```sql
--Запрос 1
select job_industry_category, count(job_industry_category) as cnt_job_industry_category
from customer_new
group by job_industry_category
order by cnt_job_industry_category desc;
```

customer_new 1

select job_industry_category, count(job_industry_category) as cnt_j | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	job_industry_category	cnt_job_industry_category
1	Manufacturing	799
2	Financial Services	774
3	n/a	656
4	Health	602
5	Retail	358
6	Property	267
7	IT	223
8	Entertainment	136
9	Argiculture	113
10	Telecommunications	72

*<postgres> Голубев_Роман_Script_3.sql

```sql
--Запрос 2
select date_trunc('month', ord.order_date) as order_month,
    job_industry_category,
    sum(pd.list_price * oi.quantity) as total_income
from customer_new cs
join (select * from orders where order_status = 'Approved') ord on cs.customer_id = ord.customer_id
join order_items oi on ord.order_id = oi.order_id
join product_cor pd on oi.product_id = pd.product_id
group by order_month, job_industry_category
order by order_month, job_industry_category;
```

customer_new 1

select date_trunc('month', ord.order_date) as order_month, job_industry_ | Введите SQL выражение чтобы отфильтровать результаты

	order_month	job_industry_category	total_income
1	2017-01-01 00:00:00.000 +0300	Argiculture	277 188
2	2017-01-01 00:00:00.000 +0300	Entertainment	412 510,06
3	2017-01-01 00:00:00.000 +0300	Financial Services	2 300 634,2
4	2017-01-01 00:00:00.000 +0300	Health	1 805 116,6
5	2017-01-01 00:00:00.000 +0300	IT	689 444,6
6	2017-01-01 00:00:00.000 +0300	Manufacturing	2 320 301,2
7	2017-01-01 00:00:00.000 +0300	n/a	2 114 676,8
8	2017-01-01 00:00:00.000 +0300	Property	557 339,3
9	2017-01-01 00:00:00.000 +0300	Retail	1 174 977,1
10	2017-01-01 00:00:00.000 +0300	Telecommunications	204 494,22
11	2017-02-01 00:00:00.000 +0300	Argiculture	400 891,7
12	2017-02-01 00:00:00.000 +0300	Entertainment	379 554,62
13	2017-02-01 00:00:00.000 +0300	Financial Services	2 384 491,5
14	2017-02-01 00:00:00.000 +0300	Health	1 692 922,8
15	2017-02-01 00:00:00.000 +0300	IT	675 947,44
16	2017-02-01 00:00:00.000 +0300	Manufacturing	2 707 108
17	2017-02-01 00:00:00.000 +0300	n/a	1 679 699
18	2017-02-01 00:00:00.000 +0300	Property	794 729,4
19	2017-02-01 00:00:00.000 +0300	Retail	1 017 290,8
20	2017-02-01 00:00:00.000 +0300	Telecommunications	214 820,25
21	2017-03-01 00:00:00.000 +0300	Argiculture	294 705
22	2017-03-01 00:00:00.000 +0300	Entertainment	464 368,38
23	2017-03-01 00:00:00.000 +0300	Financial Services	2 193 274
24	2017-03-01 00:00:00.000 +0300	Health	1 815 382,4
25	2017-03-01 00:00:00.000 +0300	IT	595 833,75
26	2017-03-01 00:00:00.000 +0300	Manufacturing	2 601 364,8
27	2017-03-01 00:00:00.000 +0300	n/a	1 848 568,1
28	2017-03-01 00:00:00.000 +0300	Property	764 891,7
29	2017-03-01 00:00:00.000 +0300	Retail	1 113 549
30	2017-03-01 00:00:00.000 +0300	Telecommunications	185 370,9
31	2017-04-01 00:00:00.000 +0300	Argiculture	314 693,03
32	2017-04-01 00:00:00.000 +0300	Entertainment	426 454,38

Обновить | Save | Cancel | Экспорт данных ... | 200 | 120 | 120 строк получено - 0.0s (0.0s получ.), 2025-11-27 в 17:39:17

```sql
--Запрос 3
select pd.brand,
    sum(case when cs.job_industry_category = 'IT' and ord.online_order = 'True' then 1 else 0 end) as cnt_IT_online_orders
from customer_new cs
join (select distinct order_id, customer_id, online_order from orders where order_status = 'Approved') ord on cs.customer_id = ord.customer_id
join order_items oi on ord.order_id = oi.order_id
join product_cor pd on oi.product_id = pd.product_id
group by pd.brand;
```

product_cor 1

select pd.brand, sum(case when cs.job_industry_category = 'IT' and ord.c | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	brand	cnt_it_online_orders
1	OHM Cycles	69
2	Trek Bicycles	78
3	WeareA2B	87
4	Solex	101
5	Norco Bicycles	59
6	Giant Bicycles	102

*<postgres> Голубев_Роман_Script_3.sql

```sql
--Запрос 4
--используя только group by
select cs.customer_id,
    sum(pd.list_price * oi.quantity) as total_income,
    max(pd.list_price * oi.quantity) as max_order_price,
    min(pd.list_price * oi.quantity) as min_order_price,
    count(ord.order_id) as cnt_orders,
    avg(pd.list_price * oi.quantity) as avg_order_price
from customer_new cs
join orders ord on cs.customer_id = ord.customer_id
join order_items oi on ord.order_id = oi.order_id
join product_cor pd on oi.product_id = pd.product_id
group by cs.customer_id
order by total_income desc, cnt_orders desc;
--используя только оконные функции
select subquery.customer_id, subquery.total_income, subquery.max_order_price,
    subquery.min_order_price, subquery.cnt_orders, subquery.avg_order_price
from (
    select cs.customer_id,
        sum(pd.list_price * oi.quantity) over(w) as total_income,
        max(pd.list_price * oi.quantity) over(w) as max_order_price,
        min(pd.list_price * oi.quantity) over(w) as min_order_price,
        count(ord.order_id) over(w) as cnt_orders,
        avg(pd.list_price * oi.quantity) over(w) as avg_order_price,
        row_number() over(w) as rn
    from customer_new cs
    join orders ord on cs.customer_id = ord.customer_id
    join order_items oi on ord.order_id = oi.order_id
    join product_cor pd on oi.product_id = pd.product_id
    window w as (partition by cs.customer_id)
) as subquery
where subquery.rn = 1
order by subquery.total_income desc, subquery.cnt_orders desc;
--результаты для каждого customer_id в обоих вариантах получаются одинаковыми,
--скрипт запроса с оконной функцией получился сложнее,
--потому что понадобилось писать вложенный select-запрос
```

customer_new 1

select cs.customer_id, sum(pd.list_price * oi.quantity) as total_income, m | Введите SQL выражение чтобы отфильтровать результаты

	customer_id	total_income	max_order_price	min_order_price	cnt_orders	avg_order_price
1	1 129	157 178,6	20 914,7	4 757,4004	13	12 090,6607196514
2	2 183	151 898,42	20 056,6	1 469,44	14	10 849,8878348214
3	1 597	136 846,48	20 914,7	1 720,7	11	12 440,589954723
4	941	134 413,31	20 914,7	2 807	10	13 441,3318359375
5	2 309	127 089,78	18 739,7	2 073,18	12	10 590,8150227865
6	2 914	121 689,234	19 929,3	1 469,44	12	10 140,7693583171
7	1 329	118 900,36	18 739,7	1 065,03	11	10 809,1234796697
8	2 637	118 364,97	20 914,7	1 793,43	11	10 760,4517378374
9	3 240	117 282,47	18 050,94	1 285,4	10	11 728,2480834961
10	2 046	114 531,67	18 823,23	5 307,8003	9	12 725,7414822049
11	2 753	113 960,34	20 839,398	1 928,1001	11	10 360,031693892
12	1 517	112 454,39	16 865,73	1 635,3	11	10 223,1274192116

Обновить | Save | Cancel | Экспорт данных ... | 200 | 200+ | 200 строк получено - 0.0s (0.0s получ.), 2025-11-27 в 17:51:41

GMT+03:00 | ru | Запись | Инт. вставка | 98 : 45 : 2725

*<postgres> Голубев_Роман_Script_3.sql

```sql
--Запрос 4
--используя только group by
select cs.customer_id,
    sum(pd.list_price * oi.quantity) as total_income,
    max(pd.list_price * oi.quantity) as max_order_price,
    min(pd.list_price * oi.quantity) as min_order_price,
    count(ord.order_id) as cnt_orders,
    avg(pd.list_price * oi.quantity) as avg_order_price
from customer_new cs
join orders ord on cs.customer_id = ord.customer_id
join order_items oi on ord.order_id = oi.order_id
join product_cor pd on oi.product_id = pd.product_id
group by cs.customer_id
order by total_income desc, cnt_orders desc;
--используя только оконные функции
select subquery.customer_id, subquery.total_income, subquery.max_order_price,
    subquery.min_order_price, subquery.cnt_orders, subquery.avg_order_price
from (
    select cs.customer_id,
        sum(pd.list_price * oi.quantity) over(w) as total_income,
        max(pd.list_price * oi.quantity) over(w) as max_order_price,
        min(pd.list_price * oi.quantity) over(w) as min_order_price,
        count(ord.order_id) over(w) as cnt_orders,
        avg(pd.list_price * oi.quantity) over(w) as avg_order_price,
        row_number() over(w) as rn
    from customer_new cs
    join orders ord on cs.customer_id = ord.customer_id
    join order_items oi on ord.order_id = oi.order_id
    join product_cor pd on oi.product_id = pd.product_id
    window w as (partition by cs.customer_id)
) as subquery
where subquery.rn = 1
order by subquery.total_income desc, subquery.cnt_orders desc;
--результаты для каждого customer_id в обоих вариантах получаются одинаковыми,
--скрипт запроса с оконной функцией получился сложнее,
--потому что понадобилось писать вложенный select-запрос
```

customer_new 1

select subquery.customer_id, subquery.total_income, subquery.max_ord… | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст | Запись

	customer_id	total_income	max_order_price	min_order_price	cnt_orders	avg_order_price
1	1 129	157 178,58	20 914,7	4 757,4004	13	12 090,6607196514
2	2 183	151 898,44	20 056,6	1 469,44	14	10 849,8878348214
3	1 597	136 846,48	20 914,7	1 720,7	11	12 440,589954723
4	941	134 413,31	20 914,7	2 807	10	13 441,3318359375
5	2 309	127 089,79	18 739,7	2 073,18	12	10 590,8150227865
6	2 914	121 689,234	19 929,3	1 469,44	12	10 140,7693583171
7	1 329	118 900,36	18 739,7	1 065,03	11	10 809,1234796697
8	2 637	118 364,97	20 914,7	1 793,43	11	10 760,4517378374
9	3 240	117 282,48	18 050,94	1 285,4	10	11 728,2480834961
10	2 046	114 531,67	18 823,23	5 307,8003	9	12 725,7414822049
11	2 753	113 960,35	20 839,398	1 928,1001	11	10 360,031693892
12	1 517	112 454,41	16 865,73	1 635,3	11	10 223,1274192116

Обновить | Save | Cancel | Экспорт данных … | 200 | 200+ | 200 строк получено - 0.0s, 2025-11-27 в 17:52:23

GMT+03:00 | ru | Запись | Инт. вставка | 117 : 63 : 3609

```sql
--Запрос 5
select *
from (
    select cs.customer_id, cs.first_name, cs.last_name,
        sum(case when ord.order_id is not null then pd.list_price * oi.quantity else 0 end) as total_income
    from customer_new cs
    left join orders ord on cs.customer_id = ord.customer_id
    left join order_items oi on ord.order_id = oi.order_id
    left join product_cor pd on oi.product_id = pd.product_id
    group by cs.customer_id
    order by total_income asc
    limit 3
)
union all
select *
from (
    select cs.customer_id, first_name, last_name,
        sum(pd.list_price * oi.quantity) as total_income
    from customer_new cs
    join orders ord on cs.customer_id = ord.customer_id
    join order_items oi on ord.order_id = oi.order_id
    join product_cor pd on oi.product_id = pd.product_id
    group by cs.customer_id
    order by total_income desc
    limit 3
);
```

Результат 1

select * from (select cs.customer_id, cs.first_name, cs.last_name, sum(ca | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	customer_id	first_name	last_name	total_income
1	3 725	Elisha	Venny	0
2	3 936	Rodd	Spare	0
3	1 373	Shaylynn	Epsley	0
4	1 129	Hercule		157 178,6
5	2 183	Jillie	Fyndon	151 898,42
6	1 597	Jeffry	Slowly	136 846,48

<postgres> Голубев_Роман_Script_3.sql

```sql
--Запрос 6
select subquery.order_id, subquery.customer_id, subquery.order_date,
    subquery.online_order, subquery.order_status
from (
    select *,
        row_number() over(partition by customer_id order by order_date) as rn
    from orders
) as subquery
where subquery.rn = 2;
```

orders 1

select subquery.order_id, subquery.customer_id, subquery.order_date, su | Введите SQL выражение чтобы отфильтровать результаты

	order_id	customer_id	order_date	online_order	order_status
1	13 424	1	2017-02-21	False	Approved
2	6 743	2	2017-06-11	False	Approved
3	15 188	3	2017-03-24	False	Approved
4	14 648	4	2017-06-18	True	Approved
5	19 993	5	2017-04-28	False	Approved
6	8 204	6	2017-02-06	True	Approved
7	18 549	7	2017-02-24	True	Approved
8	8 415	8	2017-04-11	False	Approved
9	2 979	9	2017-03-06	False	Approved
10	10 250	10	2017-07-13	True	Approved
11	14 370	11	2017-07-09	False	Approved
12	12 242	12	2017-07-23	False	Approved
13	8 905	13	2017-02-16	False	Approved
14	8 486	14	2017-08-16	True	Approved
15	434	15	2017-03-10	False	Approved
16	5 083	16	2017-05-10	False	Approved
17	10 775	17	2017-05-01	False	Approved
18	3 777	18	2017-05-11	False	Approved
19	14 850	19	2017-03-25	False	Approved
20	6 657	20	2017-11-01	True	Approved
21	2 932	21	2017-08-17	True	Approved
22	16 626	22	2017-03-10	False	Approved
23	7 695	23	2017-04-02	True	Approved
24	7 610	24	2017-05-11	False	Approved
25	2 354	25	2017-03-04	True	Approved
26	3 749	26	2017-04-15	True	Approved
27	5 662	27	2017-05-28	True	Approved
28	8 139	28	2017-01-31	True	Approved
29	19 532	29	2017-01-25	False	Approved
30	1 739	30	2017-06-16	False	Approved
31	420	31	2017-05-25	True	Approved
32	99	32	2017-03-26	False	Approved
33	3 731	33	2017-05-30	False	Approved

Обновить | Save | Cancel | Экспорт данных ... | 200 | 200+ | 200 строк получено - 0.0s (0.0s получ.), 2025-11-27 в 18:01:11

GMT+03:00 | ru | Запись | Инт. вставка | 159 : 23 : 4926

*<postgres> Голубев_Роман_Script_3.sql

```sql
--Запрос 7
select subquery.first_name, subquery.last_name, subquery.job_title,
    max(subquery.days_between_orders) max_days_between_orders
from (
    select first_name, last_name, job_title,
        order_date - lag(order_date) over(partition by cs.customer_id order by order_date) days_between_orders,
        row_number() over(partition by cs.customer_id) as rn
    from customer_new cs
    join orders ord on cs.customer_id = ord.customer_id
) as subquery
where subquery.rn >= 2
group by subquery.first_name, subquery.last_name, subquery.job_title;
```

customer_new 1

select subquery.first_name, subquery.last_name, subquery.job_title, max(Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	first_name	last_name	job_title	max_days_between_orders
1	Krissy	Robard	VP Product Management	109
2	Fabien	Iacapucci	Community Outreach Specialist	58
3	Teressa	Tague	Marketing Assistant	157
4	Ira	Lamlin	Financial Analyst	133
5	Audry	Scurrey	Junior Executive	78
6	Rahal	Woodman	Civil Engineer	156
7	Nerita	Keppie	Product Engineer	206
8	Benedicto	Blumire		125
9	Tobit			78
10	Cathyleen	Bern	Director of Sales	64
11	Caro	McKirton	Administrative Officer	174
12	Reginald	Hanwright	Professor	168
13	Ranee	Henriksson		174
14	Berthe	Ludwell	Account Representative II	190
15	Nevsa	Washtell	Biostatistician III	190
16	Basile	Bowlas	Registered Nurse	234
17	Augusta	Henryson	Web Designer IV	162
18	Clarita	Penright	Business Systems Development Analyst	64
19	Crystal	Assur	Clinical Specialist	151
20	Madel	Palffrey	Systems Administrator I	144
21	Anselm	Gawne	Account Executive	144
22	Rutter	Marlen	Sales Associate	3
23	Fae	Done	Structural Engineer	133
24	Tiffi	Wortt	Database Administrator III	127
25	Duke	Allnatt	Nurse	90
26	Lorinda	Malpass	Nurse Practicioner	31
27	Thomasine	McCloch	GIS Technical Architect	186
28	Matelda	Kordas	Product Engineer	116
29	Jerome	Muggeridge	GIS Technical Architect	150
30	Jason	De Lorenzo	Operator	91
31	Danella	Chevers		119

Запись

Обновить | Save | Cancel | Экспорт данных ... | 200 | 200+ | 200 строк получено - 0.0s, 2025-11-27 в 17:22:37

```sql
--Запрос 8
select subquery.first_name, subquery.last_name, subquery.wealth_segment, subquery.total_income
from (
    select cs.first_name, cs.last_name, cs.wealth_segment,
        sum(pd.list_price * oi.quantity) as total_income,
        row_number() over(partition by cs.wealth_segment order by sum(pd.list_price * oi.quantity) desc) as rn
    from customer_new cs
    join orders ord on cs.customer_id = ord.customer_id
    join order_items oi on ord.order_id = oi.order_id
    join product_cor pd on oi.product_id = pd.product_id
    group by cs.customer_id
) as subquery
where subquery.rn <= 5;
```

customer_new 1

select subquery.first_name, subquery.last_name, subquery.wealth_segme | Введите SQL выражение чтобы отфильтровать результаты

	first_name	last_name	wealth_segment	total_income
1	Jeffry	Slowly	Affluent Customer	136 846,48
2	Tye	Doohan	Affluent Customer	134 413,31
3	Herc	McIlhone	Affluent Customer	127 089,78
4	Jessamine	Brazear	Affluent Customer	121 689,234
5	Murdoch	Twort	Affluent Customer	112 454,39
6	Mercy	Wilsone	High Net Worth	103 528,72
7	Lockwood	Exroll	High Net Worth	85 197,29
8	Andee	Ormrod	High Net Worth	84 376,13
9	Cleveland	Coxon	High Net Worth	83 003,12
10	Morley	Shutt	High Net Worth	81 070,51
11	Hercule		Mass Customer	157 178,6
12	Jillie	Fyndon	Mass Customer	151 898,42
13	Wendy	Randlesome	Mass Customer	118 900,36
14	Marcile	Christley	Mass Customer	118 364,97
15	Ryon	Elsay	Mass Customer	117 282,47

Таблица Текст Запись

Обновить Save Cancel Экспорт данных ... 200 15 15 строк получено - 0.0s, 2025-11-27 в 17:23:36