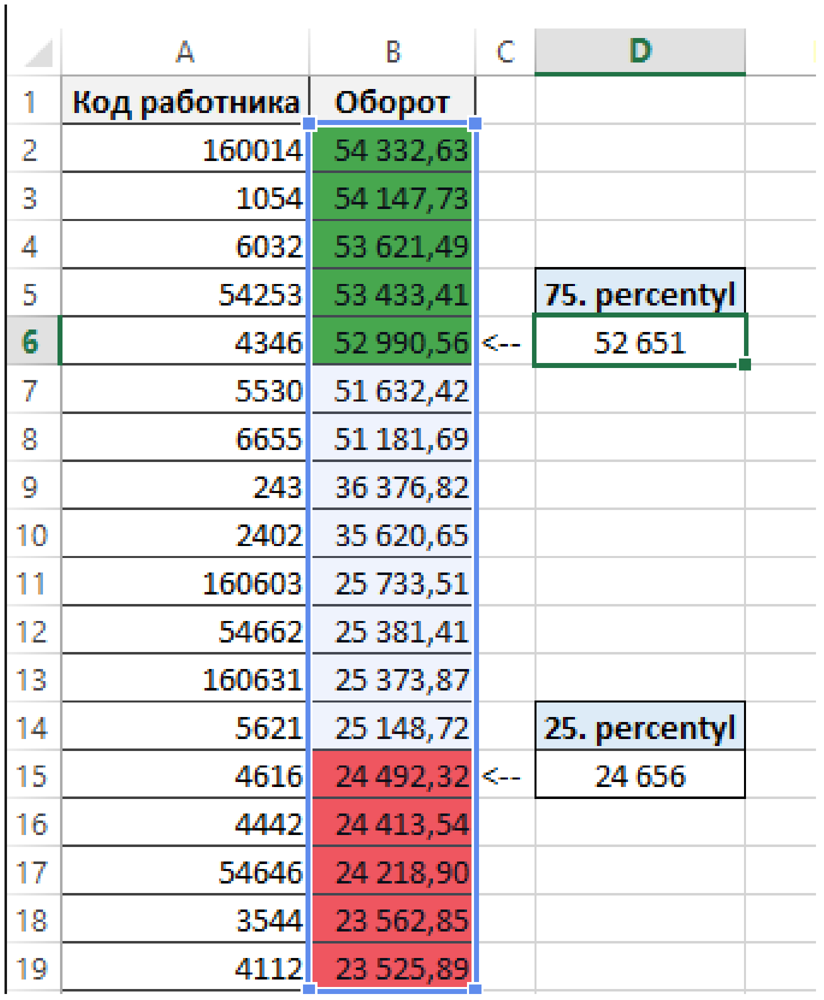


```
SELECT  Местное_время as Time,
        Day(Местное_время) as Day,
        Month(Местное_время) as Month,
        Year(Местное_время) as Year,
        Температура_воздуха_2_м_от_земли_гр_С as Temp,
        CUME_DIST () OVER (PARTITION BY Day(Местное_время), Month(Местное_время), Year(Местное_время)
                           ORDER BY Температура_воздуха_2_м_от_земли_гр_С) AS CumeDistAsc,
        CUME_DIST () OVER (PARTITION BY Day(Местное_время), Month(Местное_время), Year(Местное_время)
                           ORDER BY Температура_воздуха_2_м_от_земли_гр_С Desc) AS CumeDistDesc,
        PERCENT_RANK() OVER (PARTITION BY Day(Местное_время), Month(Местное_время), Year(Местное_время)
                            ORDER BY Температура_воздуха_2_м_от_земли_гр_С ) AS PctRank
FROM [Fallout_Minsk_2006_2016];
```

	Time	Day	Month	Year	Temp	CumeDistAsc	CumeDistDesc	PctRank
19	01.01.2009 00:00	1	1	2009	-6.9	0,375	0,75	0,285714285714286
20	01.01.2009 09:00	1	1	2009	-6.6	0,5	0,625	0,428571428571429
21	01.01.2009 21:00	1	1	2009	-5.1	0,625	0,5	0,571428571428571
22	01.01.2009 15:00	1	1	2009	-4.3	0,75	0,375	0,714285714285714
23	01.01.2009 12:00	1	1	2009	-4.2	0,875	0,25	0,857142857142857
24	01.01.2009 18:00	1	1	2009	-3.8	1	0,125	1
25	01.01.2010 09:00	1	1	2010	-6.5	0,125	1	0
26	01.01.2010 06:00	1	1	2010	-6.3	0,25	0,875	0,142857142857143
27	01.01.2010 03:00	1	1	2010	-5.4	0,375	0,75	0,285714285714286
28	01.01.2010 12:00	1	1	2010	-5.2	0,5	0,625	0,428571428571429
29	01.01.2010 21:00	1	1	2010	-4.7	0,625	0,5	0,571428571428571
30	01.01.2010 00:00	1	1	2010	-4.3	0,875	0,375	0,714285714285714
31	01.01.2010 18:00	1	1	2010	-4.3	0,875	0,375	0,714285714285714
32	01.01.2010 15:00	1	1	2010	-4.2	1	0,125	1
33	01.01.2011 00:00	1	1	2011	-5.8	0,125	1	0
34	01.01.2011 06:00	1	1	2011	-5.1	0,25	0,875	0,142857142857143
35	01.01.2011 09:00	1	1	2011	-4.8	0,5	0,75	0,285714285714286
36	01.01.2011 03:00	1	1	2011	-4.8	0,5	0,75	0,285714285714286
37	01.01.2011 12:00	1	1	2011	-4.4	0,625	0,5	0,571428571428571

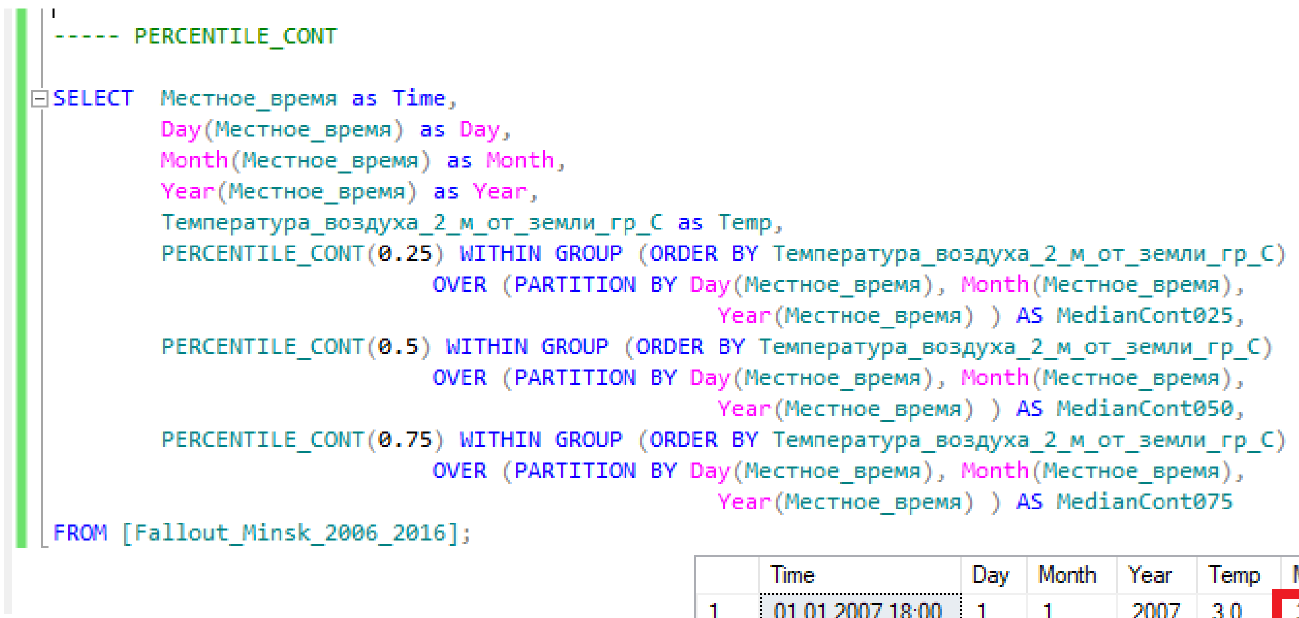
Процентиль

- Процентиль — мера, в которой процентное значение общих значений равно этой мере или меньше ее
- 90 % значений данных находятся ниже 90-го процентиля
- 10 % значений данных находятся ниже 10-го процентиля

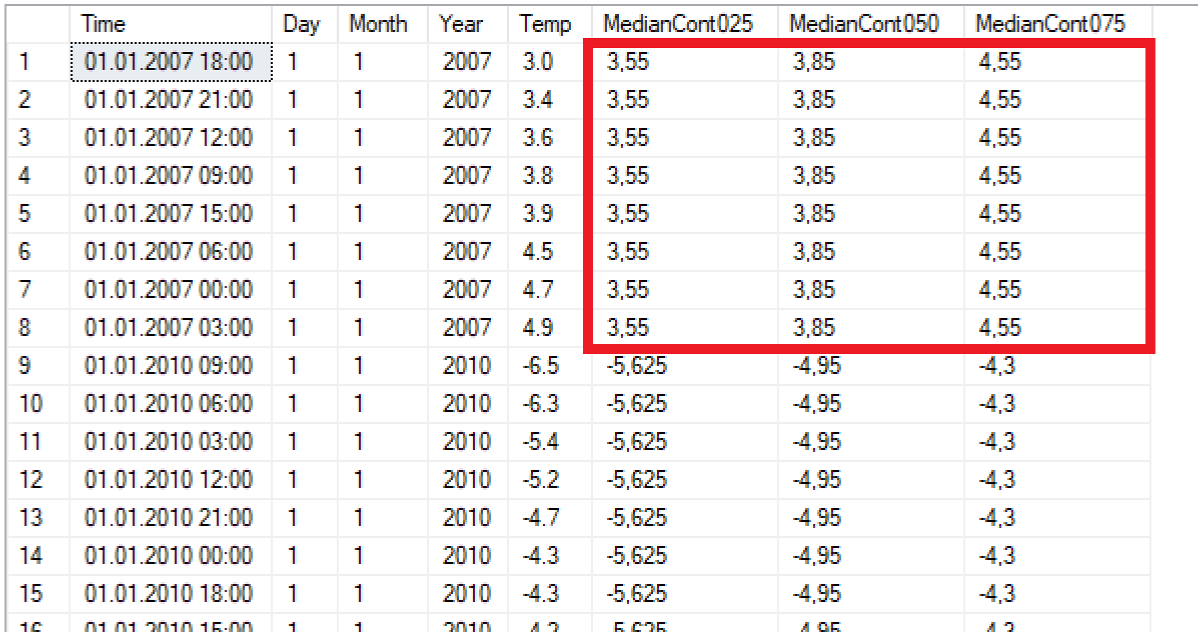


	A	B	C	D
1	Код работника	Оборот		
2	160014	54 332,63		
3	1054	54 147,73		
4	6032	53 621,49		
5	54253	53 433,41		75. percentyl
6	4346	52 990,56	<--	52 651
7	5530	51 632,42		
8	6655	51 181,69		
9	243	36 376,82		
10	2402	35 620,65		
11	160603	25 733,51		
12	54662	25 381,41		
13	160631	25 373,87		
14	5621	25 148,72		25. percentyl
15	4616	24 492,32	<--	24 656
16	4442	24 413,54		
17	54646	24 218,90		
18	3544	23 562,85		
19	4112	23 525,89		

PERCENTILE_CONT



```
----- PERCENTILE_CONT

SELECT  Местное_время as Time,
        Day(Местное_время) as Day,
        Month(Местное_время) as Month,
        Year(Местное_время) as Year,
        Температура_воздуха_2_м_от_земли_гр_С as Temp,
        PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY Температура_воздуха_2_м_от_земли_гр_С)
                            OVER (PARTITION BY Day(Местное_время), Month(Местное_время),
                                                     Year(Местное_время) ) AS MedianCont025,
        PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY Температура_воздуха_2_м_от_земли_гр_С)
                            OVER (PARTITION BY Day(Местное_время), Month(Местное_время),
                                                     Year(Местное_время) ) AS MedianCont050,
        PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY Температура_воздуха_2_м_от_земли_гр_С)
                            OVER (PARTITION BY Day(Местное_время), Month(Местное_время),
                                                     Year(Местное_время) ) AS MedianCont075
FROM [Fallout_Minsk_2006_2016];
```



	Time	Day	Month	Year	Temp	MedianCont025	MedianCont050	MedianCont075
1	01.01.2007 18:00	1	1	2007	3.0	3,55	3,85	4,55
2	01.01.2007 21:00	1	1	2007	3.4	3,55	3,85	4,55
3	01.01.2007 12:00	1	1	2007	3.6	3,55	3,85	4,55
4	01.01.2007 09:00	1	1	2007	3.8	3,55	3,85	4,55
5	01.01.2007 15:00	1	1	2007	3.9	3,55	3,85	4,55
6	01.01.2007 06:00	1	1	2007	4.5	3,55	3,85	4,55
7	01.01.2007 00:00	1	1	2007	4.7	3,55	3,85	4,55
8	01.01.2007 03:00	1	1	2007	4.9	3,55	3,85	4,55
9	01.01.2010 09:00	1	1	2010	-6.5	-5,625	-4,95	-4,3
10	01.01.2010 06:00	1	1	2010	-6.3	-5,625	-4,95	-4,3
11	01.01.2010 03:00	1	1	2010	-5.4	-5,625	-4,95	-4,3
12	01.01.2010 12:00	1	1	2010	-5.2	-5,625	-4,95	-4,3
13	01.01.2010 21:00	1	1	2010	-4.7	-5,625	-4,95	-4,3
14	01.01.2010 00:00	1	1	2010	-4.3	-5,625	-4,95	-4,3
15	01.01.2010 18:00	1	1	2010	-4.3	-5,625	-4,95	-4,3
16	01.01.2010 15:00	1	1	2010	-4.2	-5,625	-4,95	-4,3

Вопросы?