

## Домашнее задание 2. Основные операторы PostgreSQL

Даны CSV-файлы:

**customer.csv**

**product.csv**

**orders.csv**

**order\_items.csv**

Предварительный экспресс анализ данных выполнен с помощью pandas. Результаты сохранены в файле `pandas_analysis_hw2.ipynb`. В результате было выявлено, что из 190 `product_id` лишь 101 являются уникальными, поэтому `product_id` в таблице **product** не будет являться primary key. Кроме того, с помощью такого экспресс анализа удалось определиться с типами данных для создания соответствующих таблиц.

Создаем таблицы с перечисленными ниже структурами, используя CSV-файлы.

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)

Создание таблиц и просмотр загруженных из файлов данных приведены ниже на рисунках 1 – 5.

DBeaver 25.2.5 - <postgres> Script

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

SQL Commit Rollback Auto postgres public@postgres

Базы данных Проекты

Фильтровать соединения по имени

> DBeaver Sample Database (SQLite)

postgres localhost:5432

Базы данных

postgres

Схемы

public

Таблицы

- address 816K
- customer 752K
- customer\_hw1 672K
- customer\_old 752K
- job\_industry 144K
- order\_items 1,6M
- orders 1,5M
- product 40K
- product\_hw1 72K
- transaction 2,1M
- transaction\_old 2,5M
- transaction\_product 1,8M
- wealth\_segment 24K

Внешние таблицы

Представления

Мат. представления

Индексы

Функции

Последовательности

Типы данных

Агрегатные функции

Событийные триггеры

Расширения

Хранилище

Системные объекты

Роли

Администрирование

Files - General Источник данных

Название

Bookmarks

Dashboards

Diagrams

Scripts

\*<postgres>... customer order\_items product orders »  
-- Переименование таблиц из ДЗ 1 для того чтобы делать ДЗ 2  
alter table customer rename to customer\_hw1;  
alter table product rename to product\_hw1;  
-- ДЗ 2  
create table if not exists customer (  
 customer\_id integer primary key,  
 first\_name varchar(50) not null,  
 last\_name varchar(50) not null,  
 gender varchar(10),  
 dob date,  
 job\_title varchar(100),  
 job\_industry\_category varchar(50),  
 wealth\_segment varchar(20),  
 deceased\_indicator varchar(1),  
 owns\_car varchar(3),  
 address varchar(200),  
 postcode integer,  
 state varchar(20),  
 country varchar(20),  
 property\_valuation integer  
);  
create table if not exists product (  
 product\_id integer,  
 brand varchar(50),  
 product\_line varchar(20),  
 product\_class varchar(10),  
 product\_size varchar(10),  
 list\_price decimal(10,2),  
 standard\_cost decimal(10,2)  
);  
create table if not exists orders (  
 order\_id integer primary key,  
 customer\_id integer,  
 order\_date date,  
 online\_order boolean,  
 order\_status varchar(10)  
);  
create table if not exists order\_items (  
 order\_item\_id integer primary key,  
 order\_id integer,  
 product\_id integer,  
 quantity decimal(10,2),  
 item\_list\_price\_at\_sale decimal(10,2),  
 item\_standard\_cost\_at\_sale decimal(10,2)  
);

Рисунок 1 – Создание таблиц **customer, product, orders, order\_items**

DBeaver 25.2.5 - customer

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

Auto postgres public@postgres

**Базы данных** **Проекты**

Фильтровать соединения по имени

> DBeaver Sample Database (SQLite)

postgres localhost:5432

Базы данных postgres Схемы public Таблицы address customer Колонки customer\_id (int4) first\_name (varchar(50)) last\_name (varchar(50)) gender (varchar(10)) dob (date) job\_title (varchar(100)) job\_industry\_category (varchar(50)) wealth\_segment (varchar(20)) deceased\_indicator (varchar(1)) owns\_car (varchar(3)) address (varchar(200)) postcode (int4) state (varchar(20)) country (varchar(20)) property\_valuation (int4) Ограничения Внешние ключи Индексы Зависимости

816K 752K

customer

Свойства Данные Диаграмма

Показать SQL Введите SQL выражение чтобы отфильтровать результаты

Таблица	customer_id	first_name	last_name	gender	dob
1	1	Laraine	Medendorp	F	1953-
2	2	Eli	Bockman	Male	1980-
3	3	Arlin	Dearle	Male	1954-
4	4	Talbot		Male	1961-
5	5	Sheila-kathryn	Calton	Female	1977-
6	6	Curr	Duckhouse	Male	1966-
7	7	Fina	Merali	Female	1976-
8	8	Rod	Inder	Male	1962-
9	9	Mala	Lind	Female	1973-
10	10	Fiorenze	Birdall	Female	1988-
11	11	Uriah	Bisatt	Male	1954-
12	12	Sawyere	Flattman	Male	1994-
13	13	Gabriele	Norcross	Male	1955-
14	14	Rayshell	Kitteman	Female	1983-
15	15	Erroll	Radage	Male	2000-
16	16	Harlin	Parr	Male	1977-
17	17	Heath	Faraday	Male	1962-
18	18	Marjie	Neasham	Female	1967-
19	19	Sorcha	Keyson	Female	2001-
20	20	Basile	Firth	Male	1980-
21	21	Mile	Cammocke	Male	1980-
22	22	Deeanne	Durtnell	Female	1962-
23	23	Olav	Polak	Male	1995-
24	24	Kim	Skpsey	Female	1977-

Рисунок 2 – Просмотр данных таблицы **customer**

DBeaver 25.2.5 - product

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

Auto postgres public@postgres

**Базы данных** **Проекты**

Фильтровать соединения по имени

> DBeaver Sample Database (SQLite)

postgres localhost:5432

Базы данных postgres Схемы public Таблицы address customer customer\_hw1 customer\_old job\_industry order\_items orders product Колонки product\_id (int4) brand (varchar(50)) product\_line (varchar(20)) product\_class (varchar(10)) product\_size (varchar(10)) list\_price (numeric(10, 2)) standard\_cost (numeric(10, 2)) Ограничения Внешние ключи Индексы Зависимости Ссылки Секции таблиц

816K 752K 672K 144K 1,6M 1,5M 40K

product

Свойства Данные Диаграмма

Показать SQL Введите SQL выражение чтобы отфильтровать результаты

Таблица	product_id	brand	product_line	product_class	prod
1	14	Trek Bicycles	Standard	medium	small
2	28	Norco Bicycles	Standard	medium	small
3	0	Solex	Standard	medium	medium
4	5	Giant Bicycles	Standard	high	medium
5	22	WeareA2B	Standard	medium	medium
6	0	OHM Cycles	Standard	high	medium
7	12	WeareA2B	Standard	medium	medium
8	53	OHM Cycles	Standard	medium	medium
9	7	Giant Bicycles	Standard	medium	small
10	6	OHM Cycles	Standard	high	medium
11	0	Giant Bicycles	Standard	medium	large
12	18	Norco Bicycles	Standard	high	medium
13	87	OHM Cycles	Standard	medium	medium
14	6	Solex	Standard	high	medium
15	84	Trek Bicycles	Road	medium	medium
16	0	WeareA2B	Standard	medium	small
17	48	WeareA2B	Standard	medium	medium
18	91	WeareA2B	Standard	low	medium
19	97	OHM Cycles	Road	medium	medium
20	82	Norco Bicycles	Standard	high	medium
21	87	Giant Bicycles	Standard	high	medium
22	28	Solex	Road	medium	small
23	62	Solex	Standard	medium	medium
24	60	Giant Bicycles	Standard	high	small

Рисунок 3 – Просмотр данных таблицы **product**

DBeaver 25.2.5 - orders

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

Auto postgres public@postgres

**Базы данных** **Проекты**

Фильтровать соединения по имени

> DBeaver Sample Database (SQLite)

postgres localhost:5432

Базы данных postgres Схемы public Таблицы address customer customer\_hw1 customer\_old job\_industry order\_items orders Колонки order\_id (int4) customer\_id (int4) order\_date (date) online\_order (bool) order\_status (varchar(10)) Ограничения Внешние ключи Индексы Зависимости Ссылки Секции таблиц Триггеры Правила Policies

Текст

Показать SQL Введите SQL выражение чтобы отфильтровать результаты

Таблица

order_id	customer_id	order_date	online_order
1	2 950	2017-02-25	Appro
2	3 120	2017-05-21	Appro
3	402	2017-10-16	Appro
4	3 135	2017-08-31	Appro
5	787	2017-10-01	Appro
6	2 339	2017-03-08	Appro
7	1 542	2017-04-21	Appro
8	2 459	2017-07-15	Appro
9	1 305	2017-08-10	Appro
10	3 262	2017-08-30	Appro
11	1 986	2017-01-17	Appro
12	2 783	2017-01-05	Appro
13	1 243	2017-02-26	Appro
14	2 717	2017-09-10	Appro
15	247	2017-06-11	Appro
16	2 961	2017-10-10	Appro
17	2 426	2017-04-03	Appro
18	1 842	2017-06-02	Appro
19	2 268	2017-04-06	Appro
20	3 002	2017-01-28	Appro
21	1 582	2017-10-09	Appro
22	595	2017-06-29	Appro
23	2 001	2017-04-08	Appro
24	515	2017-10-18	Appro

Рисунок 4 – Просмотр данных таблицы orders

DBeaver 25.2.5 - order\_items

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

Auto postgres public@postgres

**Базы данных** **Проекты**

Фильтровать соединения по имени

> DBeaver Sample Database (SQLite)

postgres localhost:5432

Базы данных postgres Схемы public Таблицы address customer order\_items product orders Колонки order\_item\_id (int4) order\_id (int4) product\_id (int4) quantity (numeric(10, 2)) item\_list\_price\_at\_sale (numeric(10, 2)) item\_standard\_cost\_at\_sale (numeric(10, 2)) order\_item\_id, "order\_id", "product\_id" Ограничения Внешние ключи Индексы Зависимости Ссылки Секции таблиц Триггеры Правила Policies

Текст

Показать SQL Введите SQL выражение чтобы отфильтровать результаты

Таблица

order_item_id	order_id	product_id	quantity	item_list_price_at_sale
1	1	1	2	6
2	2	2	3	2
3	3	3	37	7
4	4	4	88	4
5	5	5	78	7
6	6	6	25	5
7	7	7	22	3
8	8	8	15	10
9	9	9	67	7
10	10	10	12	2
11	11	11	5	8
12	12	12	61	1
13	13	13	35	8
14	14	14	16	3
15	15	15	12	7
16	16	16	3	4
17	17	17	79	10
18	18	18	33	6
19	19	19	54	6
20	20	20	25	2
21	21	21	27	3
22	22	22	37	9
23	23	23	37	9
24	24	24	82	1

Рисунок 5 – Просмотр данных таблицы order\_items

## Выполнение запросов

1. Вывести все уникальные бренды, у которых есть хотя бы один продукт со стандартной стоимостью выше 1500 долларов, и суммарными продажами не менее 1000 единиц.

The screenshot shows the DBeaver 25.2.5 interface with the following details:

- Left Panel (Object Navigator):** Shows the database schema with tables `orders` and `product`. The `orders` table has 1.5M rows and the `product` table has 40K rows.
- Central Panel (SQL Editor):** Displays the SQL query being executed:

```
-- 1. Вывести все уникальные бренды, у которых есть хотя бы один продукт
-- со стандартной стоимостью выше 1500 долларов,
-- и суммарными продажами не менее 1000 единиц.

select distinct
    product.brand
from
    product
where
    product.standard_cost > 1500
    and product.product_id in (
        select
            order_items.product_id
        from
            order_items
        group by
            order_items.product_id
        having
            sum(order_items.quantity) >= 1000
);
```
- Bottom Panel (Results):** Shows the results of the query in a table format. The table has one column labeled "brand".

brand
Giant Bicycles
OHM Cycles

Details at the bottom of the results panel:
  - 200 rows selected
  - 2 rows found
  - 0.0s execution time
  - 2025-11-23 19:40:29 timestamp

Рисунок 6 – Реализация запроса № 1

2. Для каждого дня в диапазоне с 2017-04-01 по 2017-04-09 включительно вывести количество подтвержденных онлайн-заказов и количество уникальных клиентов, совершивших эти заказы.

The screenshot shows the DBeaver interface with the following details:

- Left Panel (Object Navigator):** Shows the database structure for the 'postgres' schema. It includes tables like address, customer, customer\_hw1, customer\_old, job\_industry, order\_items, orders, and product. The 'orders' table is currently selected.
- Center Panel (SQL Editor):** Displays the SQL query:
 

```
-- 2. Для каждого дня в диапазоне с 2017-04-01 по 2017-04-09 включительно
-- вывести количество подтвержденных онлайн-заказов и количество уникальных
-- клиентов, совершивших эти заказы.

select
    order_date,
    count(*) as number_of_confirmed_online_orders,
    count(distinct customer_id) as number_of_unique_customers
from
    orders
where
    order_date between '2017-04-01' and '2017-04-09'
    and online_order = true
    and order_status = 'Approved'
group by
    order_date
order by
    order_date;
```
- Bottom Panel (Results View):** Shows the results of the query in a table named 'orders 1'. The table has three columns: 'order\_date', 'number\_of\_confirmed\_online\_orders', and 'number\_of\_unique\_customers'. The data is as follows:
 

order_date	number_of_confirmed_online_orders	number_of_unique_customers
2017-04-01	37	37
2017-04-02	29	29
2017-04-03	27	27
2017-04-04	32	32
2017-04-05	33	32
2017-04-06	36	36
2017-04-07	24	24
2017-04-08	33	33
2017-04-09	30	30

Рисунок 7 – Реализация запроса № 2

3. Вывести профессии клиентов:

- из сферы IT, чья профессия начинается с Senior;
  - из сферы Financial Services, чья профессия начинается с Lead.
- Для обеих групп учитывать только клиентов старше 35 лет. Объединить выборки с помощью UNION ALL.

The screenshot shows the DBeaver interface with the following details:

- SQL Editor:** Contains the following SQL query:
 

```
-- 3. Вывести профессии клиентов:
--   - из сферы IT, чья профессия начинается с Senior;
--   - из сферы Financial Services, чья профессия начинается с Lead.
-- Для обеих групп учитывать только клиентов старше 35 лет.
-- Объединить выборки с помощью UNION ALL.

select distinct
    job_title as profession
from
    customer
where
    job_industry_category = 'IT'
    and job_title like 'Senior%'
    and extract(year from age(current_date, dob)) > 35

union all

select distinct
    job_title as profession
from
    customer
where
    job_industry_category = 'Financial Services'
    and job_title like 'Lead%'
    and extract(year from age(current_date, dob)) > 35;
```
- Results Panel:** Shows the output of the query in a table format:
 

	AZ profession
1	Senior Developer
2	Senior Sales Associate
- Left Sidebar:** Shows the database structure for the 'customer' table, including columns like customer\_id, first\_name, last\_name, gender, dob, job\_title, job\_industry\_category, wealth\_segment, deceased\_indicator, owns\_car, address, postcode, state, country, and property\_valuation.
- Bottom Status Bar:** Displays the message "2 строки получено - 0.0s (0.0s получ.), 2025-11-23 в 15:18:20".

Рисунок 8 – Реализация запроса № 3

Оказалось, что клиенты из сферы Financial Services не имеют профессий, которые начинаются со слова Lead.

4. Вывести бренды, которые были куплены клиентами из сферы Financial Services, но не были куплены клиентами из сферы IT.

The screenshot shows the DBeaver interface with the following details:

- Toolbar:** Includes File, Редактирование (Edit), Навигация (Navigation), Поиск (Search), Редактор SQL (SQL Editor), База данных (Database), Okna (Windows), Справка (Help).
- Left Panel (Object Navigator):**
  - Базы данных (Databases) node has a 'Фильтровать соединения по имени' (Filter connections by name) input field.
  - orders node has 1,5M rows.
  - product node has 40K rows.
- Central Panel (SQL Editor):**

```

-- 4. Вывести бренды, которые были куплены клиентами из сферы Financial Services,
-- но не были куплены клиентами из сферы IT.

select distinct
    product.brand
from
    orders
inner join
    customer on orders.customer_id = customer.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
inner join
    product on order_items.product_id = product.product_id
where
    customer.job_industry_category = 'Financial Services'

except

select distinct
    product.brand
from
    orders
inner join
    customer on orders.customer_id = customer.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
inner join
    product on order_items.product_id = product.product_id
where
    customer.job_industry_category = 'IT';
  
```
- Bottom Panel (Results):**
  - Result set titled 'Результат 1' (Result 1) is displayed.
  - Table view shows a single column 'brand' with one row labeled 'AZ brand'.
  - Status bar at the bottom indicates 'Нет данных - 0.0s, 2025-11-23 в 16:16:04' (No data - 0.0s, 2025-11-23 at 16:16:04).

Рисунок 9 – Реализация запроса № 4 (вариант-1)

Запрос показал, что таких брендов, которые были куплены клиентами из сферы Financial Services, но не были куплены клиентами из сферы IT нет.

Представим еще один вариант этого запроса

DBeaver 25.2.5 - <postgres> Script

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

SQL Commit Rollback Auto postgres public@postgres

**Базы данных** X **Проекты**

Фильтровать соединения по имени

Зависимости  
Ссылки  
Секции таблиц  
Триггеры  
Правила  
Policies  
orders  
Колонки  
123 order\_id (int4)  
123 customer\_id (int4)  
order\_date (date)  
online\_order (bool)  
AZ order\_status (varchar(10))  
Ограничения  
Внешние ключи  
Индексы  
Зависимости  
Ссылки  
Секции таблиц  
Триггеры  
Правила  
Policies  
product  
Колонки  
123 product\_id (int4)  
AZ brand (varchar(50))  
AZ product\_line (varchar(20))  
AZ product\_class (varchar(10))  
AZ product\_size (varchar(10))  
123 list\_price (numeric(10, 2))  
123 standard\_cost (numeric(10, 2))  
Ограничения  
Внешние ключи

1,5M 40K

\*<postgres>... X customer order\_items product orders >24

```

select distinct
    financial_services_brands.brand
from (
    select distinct
        product.brand
    from
        orders
    inner join
        customer on orders.customer_id = customer.customer_id
    inner join
        order_items on orders.order_id = order_items.order_id
    inner join
        product on order_items.product_id = product.product_id
    where
        customer.job_industry_category = 'Financial Services'
) as financial_services_brands
left join (
    select distinct
        product.brand
    from
        orders
    inner join
        customer on orders.customer_id = customer.customer_id
    inner join
        order_items on orders.order_id = order_items.order_id
    inner join
        product on order_items.product_id = product.product_id
    where
        customer.job_industry_category = 'IT'
) as it_brands on financial_services_brands.brand = it_brands.brand
where
    it_brands.brand is null;

```

product 1 X

select distinct financial\_ser

Ведите SQL выражение чтобы отфильтровать результат

Таблица AZ brand

Обновить Save Cancel Экспорт данных ...

200 0 Нет данных - 0.0s, 2025-11-23 в 16:18:44

MSK | Запись | Инт. вставка | 462...985

Рисунок 10 – Реализация запроса № 4 (вариант-2)

Запрос также показал, что брендов, которые были куплены клиентами из сферы Financial Services, но не были куплены клиентами из сферы IT нет.

Проверим какие бренды покупали клиенты из этих двух сфер.

DBeaver 25.2.5 - <postgres> Script

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

SQL Commit Rollback Auto postgres public@postgres

**Базы данных** X **Проекты**

Фильтровать соединения по имени

Зависимости  
Ссылки  
Секции таблиц  
Триггеры  
Правила  
Policies  
orders  
Колонки  
order\_id (int4)  
customer\_id (int4)  
order\_date (date)  
online\_order (bool)  
order\_status (varchar(10))  
Ограничения  
Внешние ключи  
Индексы  
Зависимости  
Ссылки  
Секции таблиц  
Триггеры  
Правила  
Policies  
product  
Колонки  
product\_id (int4)  
brand (varchar(50))  
product\_line (varchar(20))  
product\_class (varchar(10))  
product\_size (varchar(10))  
list\_price (numeric(10, 2))  
standard\_cost (numeric(10, 2))  
Ограничения  
Внешние ключи

1,5M 40K

\*<postgres>... X customer order\_items product orders >24

```

-- Проверим, какие бренды вообще покупали клиенты Financial Services
select distinct
    product.brand
from
    orders
inner join
    customer on orders.customer_id = customer.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
inner join
    product on order_items.product_id = product.product_id
where
    customer.job_industry_category = 'Financial Services'
order by
    product.brand;

-- Проверим, какие бренды вообще покупали клиенты IT
select distinct
    product.brand
from
    orders
inner join
    customer on orders.customer_id = customer.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
inner join
    product on order_items.product_id = product.product_id
where
    customer.job_industry_category = 'IT'
order by
    product.brand;

```

product 1

select distinct product.brand | Введите SQL выражение чтобы отфильтровать результат

	AZ brand
1	
2	Giant Bicycles
3	Norco Bicycles
4	OHM Cycles
5	Solex
6	Trek Bicycles
7	WeareA2B

Обновить Save Cancel | < < < > > | Экспорт данных ...

200 7 7 строк получено - 0.0s, 2025-11-23 в 16:26:13

Рисунок 11 – Проверка запроса № 4

Запросы на проверку, какие бренды покупали клиенты из сферы Financial Services и из сферы IT,

показали один и тот же список брендов.

Таким образом, бренды, которые были куплены клиентами из сферы Financial Services, но не были куплены клиентами из сферы IT нет.

5. Вывести 10 клиентов (ID, имя, фамилия), которые совершили наибольшее количество онлайн-заказов (в штуках) брендов Giant Bicycles, Norco Bicycles, Trek Bicycles, при условии, что они активны и имеют оценку имущества (property\_valuation) выше среднего среди клиентов из того же штата.

The screenshot shows the DBeaver interface with the following details:

- Toolbar:** Includes icons for connection status, file operations (New, Open, Save, Import, Export), SQL editor, commit/rollback, and various database management functions.
- Menu Bar:** Файл (File), Редактирование (Edit), Навигация (Navigation), Поиск (Search), Редактор SQL (SQL Editor), База данных (Database), Окна (Windows), Справка (Help).
- Left Panel (Bases de datos):**
  - customer table (selected):
    - Колонки (Columns): customer\_id (int4), first\_name (varchar(50)), last\_name (varchar(50)), gender (varchar(10)), dob (date), job\_title (varchar(100)), job\_industry\_category (varchar(50)), wealth\_segment (varchar(20)), deceased\_indicator (varchar(1)), owns\_car (varchar(3)), address (varchar(200)), postcode (int4), state (varchar(20)), country (varchar(20)), property\_valuation (int4).
    - Ограничения (Constraints): primary key customer\_id, foreign keys customer\_hw1, customer\_old, job\_industry, order\_items.
    - Внешние ключи (Foreign Keys): customer\_hw1, customer\_old, job\_industry, order\_items.
    - Индексы (Indexes):
    - Зависимости (Dependencies):
    - Ссылки (References):
    - Секции таблиц (Table Sections):
    - Триггеры (Triggers):
    - Правила (Rules):
    - Policies:
  - order\_items table:
    - Колонки (Columns): order\_item\_id (int4).
- Central SQL Editor:**

```
-- 5. Вывести 10 клиентов (ID, имя, фамилия), которые совершили
-- наибольшее количество онлайн-заказов (в штуках) брендов Giant Bicycles,
-- Norco Bicycles, Trek Bicycles, при условии,
-- что они активны и имеют оценку имущества (property_valuation)
-- выше среднего среди клиентов из того же штата.

with state_avg_property as (
    select
        state,
        avg(property_valuation) as avg_property_valuation
    from
        customer
    where
        deceased_indicator = 'N'
    group by
        state
),
filtered_customers as (
    select
        customer.customer_id,
        customer.first_name,
        customer.last_name,
        customer.state,
        customer.property_valuation
    from
        customer
    inner join
        state_avg_property on customer.state = state_avg_property.state
    where
        customer.deceased_indicator = 'N'
        and customer.property_valuation > state_avg_property.avg_property_valuation
),
online_orders_count as (

```
- Bottom Result Grid:**

	customer_id	first_name	last_name
1	353	Antonia	Cardis
2	3 221	Brigid	Quigley
3	1 302	Ericka	Eggers
4	534	Madel	Palffrey
5	787	Norma	Batrim
6	1 117	Georgena	Guilaem
7	25	Geoff	Assaf
8	1	Laraine	Medendorp
9	478	Darcey	Harberer
10	714	Burtie	Scisbury

Обновить Save Cancel Экспорт данных ...

Рисунок 12 – Реализация запроса № 5

Полная версия запроса представлена ниже.

```

with state_avg_property as (
    select
        state,
        avg(property_valuation) as avg_property_valuation
    from
        customer
    where
        deceased_indicator = 'N'
    group by
        state
),
filtered_customers as (
    select
        customer.customer_id,
        customer.first_name,
        customer.last_name,
        customer.state,
        customer.property_valuation
    from
        customer
    inner join
        state_avg_property on customer.state = state_avg_property.state
    where
        customer.deceased_indicator = 'N'
        and customer.property_valuation > state_avg_property.avg_property_valuation
),
online_orders_count as (
    select
        filtered_customers.customer_id,
        filtered_customers.first_name,
        filtered_customers.last_name,
        count(distinct orders.order_id) as online_orders_count
    from
        filtered_customers
    inner join
        orders on filtered_customers.customer_id = orders.customer_id
    inner join
        order_items on orders.order_id = order_items.order_id
    inner join
        product on order_items.product_id = product.product_id
    where
        orders.online_order = true
        and orders.order_status = 'Approved'
        and product.brand in ('Giant Bicycles', 'Norco Bicycles', 'Trek Bicycles')
    group by
        filtered_customers.customer_id,
        filtered_customers.first_name,
        filtered_customers.last_name
)
select
    customer_id,
    first_name,
    last_name
from
    online_orders_count
order by
    online_orders_count desc
limit 10;

```

Запрос можно улучшить и выводить для наглядности также количество заказов, изменив последние 9 строчек.

DBeaver 25.2.5 - <postgres> Script

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

SQL Commit Rollback Auto postgres public@postgres

**Базы данных** X **Проекты**

Фильтровать соединения по имени

address 816K  
customer 752K

customer 752K

Колонки

- customer\_id (int4)
- first\_name (varchar(50))
- last\_name (varchar(50))
- gender (varchar(10))
- dob (date)
- job\_title (varchar(100))
- job\_industry\_category (varchar(50))
- wealth\_segment (varchar(20))
- deceased\_indicator (varchar(1))
- owns\_car (varchar(3))
- address (varchar(200))
- postcode (int4)
- state (varchar(20))
- country (varchar(20))
- property\_valuation (int4)

Ограничения  
Внешние ключи  
Индексы  
Зависимости  
Ссылки  
Секции таблиц  
Триггеры  
Правила  
Policies

customer\_hw1 672K  
customer\_old 752K  
job\_industry 144K  
order\_items 1,6M

Колонки

- order\_item\_id (int4)
- order\_id (int4)

customer 1 X

with state\_avg\_property as

```

filtered_customers.customer_id,
filtered_customers.first_name,
filtered_customers.last_name,
count(distinct orders.order_id) as online_orders_count
from
filtered_customers
inner join
orders on filtered_customers.customer_id = orders.customer_id
inner join
order_items on orders.order_id = order_items.order_id
inner join
product on order_items.product_id = product.product_id
where
orders.online_order = true
and orders.order_status = 'Approved'
and product.brand in ('Giant Bicycles', 'Norco Bicycles', 'Trek Bicycles')
group by
filtered_customers.customer_id,
filtered_customers.first_name,
filtered_customers.last_name
)
select
customer_id,
first_name,
last_name,
online_orders_count
from
online_orders_count
order by
online_orders_count desc
limit 10;

```

	customer_id	first_name	last_name	online_orders_count
1	353	Antonia	Cardis	7
2	3 221	Brigid	Quigley	7
3	1 302	Ericka	Eggers	6
4	534	Madel	Palffrey	6
5	787	Norma	Batrim	6
6	1 117	Georgena	Guilaem	6
7	25	Geoff	Assaf	6
8	1	Laraine	Medendorp	6
9	478	Darcey	Harberer	6
10	714	Burtie	Scintsbury	6

Обновить Save Cancel Экспорт данных ...

200 10 10 строк получено - 0.0s, 2025-11-23 в 17:31:02

Рисунок 13 – Реализация запроса № 5 (с выводом количества заказов)

6. Вывести всех клиентов (ID, имя, фамилия), у которых нет подтвержденных онлайн-заказов за последний год, но при этом они владеют автомобилем и их сегмент благосостояния не Mass Customer.

The screenshot shows the DBeaver interface with the following details:

- Toolbar:** Includes File, Редактирование (Edit), Навигация (Navigation), Поиск (Search), Редактор SQL (SQL Editor), База данных (Database), Okna (Windows), Справка (Help).
- Left Panel (Bases de datos):** Shows the database structure for the 'customer' table, including columns like customer\_id, first\_name, last\_name, gender, dob, job\_title, job\_industry\_category, wealth\_segment, deceased\_indicator, owns\_car, address, postcode, state, country, property\_valuation, and constraints like 'owns\_car = 'Yes''. Other tables like address, order\_items, product, and orders are also listed.
- Central Panel (SQL Editor):** Displays the SQL query for question 6:

```

-- 6. Вывести всех клиентов (ID, имя, фамилия),
-- у которых нет подтвержденных онлайн-заказов за последний год,
-- но при этом они владеют автомобилем
-- и их сегмент благосостояния не Mass Customer.

select
    customer_id,
    first_name,
    last_name
from
    customer
where
    owns_car = 'Yes'
    and wealth_segment != 'Mass Customer'
    and customer_id not in (
        select distinct
            customer_id
        from
            orders
        where
            online_order = true
            and order_status = 'Approved'
            and order_date >= current_date - interval '1 year'
    );

```
- Bottom Panel (Results):** Shows the results of the query in a table named 'customer 1'. The table has columns customer\_id, first\_name, and last\_name. The data is as follows:

	customer_id	first_name	last_name
1	5	Sheila-kathryn	Calton
2	6	Curr	Duckhouse
3	7	Fina	Merali
4	9	Mala	Lind
5	13	Gabriele	Norcross
6	17	Heath	Faraday
7	21	Mile	Cammocke
8	23	Olav	Polak
9	28	Fee	Zellmer
10	30	Derrick	Hellekas
11	31	Star	Praton

Рисунок 14 – Реализация запроса № 6

По запросу было выведено достаточно много клиентов. Это связано с тем, что заказы, в основном, датируются 2017 годом.

7. Вывести всех клиентов из сферы IT (ID, имя, фамилия), которые купили 2 из 5 продуктов с самой высокой list\_price в продуктовой линейке Road.

The screenshot shows the DBeaver interface with the following details:

- SQL Editor:** Contains the following SQL query:
 

```
-- 7. Вывести всех клиентов из сферы IT (ID, имя, фамилия),
-- которые купили 2 из 5 продуктов с самой высокой list_price
-- в продуктовой линейке Road.

select distinct
    customer.customer_id,
    customer.first_name,
    customer.last_name
from
    customer
inner join
    orders on customer.customer_id = orders.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
inner join
    product on order_items.product_id = product.product_id
where
    customer.job_industry_category = 'IT'
    and product.product_line = 'Road'
    and product.product_id in (
        select
            product_id
        from
            product
        where
            product_line = 'Road'
        order by
            list_price desc
        limit 5
    )
group by
    customer.customer_id,
    customer.first_name,
    customer.last_name
having count(distinct product.product_id) = 2;
```
- Results Viewer:** Shows a table titled "customer 1" with the following data:
 

	customer_id	first_name	last_name
1	799	Harland	Spilisy
2	983	Shaylyn	Riggs
3	1 683	Brenn	Bacon
4	1 791	Ninon	Van Der Hoog
5	1 820	Yard	Teeney
6	1 887	Kynthia	Purcer
7	3 406	Lucy	Lackmann

Рисунок 15 – Реализация запроса № 7

Недостатком такого варианта запроса является то, что он не учитывает тех клиентов, которые купили более 2 товаров из топ-5 с высокой list\_price. Поэтому логичнее было бы заменить последнюю строку

`having count(distinct product.product_id) = 2;`

на

`having count(distinct product.product_id) >= 2;`

DBeaver 25.2.5 - <postgres> Script

Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка

SQL Commit Rollback Auto postgres public@postgres

**Базы данных** X **Проекты**

Фильтровать соединения по имени

address 816K  
customer 752K

Колонки

- customer\_id (int4)
- first\_name (varchar(50))
- last\_name (varchar(50))
- gender (varchar(10))
- dob (date)
- job\_title (varchar(100))
- job\_industry\_category (varchar(50))
- wealth\_segment (varchar(20))
- deceased\_indicator (varchar(1))
- owns\_car (varchar(3))
- address (varchar(200))
- postcode (int4)
- state (varchar(20))
- country (varchar(20))
- property\_valuation (int4)

Ограничения 672K  
Внешние ключи 752K  
Индексы 144K  
Зависимости 1,6M

Секции таблиц  
Тriggers  
Правила  
Policies

customer\_hwl  
customer\_old  
job\_industry  
order\_items  
Колонки

order\_item\_id (int4)

**<postgres>... X customer order\_items product orders**

```
-- 7. Вывести всех клиентов из сферы IT (ID, имя, фамилия),
-- которые купили 2 из 5 продуктов с самой высокой list_price
-- в продуктовой линейке Road.

select distinct
    customer.customer_id,
    customer.first_name,
    customer.last_name
from
    customer
inner join
    orders on customer.customer_id = orders.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
inner join
    product on order_items.product_id = product.product_id
where
    customer.job_industry_category = 'IT'
    and product.product_line = 'Road'
    and product.product_id in (
        select
            product_id
        from
            product
        where
            product_line = 'Road'
        order by
            list_price desc
        limit 5
    )
group by
    customer.customer_id,
    customer.first_name,
    customer.last_name
having count(distinct product.product_id) >= 2;
```

**customer 1 X**

	customer_id	first_name	last_name
1	799	Harland	Spilisy
2	983	Shaylyn	Riggs
3	1 683	Brenn	Bacon
4	1 791	Ninon	Van Der Hoog
5	1 820	Yard	Teeney
6	1 887	Kynthia	Purcer
7	3 406	Lucy	Lackmann

Обновить Save Cancel Экспорт данных ...  
200 7 7 строк получено - 0.0s, 2025-11-23 в 18:35:11

Рисунок 16 – Реализация запроса № 7 (модификация запроса)

Следует отметить, что модифицированный запрос дает тот же результат. Это означает, что более 2 товаров из топ-5 с самой высокой list\_price клиенты не покупали.

8. Вывести клиентов (ID, имя, фамилия, сфера деятельности) из сфер IT или Health, которые совершили не менее 3 подтвержденных заказов в период 2017-01-01 по 2017-03-01, и при этом их общий доход от этих заказов превышает 10 000 долларов. Разделить вывод на две группы (IT и Health) с помощью UNION.

Будем считать, что доход клиенту приносит разница между ценой продажи и стандартной стоимостью товара. В таблице `order_items` `item_list_price_at_sale` превышает `item_standard_cost_at_sale`, тогда доход клиента считается как `quantity * (item_list_price_at_sale - item_standard_cost)`.

The screenshot shows the DBeaver interface with the following details:

- SQL Editor:** Contains the SQL query for question 8. It uses UNION to combine two SELECT statements. The first part selects customers from the 'IT' category where they have made at least 3 approved orders between January 1, 2017, and March 1, 2017, and their total revenue exceeds \$10,000. The second part selects customers from the 'Health' category for the same criteria.
- Results View:** Displays the query results in a table format. The columns are `customer_id`, `first_name`, `last_name`, and `job_industry_category`. The data is as follows:

	<code>customer_id</code>	<code>first_name</code>	<code>last_name</code>	<code>job_industry_category</code>
1	173	Ebba	Hanselmann	Health
2	255	Keeley	Kruger	IT
3	394	Roanne	Cowthard	Health
4	424	Dennie	Eunson	Health
5	590	Ddene	Burleton	Health
6	607	Adelaida	Redmond	Health
7	723	Madelina	Marte	Health
8	799	Harland	Spilisy	IT

Рисунок 17 – Реализация запроса № 8

Полная версия запроса представлена ниже.

```

select
    customer.customer_id,
    customer.first_name,
    customer.last_name,
    customer.job_industry_category
from
    customer
inner join
    orders on customer.customer_id = orders.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
where
    customer.job_industry_category = 'IT'
    and orders.order_status = 'Approved'
    and orders.order_date between '2017-01-01' and '2017-03-01'
group by
    customer.customer_id,
    customer.first_name,
    customer.last_name,
    customer.job_industry_category
having
    count(distinct orders.order_id) >= 3
    and sum(order_items.quantity * (order_items.item_list_price_at_sale -
order_items.item_standard_cost_at_sale)) > 10000

union

select
    customer.customer_id,
    customer.first_name,
    customer.last_name,
    customer.job_industry_category
from
    customer
inner join
    orders on customer.customer_id = orders.customer_id
inner join
    order_items on orders.order_id = order_items.order_id
where
    customer.job_industry_category = 'Health'
    and orders.order_status = 'Approved'
    and orders.order_date between '2017-01-01' and '2017-03-01'
group by
    customer.customer_id,
    customer.first_name,
    customer.last_name,
    customer.job_industry_category
having
    count(distinct orders.order_id) >= 3
    and sum(order_items.quantity * (order_items.item_list_price_at_sale -
order_items.item_standard_cost_at_sale)) > 10000

order by
    customer_id;

```