Ртищева Алена Сергеевна
Группа М08-501НД
Науки о данных

Домашнее задание 3. Группировка данных и оконные функции

Даны CSV-файлы:
customer.csv
product.csv
orders.csv
order_items.csv

Предварительный экспресс анализ данных выполненный с помощью pandas, показал, что все таблицы идентичны таблицам из предыдущего домашнего задания, поэтому создание новых таблиц не требуется. Результаты сохранены в файле pandas_analysis_3.ipynb.

Оставим описание данных таблиц для наглядности.

Таблица **customer**:

Поле	Описание	Тип данных
customer_id	ID клиента	integer [primary key]
first_name	Имя клиента	varchar(50)
last_name	Фамилия клиента	varchar(50)
gender	Пол	varchar(10)
DOB	Дата рождения	date
job_title	Профессия	varchar(100)
job_industry_category	Сфера деятельности	varchar(50)
wealth_segment	Сегмент благосостояния	varchar(20)
deceased_indicator	Индикатор актуального клиента	varchar(1)
owns_car	Индикатор наличия автомобиля	varchar(3)
address	Адрес проживания	varchar(200)
postcode	Почтовый индекс	integer
state	Штаты	varchar(20)
country	Страна проживания	varchar(20)
property_valuation	Оценка имущества	integer

Таблица **product**:

Поле	Описание	Тип данных
product_id	ID продукта	integer
brand	Бренд	varchar(50)
product_line	Линейка продуктов	varchar(20)
product_class	Класс продукта	varchar(10)
product_size	Размер продукта	varchar(10)
list_price	Цена	decimal(10,2)
standard_cost	Стандартная стоимость	decimal(10,2)

Таблица **orders**:

Поле	Описание	Тип данных
order_id	ID транзакции	integer [primary key]
customer_id	ID клиента	integer
order_date	Дата транзакции	date
online_order	Индикатор онлайн-заказа	boolean
order_status	Статус транзакции	varchar(10)

Таблица **order_items**:

Поле	Описание	Тип данных
order_item_id	ID позиции в заказе	integer [primary key]
order_id	ID заказа	integer
product_id	ID продукта	integer
quantity	Количество данного продукта в заказе	decimal(10,2)
item_list_price_at_sale	Цена продукта в момент продажи	decimal(10,2)
item_standard_cost_at_sale	Стандартная стоимость продукта в момент продажи	decimal(10,2)

Выполнение запросов

1. Вывести распределение (количество) клиентов по сферам деятельности, отсортировав результат по убыванию количества.

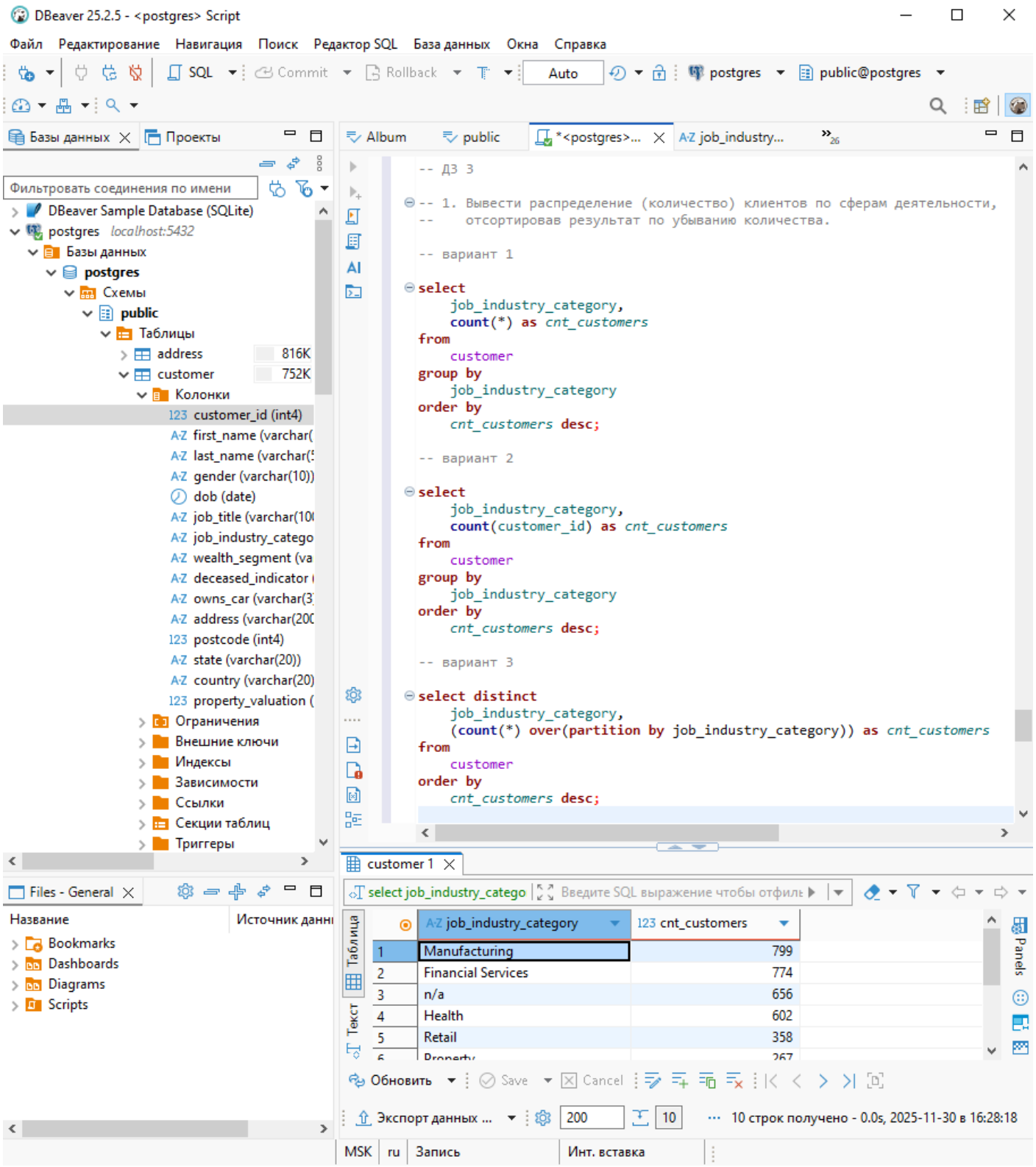

Рисунок 1 – Реализация запроса № 1

2. Найти общую сумму дохода (list_price*quantity) по всем подтвержденным заказам за каждый месяц по сферам деятельности клиентов. Отсортировать результат по году, месяцу и сфере деятельности.

Следует отметить что list_price в таблице **product** совпадает с item_list_price_at_sale в таблице **order_items**.

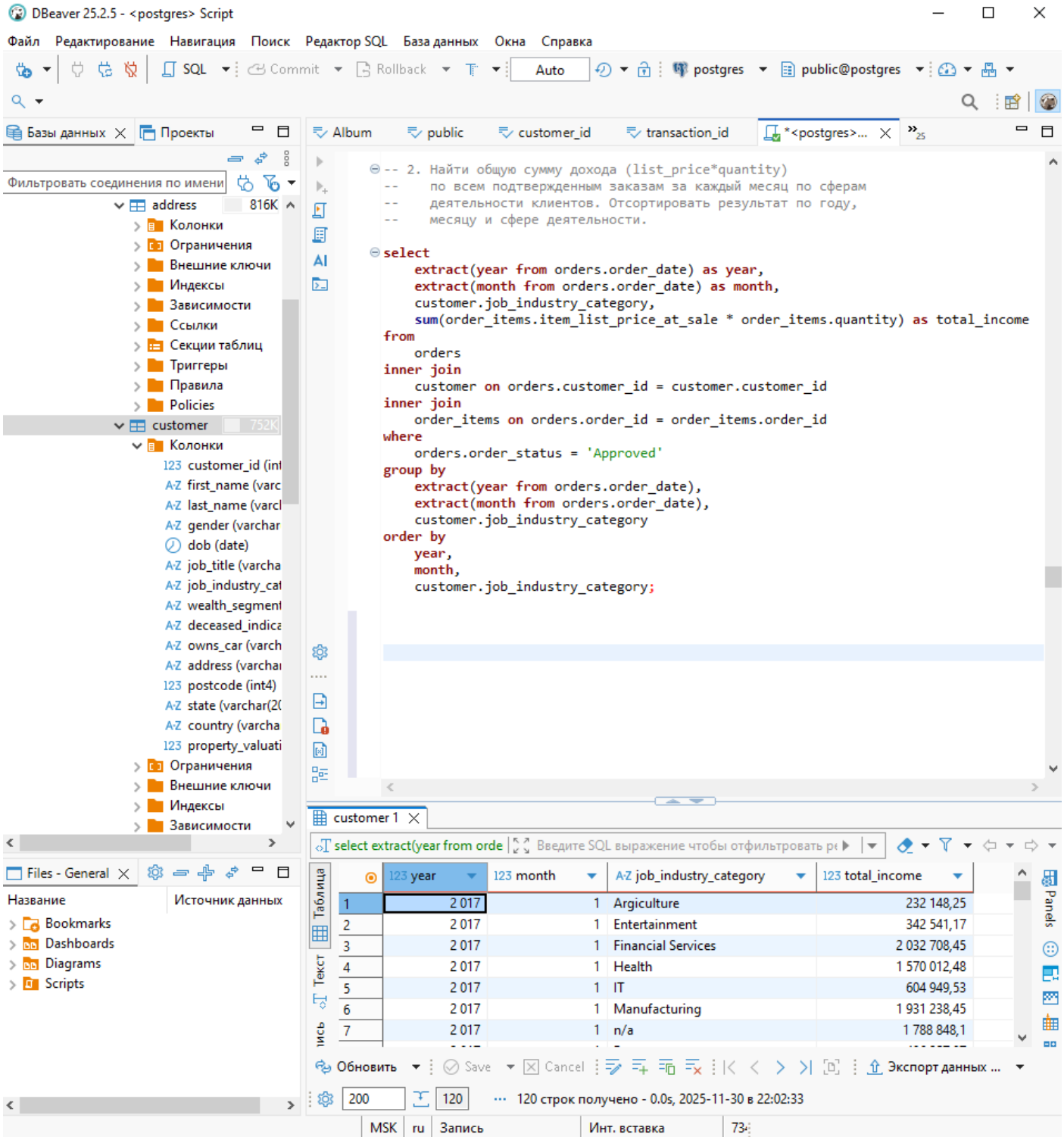

Рисунок 2 – Реализация запроса № 2
(код без подсветки почему-то, но рабочий)

3. Вывести количество уникальных онлайн-заказов для всех брендов в рамках подтвержденных заказов клиентов из сферы IT. Включить бренды, у которых нет онлайн-заказов от IT-клиентов, — для них должно быть указано количество 0.

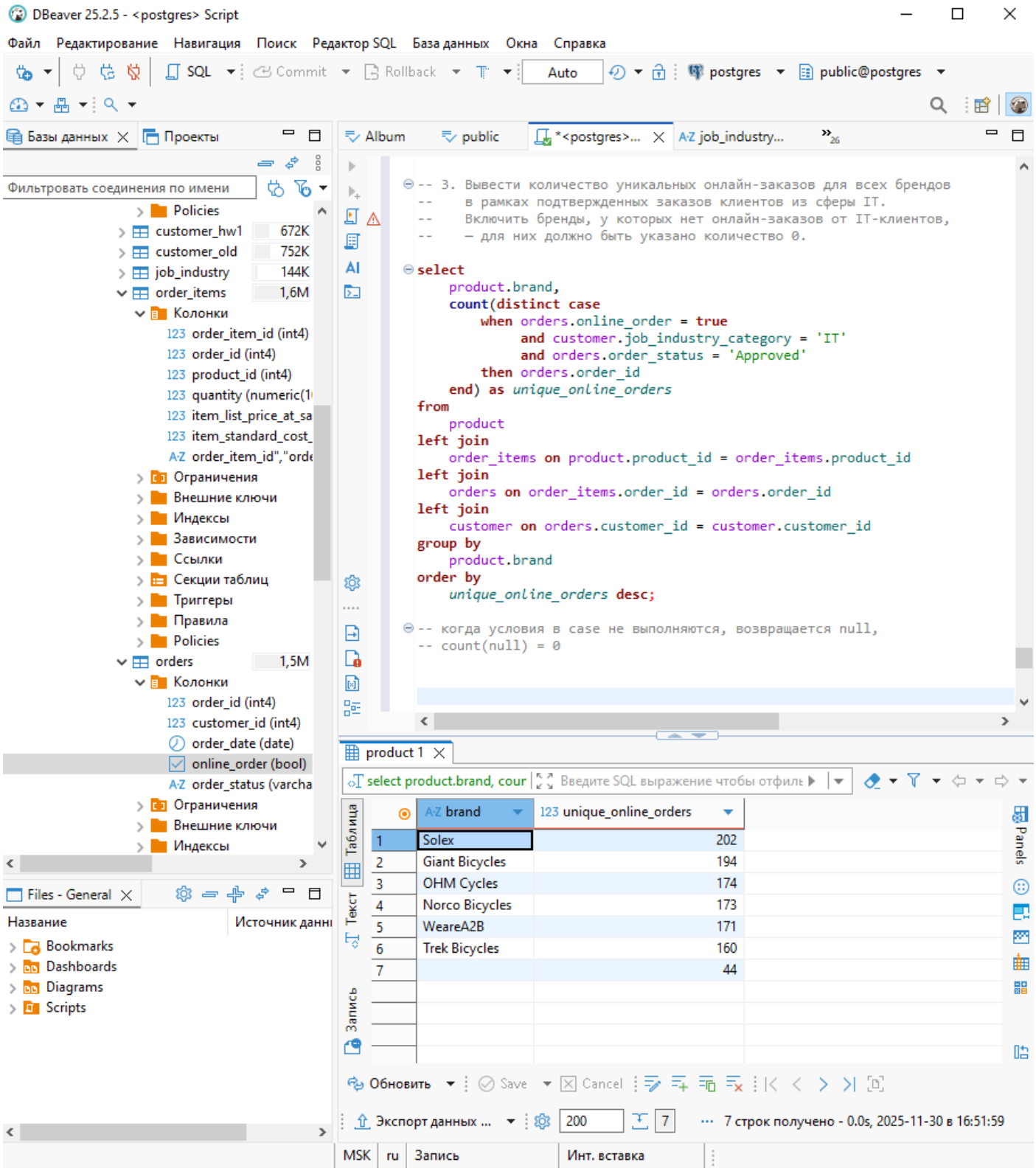

Рисунок 3 – Реализация запроса № 3

4. Найти по всем клиентам: сумму всех заказов (общего дохода), максимум, минимум и количество заказов, а также среднюю сумму заказа по каждому клиенту. Отсортировать результат по убыванию суммы всех заказов и количества заказов. Выполнить двумя способами: используя только GROUP BY и используя только оконные функции. Сравнить результат.

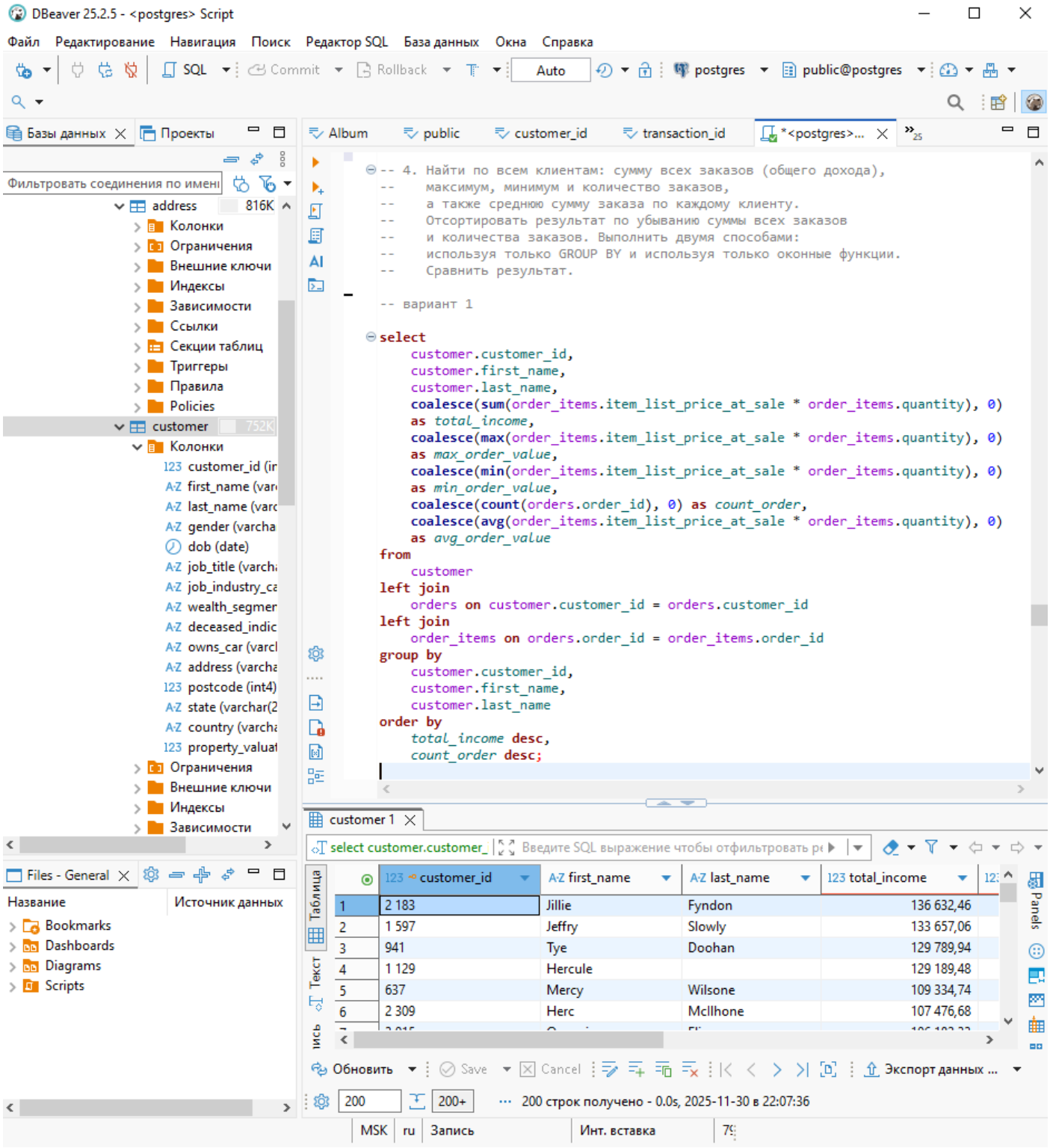

Рисунок 4 – Реализация запроса № 4
(с использованием GROUP BY)

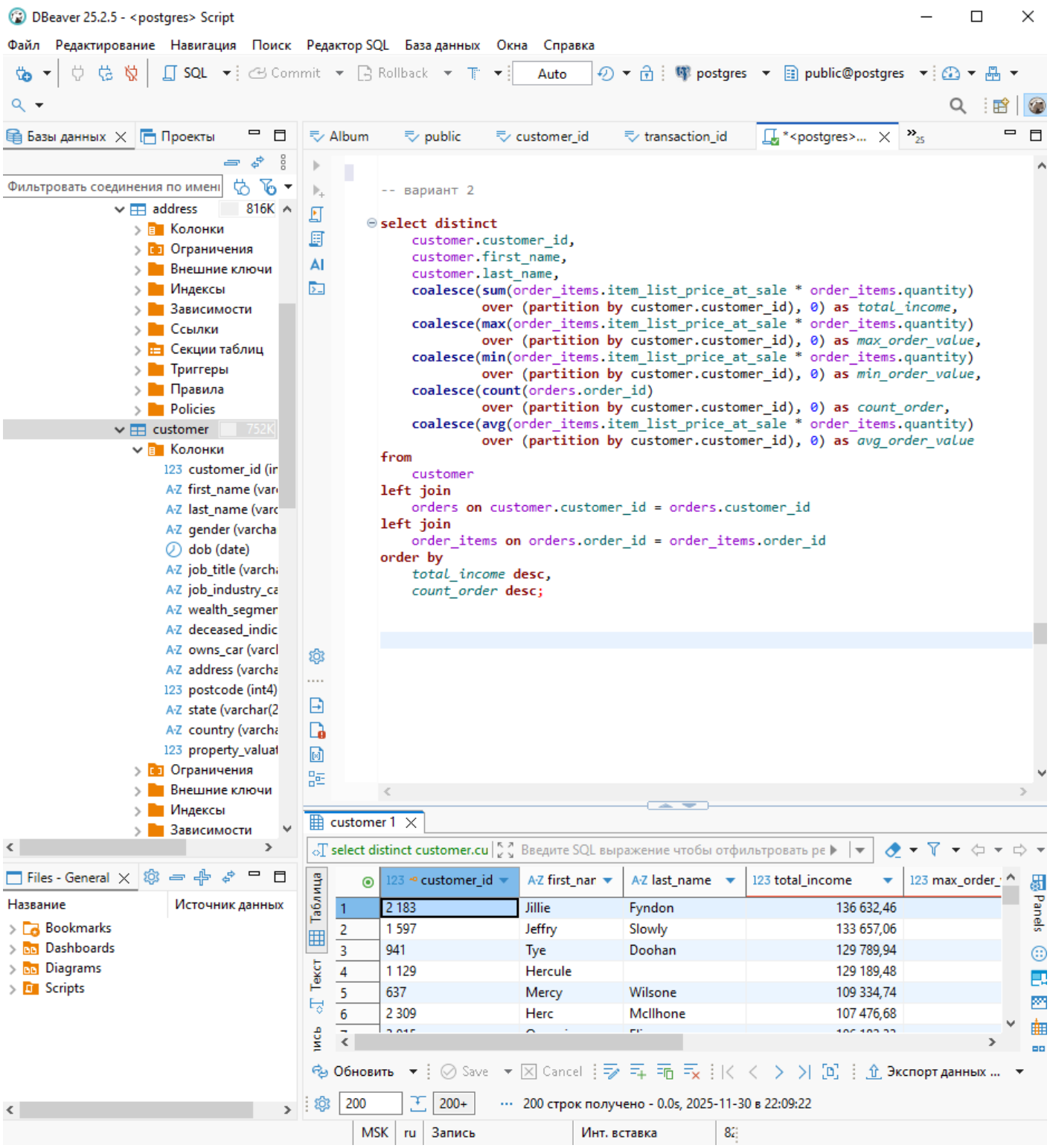

Рисунок 5 – Реализация запроса № 4
(с использованием оконных функций)

Следует отметить, что использование **`select distinct`** позволило в варианте 2 с оконными функциями получить тот же результат, что и в варианте 1. Запрос возвращает по одной строке на клиента. Просто **`select`** в варианте 2 возвращает много строк для одного клиента.

Также возможен вопрос к решению, связанный с уникальностью order_id. Уникальность была предварительно проверена и order_id в таблице orders primary key. Можно дополнительно подтвердить уникальность запросом.

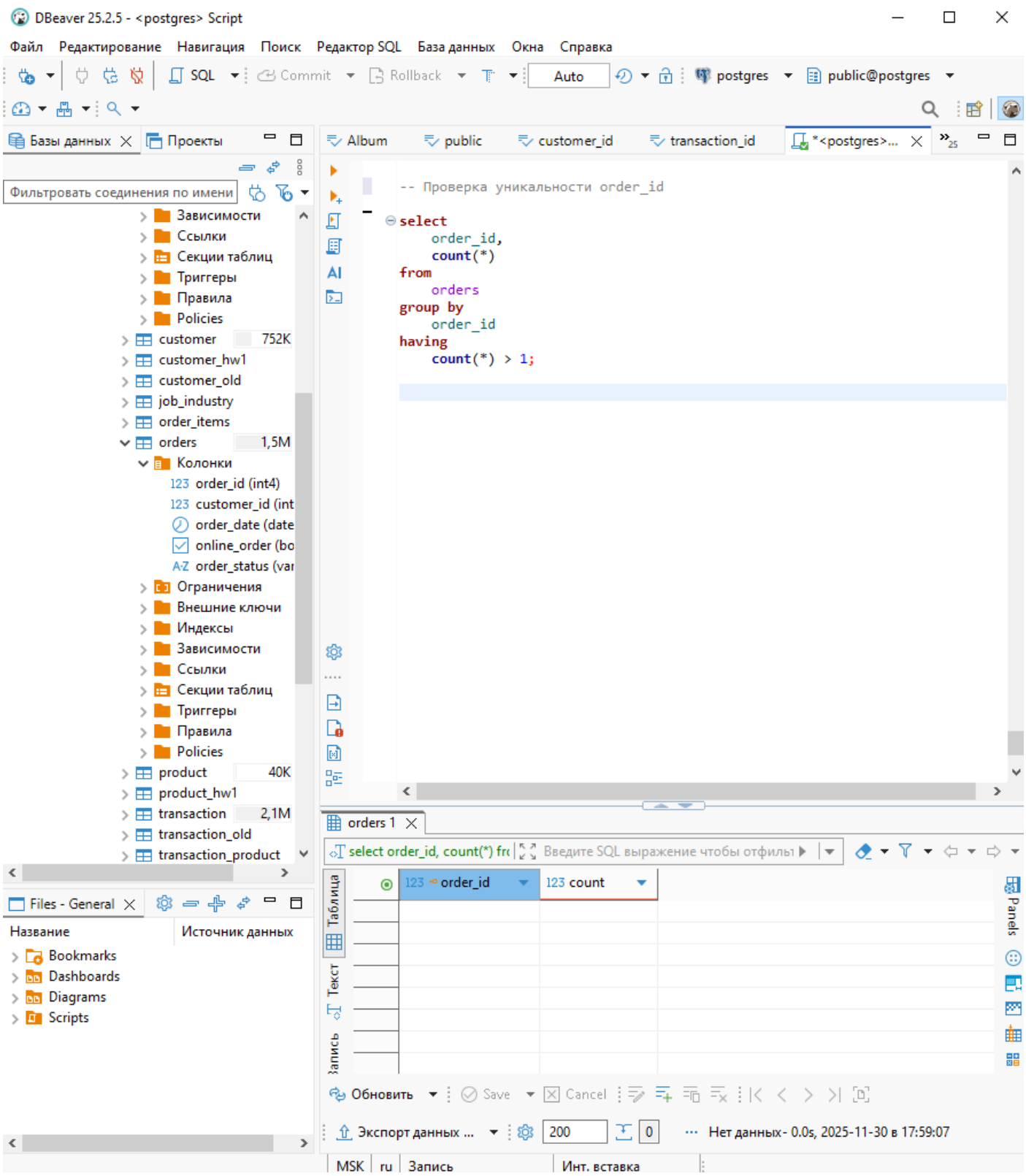

Рисунок 6 – Дополнительная проверка уникальности order_id для запроса № 4

5. Найти имена и фамилии клиентов с топ-3 минимальной и топ-3 максимальной суммой транзакций за весь период (учесть клиентов, у которых нет заказов, приняв их сумму транзакций за 0).

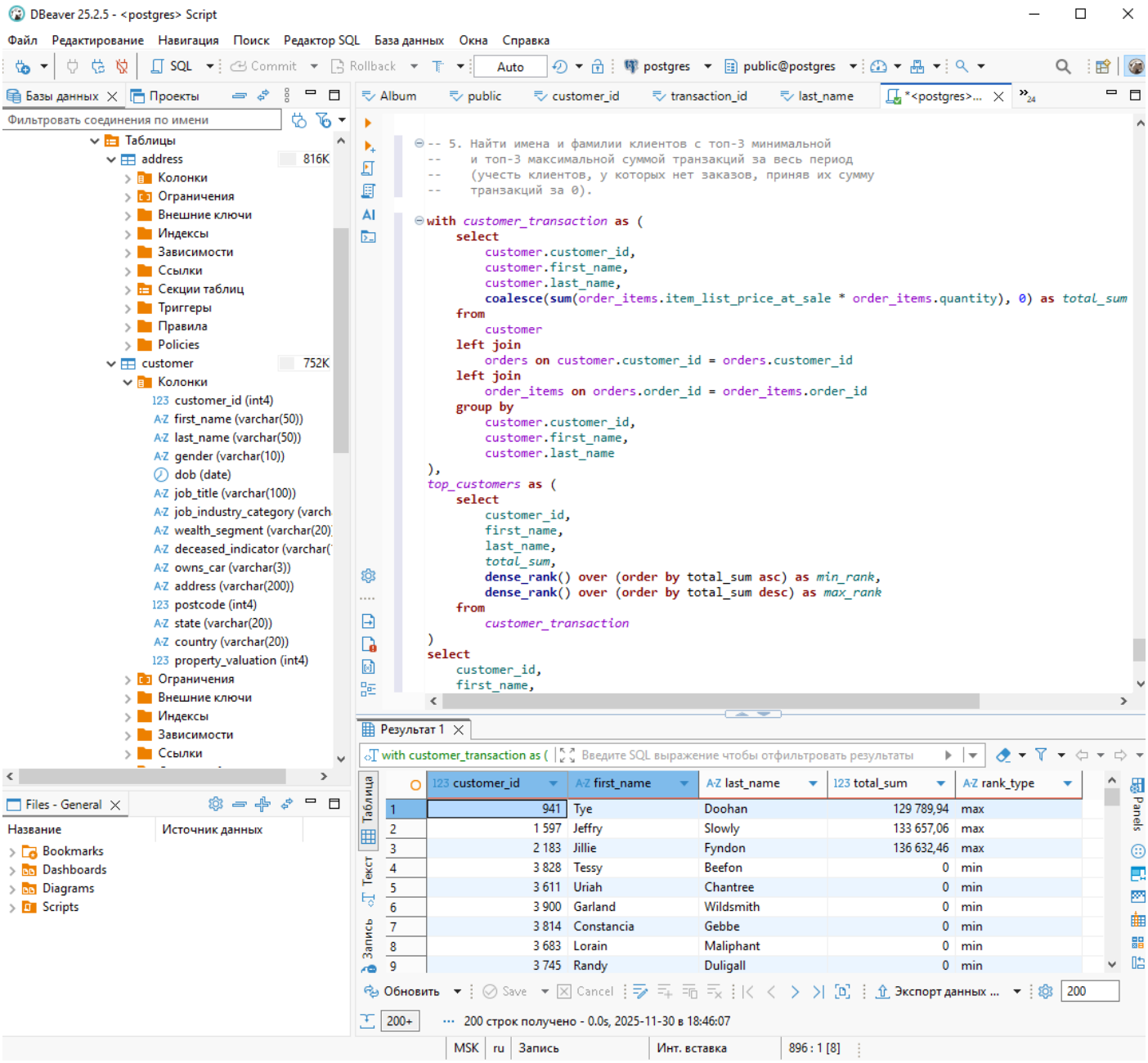

Рисунок 7 – Реализация запроса № 5

Полный код запроса представлен ниже.

```sql
with customer_transaction as (
    select
        customer.customer_id,
        customer.first_name,
        customer.last_name,
        coalesce(sum(order_items.item_list_price_at_sale * order_items.quantity), 0) as
total_sum
    from
        customer
    left join
        orders on customer.customer_id = orders.customer_id
    left join
        order_items on orders.order_id = order_items.order_id
    group by
        customer.customer_id,
        customer.first_name,
        customer.last_name
),
top_customers as (
    select
        customer_id,
        first_name,
        last_name,
        total_sum,
        dense_rank() over (order by total_sum asc) as min_rank,
        dense_rank() over (order by total_sum desc) as max_rank
    from
        customer_transaction
)
select
    customer_id,
    first_name,
    last_name,
    total_sum,
    'min' as rank_type
from
    top_customers
where
    min_rank <= 3
union all
select
    customer_id,
    first_name,
    last_name,
    total_sum,
    'max' as rank_type
from
    top_customers
where
    max_rank <= 3
order by
    rank_type,
    total_sum;
```

6. Вывести только вторые транзакции клиентов (если они есть) с помощью оконных функций. Если у клиента меньше двух транзакций, он не должен попасть в результат.

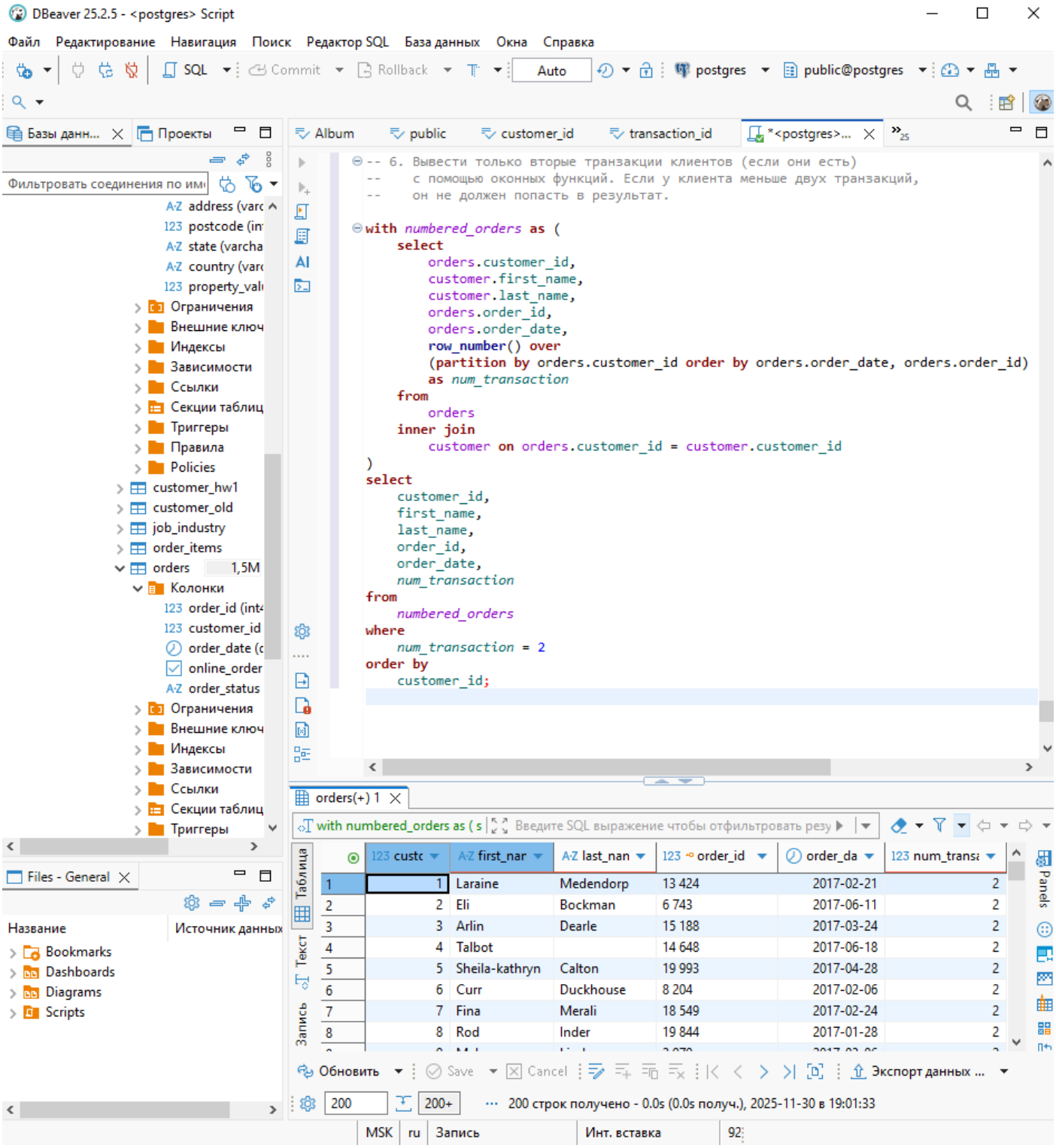

Рисунок 8 – Реализация запроса № 6

7. Вывести имена, фамилии и профессии клиентов, а также длительность максимального интервала (в днях) между двумя последовательными заказами. Исключить клиентов, у которых только один или меньше заказов.

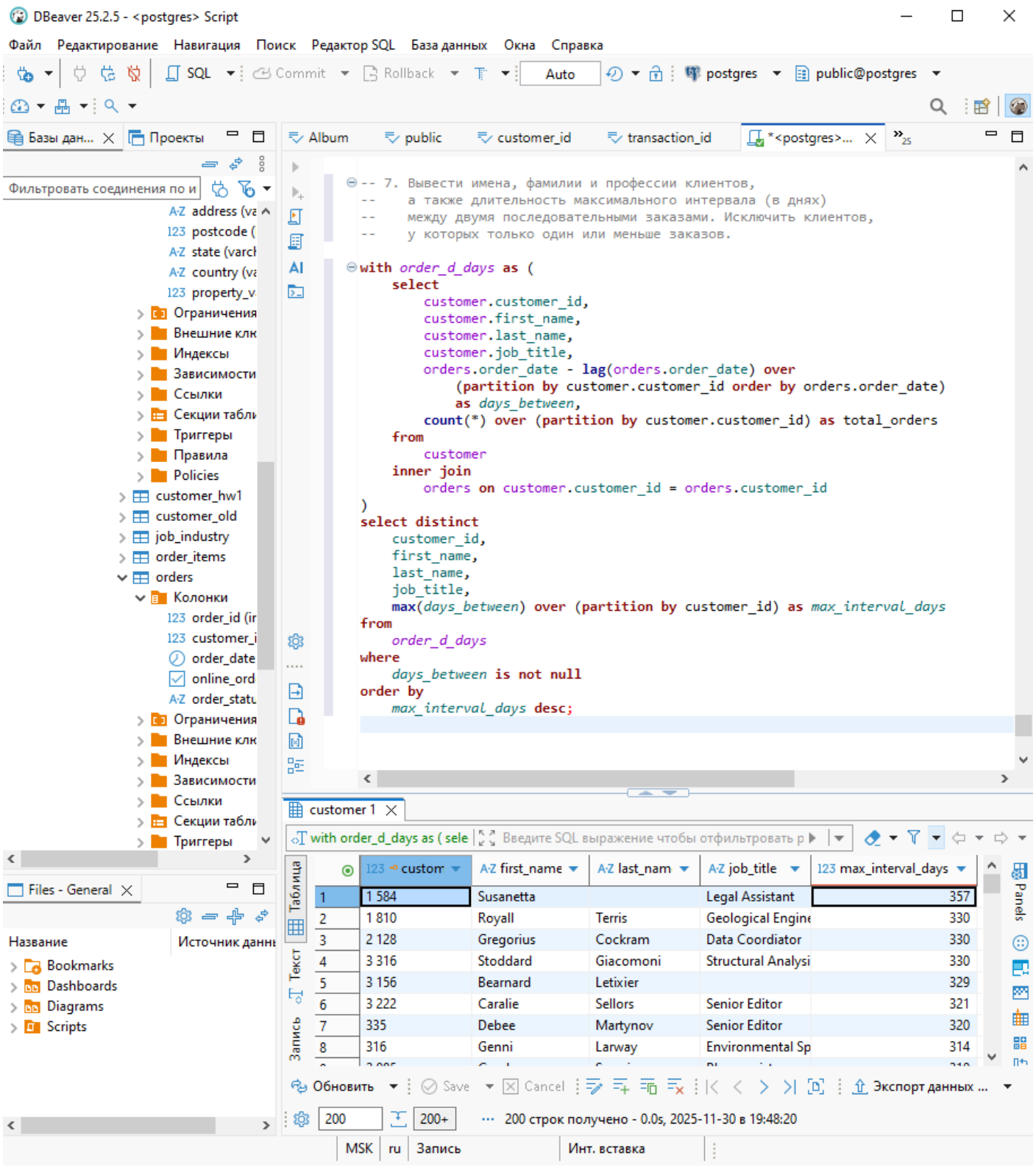

Рисунок 9 – Реализация запроса № 7

8. Найти топ-5 клиентов (по общему доходу) в каждом сегменте благосостояния (wealth_segment). Вывести имя, фамилию, сегмент и общий доход. Если в сегменте менее 5 клиентов, вывести всех.

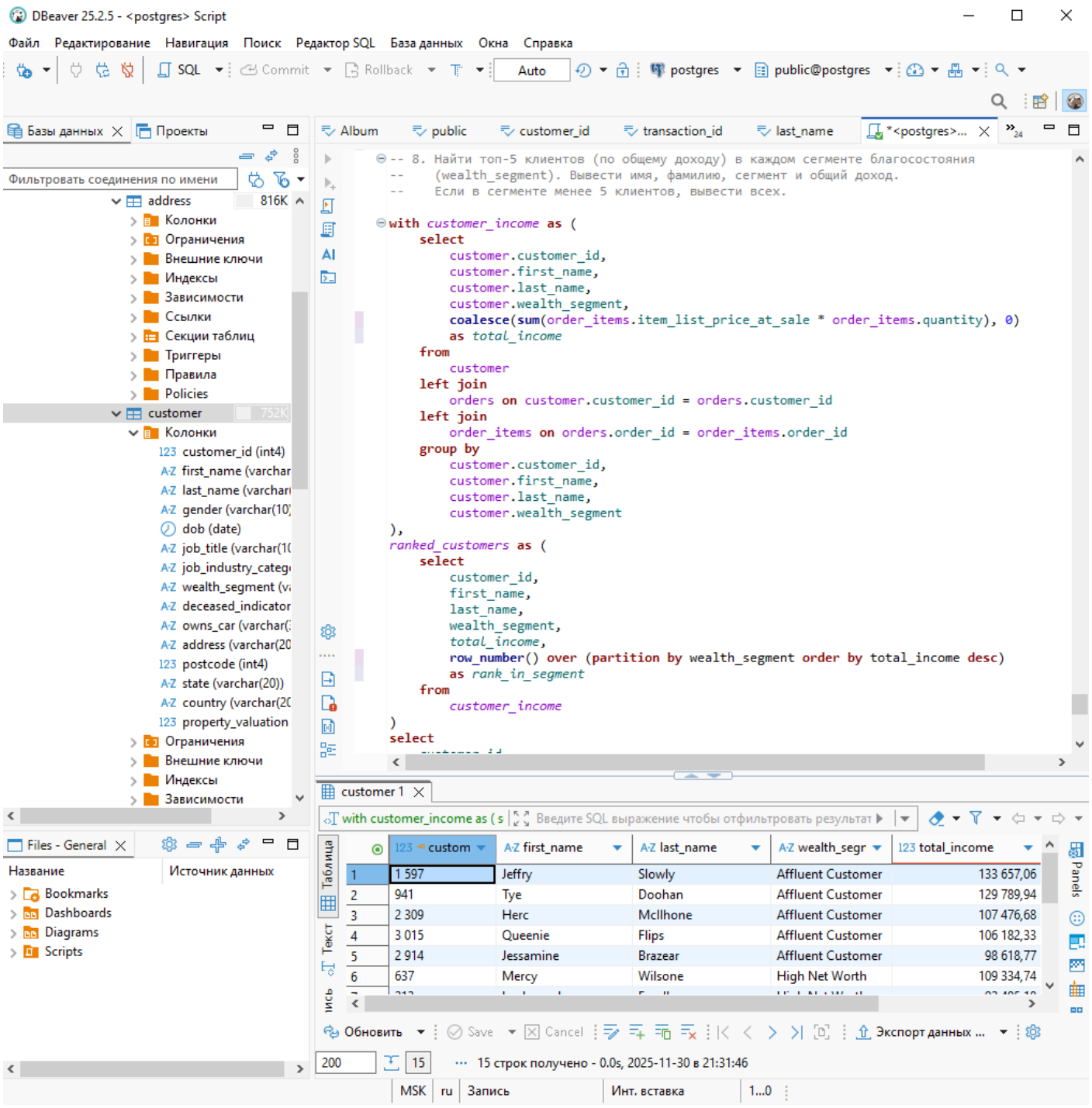

Рисунок 10 – Реализация запроса № 8

Полный код запроса представлен ниже.

```sql
with customer_income as (
    select
        customer.customer_id,
        customer.first_name,
        customer.last_name,
        customer.wealth_segment,
        coalesce(sum(order_items.item_list_price_at_sale * order_items.quantity), 0)
        as total_income
    from
        customer
    left join
        orders on customer.customer_id = orders.customer_id
    left join
        order_items on orders.order_id = order_items.order_id
    group by
        customer.customer_id,
        customer.first_name,
        customer.last_name,
        customer.wealth_segment
),
ranked_customers as (
    select
        customer_id,
        first_name,
        last_name,
        wealth_segment,
        total_income,
        row_number() over (partition by wealth_segment order by total_income desc)
        as rank_in_segment
    from
        customer_income
)
select
    customer_id,
    first_name,
    last_name,
    wealth_segment,
    total_income,
    rank_in_segment
from
    ranked_customers
where
    rank_in_segment <= 5
order by
    wealth_segment,
    rank_in_segment;
```