

БУ ВО Ханты-Мансийского округа – Югры
«Сургутский государственный университет»
Политехнический институт
Кафедра информатики и вычислительной техники

Отчет
Лабораторная работа №3
Выборка данных

Проверил:
Гавриленко А.В
Выполнил: студент
группы 606-12
Речук Д.М

Сургут
2023 г

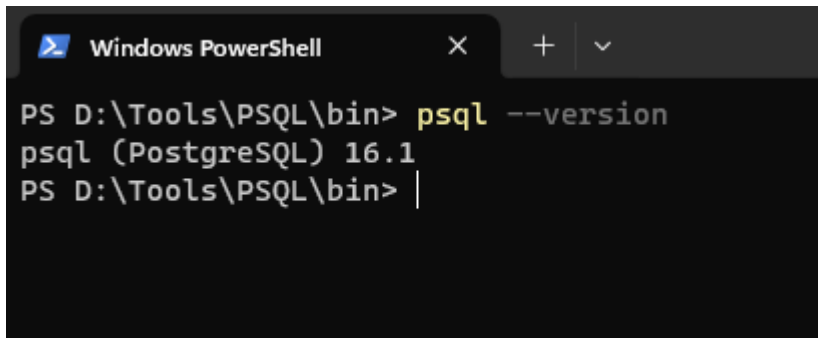
Задание 0. Для выполнения данной лабораторной работы требуется БД demo-small-20161013. Скачать БД можно по ссылке ниже.

<https://postgrespro.ru/docs/postgrespro/10/demodb-bookings-installation.html>

После скачивания, распаковать архив.

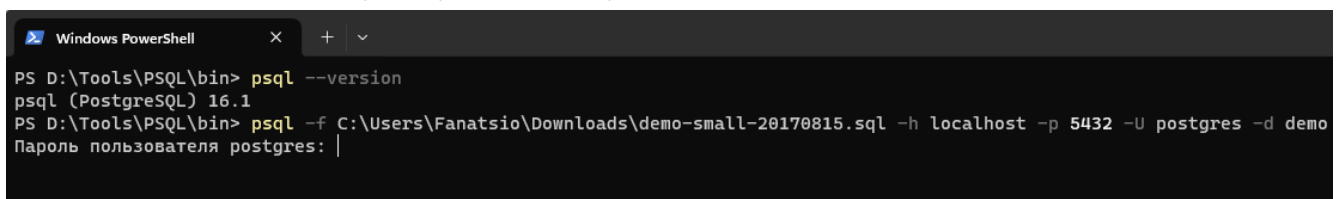
Далее, открываем терминал. Переходим в каталог, куда мы скачали PostgreSQL.

Для подключения базы данных открываем терминал, переходим в каталог, куда мы скачали PostgreSQL командой `cd` и заходим в `bin`.

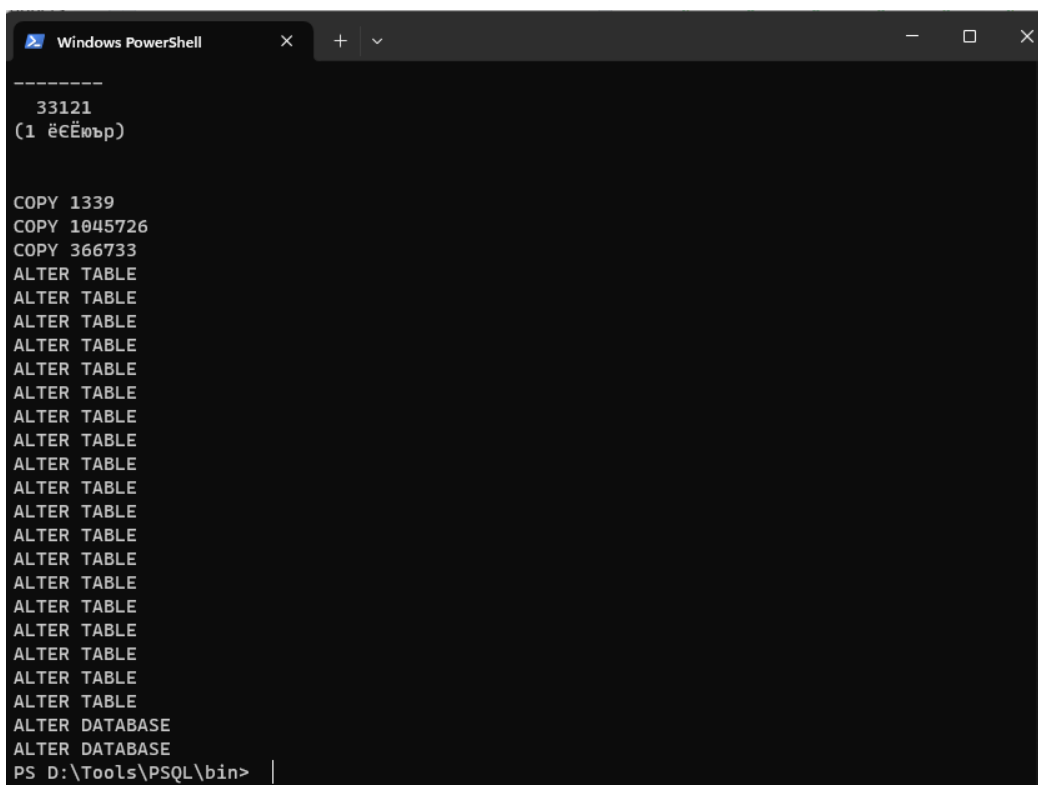


```
Windows PowerShell
PS D:\Tools\PSQL\bin> psql --version
psql (PostgreSQL) 16.1
PS D:\Tools\PSQL\bin> |
```

Далее выполняем следующую команду
























```
Windows PowerShell
PS D:\Tools\PSQL\bin> psql --version
psql (PostgreSQL) 16.1
PS D:\Tools\PSQL\bin> psql -f C:\Users\Fanatsio\Downloads\demo-small-20170815.sql -h localhost -p 5432 -U postgres -d demo
Пароль пользователя postgres: |
```



```
Windows PowerShell
-----
33121
(1 ёсёёър)

COPY 1339
COPY 1045726
COPY 366733
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER DATABASE
ALTER DATABASE
PS D:\Tools\PSQL\bin> |
```

- ✓  bookings
 - >  Aggregates
 - >  Collations
 - >  Domains
 - >  FTS Configurations
 - >  FTS Dictionaries
 - > **Aa** FTS Parsers
 - >  FTS Templates
 - >  Foreign Tables
 - >  Functions
 - >  Materialized Views
 - >  Operators
 - >  Procedures
 - > **1..3** Sequences
 - ✓  Tables (8)
 - >  aircrafts_data
 - >  airports_data
 - >  boarding_passes
 - >  bookings
 - >  flights
 - >  seats
 - >  ticket_flights
 - >  tickets

Задание 1. Получите список аэропортов с указанием их кода и города из таблицы airports_data БД demo.

demo/postgres@PostgreSQL 16

Query Query History

```
1 SELECT airport_code, airport_name, city FROM airports_data;
```

Data Output Messages Notifications

	airport_code [PK] character	airport_name jsonb	city jsonb
1	YKS	{ "en": "Yakutsk Airport", "ru": "Якутск" }	{ "en": "Yakutsk", "ru": "Якутск" }
2	MJZ	{ "en": "Mirny Airport", "ru": "Мирный" }	{ "en": "Mirnyj", "ru": "Мирный" }
3	KHV	{ "en": "Khabarovsk-Novy Airport", "ru": "Хабаровск-Новый" }	{ "en": "Khabarovsk", "ru": "Хабаровск" }
4	PKC	{ "en": "Yelizovo Airport", "ru": "Елизово" }	{ "en": "Petrovsk", "ru": "Петропавловск-Камчатский" }
5	UUS	{ "en": "Yuzhno-Sakhalinsk Airport", "ru": "Хомутово" }	{ "en": "Yuzhno-Sakhalinsk", "ru": "Южно-Сахалинск" }
6	VVO	{ "en": "Vladivostok International Airport", "ru": "Владивосток" }	{ "en": "Vladivostok", "ru": "Владивосток" }
7	LED	{ "en": "Pulkovo Airport", "ru": "Пулково" }	{ "en": "St. Petersburg", "ru": "Санкт-Петербург" }
8	KGD	{ "en": "Khrabrovo Airport", "ru": "Храброво" }	{ "en": "Kaliningrad", "ru": "Калининград" }
9	KEJ	{ "en": "Kemerovo Airport", "ru": "Кемерово" }	{ "en": "Kemerovo", "ru": "Кемерово" }

Total rows: 104 of 104 Query complete 00:00:00.185 Ln 1,

Задание 2. Получите список мест с указанием числа места (первые 2 символа), сектора (3 символ) места и класса (бизнес, эконо, комфорт), а также его идентификационного номера. Список должен быть упорядочен по номеру места.

The screenshot shows a PostgreSQL query editor interface. The top bar indicates the connection is to 'demo/postgres@PostgreSQL 16'. Below the toolbar, the 'Query' tab is active, displaying the following SQL query:

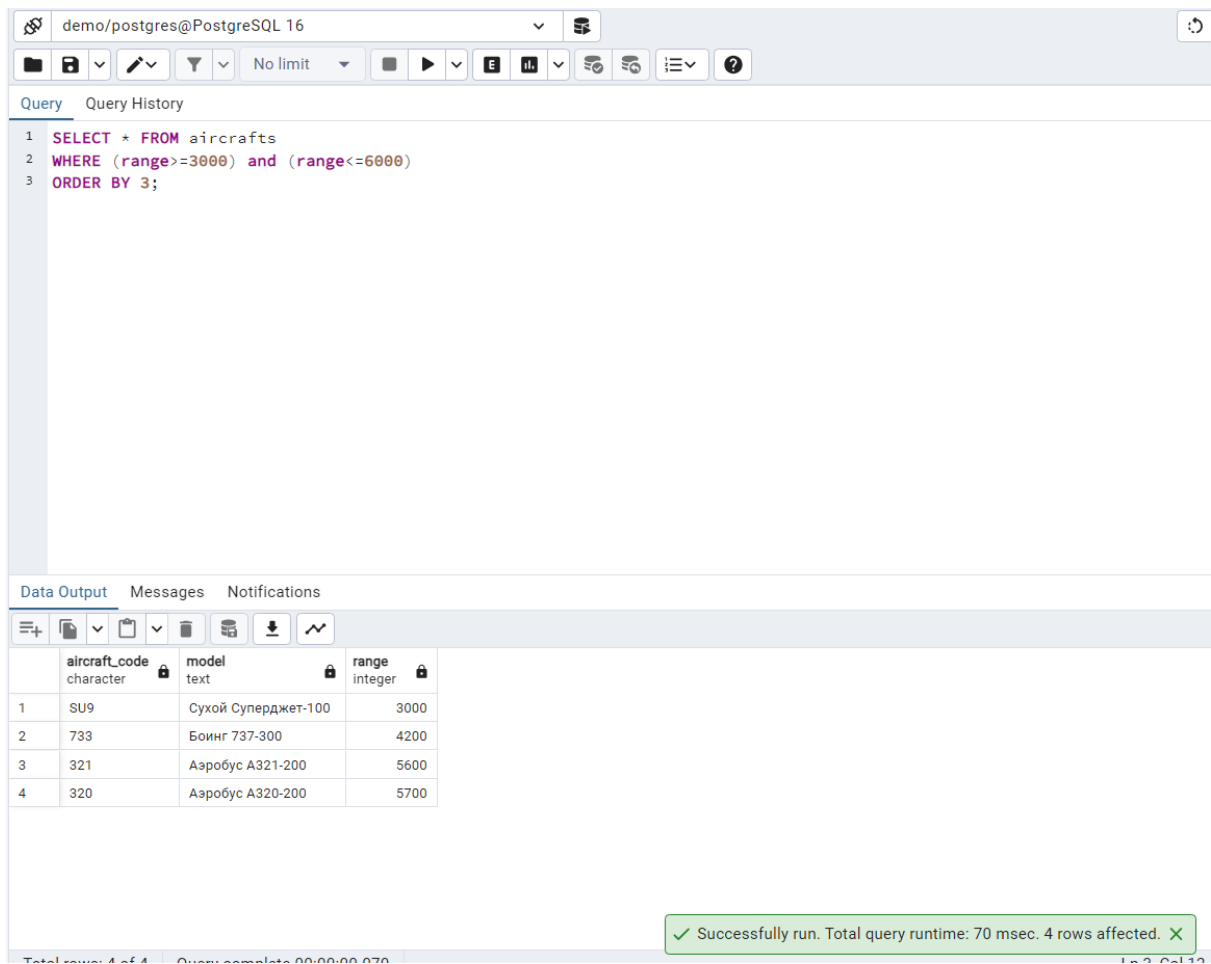
```
1 SELECT (SUBSTRING (seat_no,1,2) || ' ' || SUBSTRING (seat_no,3,1) || ' ' || fare_conditions )
2 AS seats_class, aircraft_code FROM seats
3 ORDER BY seat_no;
```

The 'Data Output' tab is also visible, showing the results of the query. The results are displayed in a table with two columns: 'seats_class' (text) and 'aircraft_code' (character). The table contains 9 rows of data, representing the first 9 seats from the 'seats' table.

	seats_class	aircraft_code
1	10 A Economy	321
2	10 A Economy	319
3	10 A Economy	320
4	10 A Economy	SU9
5	10 B Economy	321
6	10 B Economy	320
7	10 B Economy	319
8	10 B Economy	733
9	10 C Economy	SU9

At the bottom of the interface, a status bar indicates 'Total rows: 1000 of 1339' and 'Query complete 00:00:00.085'. A green notification box on the right states: 'Successfully run. Total query runtime: 85 msec. 1339 rows affected. X'.

Задание 3. Получите список самолётов, дальность полёта которых находится в диапазоне от 3000 км до 6000 км, отсортировав его по дальности



The screenshot shows a PostgreSQL client interface with the following components:

- Header:** Connection name 'demo/postgres@PostgreSQL 16' and a refresh button.
- Toolbar:** Icons for file operations, filters, and execution.
- Query Editor:** Contains the SQL query:

```
1 SELECT * FROM aircrafts
2 WHERE (range>=3000) and (range<=6000)
3 ORDER BY 3;
```
- Data Output:** A table with 4 rows and 4 columns: aircraft_code, model, range, and an unlabeled column. The data is as follows:

	aircraft_code	model	range	
1	SU9	Сухой Суперджет-100	3000	
2	733	Боинг 737-300	4200	
3	321	Аэробус А321-200	5600	
4	320	Аэробус А320-200	5700	
- Status Bar:** Shows 'Total rows: 4 of 4' and 'Query complete 00:00:00.070'.
- Message Box:** A green box at the bottom right states: '✓ Successfully run. Total query runtime: 70 msec. 4 rows affected. ✕'

demo/postgres@PostgreSQL 16

No limit

Query Query History

```
1 SELECT * FROM aircrafts
2 WHERE range BETWEEN 3000 and 6000
3 ORDER BY 3;
```

Data Output Messages Notifications

	aircraft_code character	model text	range integer
1	SU9	Сухой Суперджет-100	3000
2	733	Боинг 737-300	4200
3	321	Аэробус А321-200	5600
4	320	Аэробус А320-200	5700

✓ Successfully run. Total query runtime: 47 msec. 4 rows affected. ✕

Задание 4. Выведите все кодировки самолётов, их модели и дальность полёта, кроме самолётов модели Аэробус и Боинг.

demo/postgres@PostgreSQL 16

No limit

Query

Query History

```
1 SELECT * FROM aircrafts
2 WHERE model NOT LIKE 'Аэробус%'
3 AND model NOT LIKE 'Боинг%';
```

Data Output

Messages

Notifications

	aircraft_code character	model text	range integer
1	SU9	Сухой Суперджет-100	3000
2	CN1	Сессна 208 Караван	1200
3	CR2	Бомбардье CRJ-200	2700

Total rows: 3 of 3

Query complete 00:00:00.056

Ln 3, Col 2

✓ Successfully run. Total query runtime: 56 msec. 3 rows affected. ✕

Задание 5. Получите список самолётов компаний модели Аэробус или Боинг

demo/postgres@PostgreSQL 16

No limit

Query

Query History

```
1 SELECT * FROM aircrafts WHERE model ~ '^(A|Бои)';
```

Data Output

Messages

Notifications

	aircraft_code character	model text	range integer
1	773	Боинг 777-300	11100
2	763	Боинг 767-300	7900
3	320	Аэробус A320-200	5700
4	321	Аэробус A321-200	5600
5	319	Аэробус A319-100	6700
6	733	Боинг 737-300	4200

✓ Successfully run. Total query runtime: 63 msec. 6 rows affected.

✕

Total rows: 6 of 6

Query complete 00:00:00.063

Ln 1, Col 5

Задание 6. Получите список рейсов, у которых не указан ближайший ВЫЛЕТ.

QueryQuery History

1 SELECT * FROM flights WHERE actual_departure ISNULL;

Data OutputMessagesNotifications

	flight_id [PK] integer	flight_no character	scheduled_departure timestamp with time zone	scheduled_arrival timestamp with time zone	departure_airport character	arrival_lairport character	status character varying (20)	aircraft_code character
1	1185	PG0134	2017-09-10 11:50:00+05	2017-09-10 16:55:00+05	DME	BTK	Scheduled	319
2	3979	PG0052	2017-08-25 16:50:00+05	2017-08-25 19:35:00+05	VKO	HMA	Scheduled	CR2
3	4739	PG0561	2017-09-05 14:30:00+05	2017-09-05 16:15:00+05	VKO	AER	Scheduled	763
4	5502	PG0529	2017-09-12 11:50:00+05	2017-09-12 13:20:00+05	SVO	UFA	Scheduled	763
5	6938	PG0461	2017-09-04 14:25:00+05	2017-09-04 15:20:00+05	SVO	ULV	Scheduled	SU9
6	7784	PG0667	2017-09-10 17:00:00+05	2017-09-10 19:30:00+05	SVO	KRO	Scheduled	CR2
7	9478	PG0360	2017-08-28 11:00:00+05	2017-08-28 13:35:00+05	LED	REN	Scheduled	CR2
8	11085	PG0569	2017-08-24 17:05:00+05	2017-08-24 18:10:00+05	SVX	SCW	Scheduled	733
...

✓ Successfully run. Total query runtime: 157 msec. 16348 rows affected. ✕

Total rows: 1000 of 16348Query complete 00:00:00.157Ln 1, Col 53

Самостоятельная работа

1. Запрос на полную выборку данных.

```
1 SELECT * FROM provider;
```

Data Output Messages Notifications

	provider_id [PK] integer	inn integer	provider_name character varying (255)	address character varying (255)
1	1	[null]	provider 0	[null]
2	2	[null]	provider 1	[null]
3	3	[null]	provider 2	[null]
4	4	[null]	provider 3	[null]
5	5	[null]	provider 4	[null]
6	6	[null]	provider 5	[null]
7	7	[null]	provider 6	[null]
8	8	[null]	provider 7	[null]
9	9	[null]	provider 8	[null]

✓ Successfully run. Total query runtime: 84 msec. 749 rows affected. ✕

2. Запрос на выборку данных без повторений.

query query history

```
1 SELECT DISTINCT inn, provider_name, address
2 FROM provider;
3
```

Data Output Messages Notifications

	inn integer	provider_name character varying (255)	address character varying (255)
1	[null]	provider 212	[null]
2	[null]	provider 337	[null]
3	[null]	provider 640	[null]
4	[null]	provider 550	[null]

3. Запрос на выборку первых 10 записей.

query query history

```
1 SELECT inn, provider_name, address
2 FROM provider
3 LIMIT 10;
4
```

Data Output Messages Notifications

	inn integer	provider_name character varying (255)	address character varying (255)
1	[null]	provider 0	[null]
2	[null]	provider 1	[null]
3	[null]	provider 2	[null]
4	[null]	provider 3	[null]
5	[null]	provider 4	[null]

4. Запрос на выборку последних 15 записей.

Query

Query History

1

2

3

4

5

SELECT *
FROM provider
ORDER BY provider_id DESC
LIMIT 15;

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	provider_id [PK] integer	inn integer	provider_name character varying (255)	address character varying (255)
1	749	[null]	provider 748	[null]
2	748	[null]	provider 747	[null]
3	747	[null]	provider 746	[null]
4	746	[null]	provider 745	[null]

5. Запросы на выполнение функций Average, Max, Min.

1

2

3

4

5

6

7

8

9

10

11

12

-- Пример для столбца price в таблице product
SELECT AVG(price) AS average_price
FROM product;

-- Пример для столбца quantity в таблице product
SELECT MAX(quantity) AS max_quantity
FROM product;

-- Пример для столбца weight в таблице product
SELECT MIN(weight) AS min_weight
FROM product;

6. Сконструируйте запросы с использованием оператора Where: - запрос на возвращение определенного кортежа по первичному ключу; - запросы на возвращение значения по условиям больше, меньше и между; - запросы на возвращении всех кортежей по условию с использованием оператора LIKE и ESCAPE; - запрос на возвращение кортежей со сложным условием на основе логических операторов И, ИЛИ, НЕ, EXISTS; - запрос с использованием оператора NOT NULL в условии отбора.

Query	Query History
1	-- Пример для таблицы natural_person
2	SELECT *
3	FROM natural_person
4	WHERE natural_person_id = 1;
5	
6	-- Пример для таблицы product, выбор продуктов с количеством больше 100
7	SELECT *
8	FROM product
9	WHERE quantity > 100;
10	
11	-- Пример для таблицы product, выбор продуктов с ценой меньше 50
12	SELECT *
13	FROM product
14	WHERE price < 50;
15	
16	-- Пример для таблицы product, выбор продуктов с весом между 5 и 10
17	SELECT *
18	FROM product
19	WHERE weight BETWEEN 5 AND 10;
20	
21	-- Пример для таблицы natural_person, выбор физических лиц с именем, начинающимся на "Иван"
22	SELECT *
23	FROM natural_person
24	WHERE natural_customer_name LIKE 'Иван%';
25	
26	-- Пример с использованием ESCAPE для поиска значений, содержащих символ "%"
27	SELECT *
28	FROM table_name
29	WHERE column_name LIKE '%\%%' ESCAPE '\';
30	

Query	Query History
1	-- Пример для таблицы orders, выбор заказов, сделанных физическими лицами и сумма больше 100
2	SELECT *
3	FROM orders
4	WHERE category_customer = 'Физ. лицо' AND total_cost > 100;
5	
6	-- Пример для таблицы product, выбор продуктов, у которых есть заказы
7	SELECT *
8	FROM product
9	WHERE EXISTS (SELECT 1 FROM orders WHERE orders.product_id = product.product_id);
10	
11	-- Пример для таблицы natural_person, выбор физических лиц с указанным номером телефона
12	SELECT *
13	FROM natural_person
14	WHERE phone_number IS NOT NULL;
15	

7. Запрос с простыми условиями, условиями, содержащими IN или BETWEEN.

```
1  -- С использованием IN
2  SELECT *
3  FROM product
4  WHERE price IN (50, 75, 100);
5
6  -- С использованием BETWEEN
7  SELECT *
8  FROM product
9  WHERE price BETWEEN 50 AND 100;
10
```

8. Запросы с сортировкой по нескольким полям, направлениям.

```
1  -- Сортировка по нескольким полям (product_name в алфавитном порядке, price по убыванию)
2  SELECT *
3  FROM product
4  ORDER BY product_name ASC, price DESC;
5
```

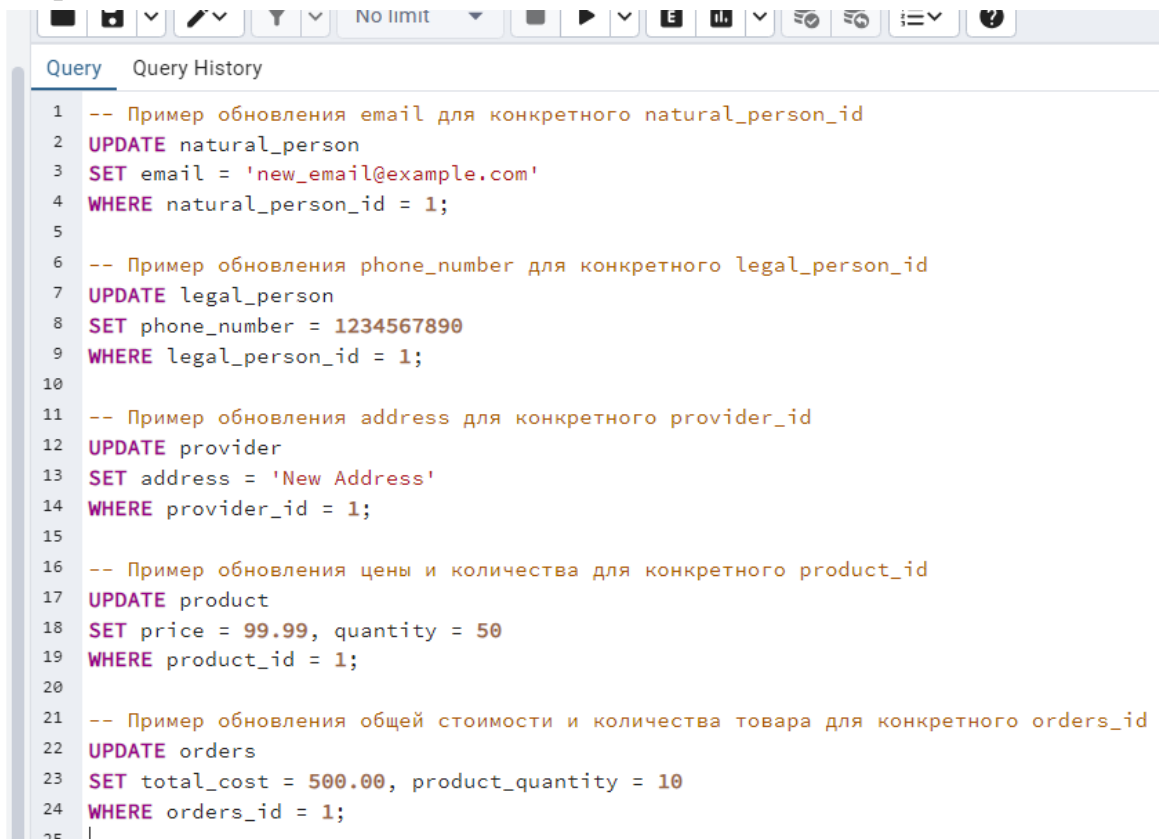
9. Запросы с использованием групповых операций (группировка статистические функции, отбор по групповым функциям).

```
1  -- С использованием статических функций
2  SELECT customer_id, AVG(total_cost) AS avg_cost, SUM(total_cost) AS total_cost
3  FROM orders
4  GROUP BY customer_id;
5
6  -- С использованием групповых функций
7  SELECT customer_id, SUM(total_cost) AS total_cost
8  FROM orders
9  GROUP BY customer_id
10 HAVING SUM(total_cost) > 1000;
11
```


10. Запросы с операцией над множествами (обязательно используя сортировку).

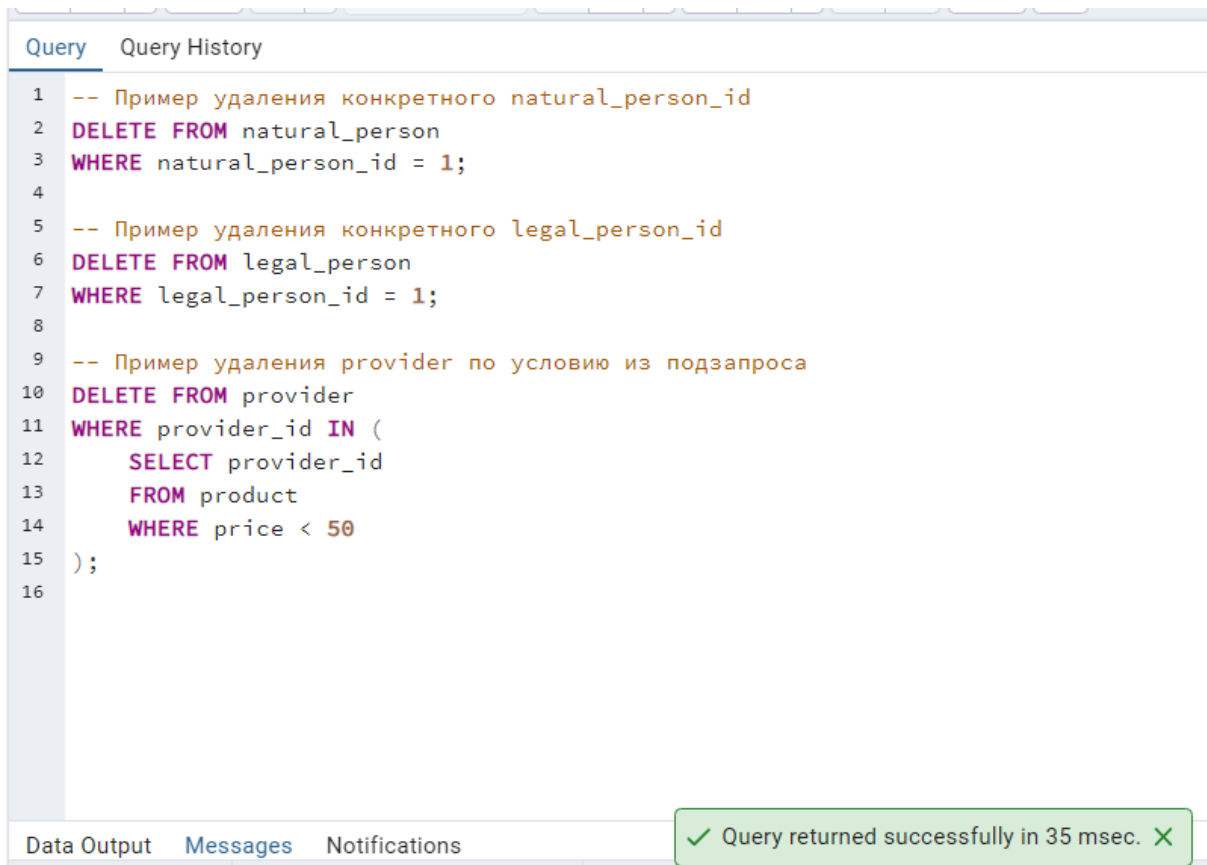
```
1  -- Использование UNION (объединение множеств)
2  SELECT email
3  FROM natural_person
4  UNION
5  SELECT email
6  FROM legal_person
7  ORDER BY email ASC;
8
9  -- Использование UNION ALL (объединение множеств с дубликатами)
10 SELECT email
11 FROM natural_person
12 UNION ALL
13 SELECT email
14 FROM legal_person
15 ORDER BY email ASC;
16 |
```

11. Запросы на обновление.

A screenshot of a SQL query editor interface. The top toolbar contains various icons for query execution and management. Below the toolbar, there are two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a series of SQL UPDATE statements. The statements are numbered 1 through 25. Statements 1-5 update the email of a natural person. Statements 6-9 update the phone number of a legal person. Statements 11-14 update the address of a provider. Statements 16-19 update the price and quantity of a product. Statements 21-24 update the total cost and product quantity of an order. Statement 25 is a partial line.

```
1  -- Пример обновления email для конкретного natural_person_id
2  UPDATE natural_person
3  SET email = 'new_email@example.com'
4  WHERE natural_person_id = 1;
5
6  -- Пример обновления phone_number для конкретного legal_person_id
7  UPDATE legal_person
8  SET phone_number = 1234567890
9  WHERE legal_person_id = 1;
10
11 -- Пример обновления address для конкретного provider_id
12 UPDATE provider
13 SET address = 'New Address'
14 WHERE provider_id = 1;
15
16 -- Пример обновления цены и количества для конкретного product_id
17 UPDATE product
18 SET price = 99.99, quantity = 50
19 WHERE product_id = 1;
20
21 -- Пример обновления общей стоимости и количества товара для конкретного orders_id
22 UPDATE orders
23 SET total_cost = 500.00, product_quantity = 10
24 WHERE orders_id = 1;
25 |
```

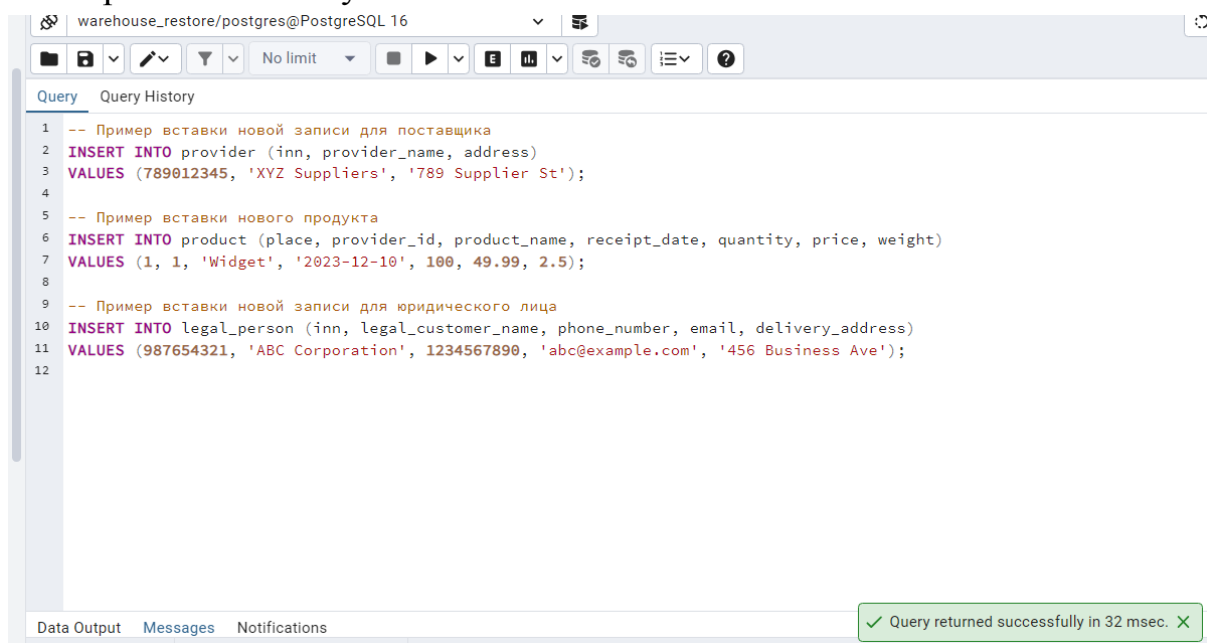
12. Запросы на удаление.



```
1 -- Пример удаления конкретного natural_person_id
2 DELETE FROM natural_person
3 WHERE natural_person_id = 1;
4
5 -- Пример удаления конкретного legal_person_id
6 DELETE FROM legal_person
7 WHERE legal_person_id = 1;
8
9 -- Пример удаления provider по условию из подзапроса
10 DELETE FROM provider
11 WHERE provider_id IN (
12     SELECT provider_id
13     FROM product
14     WHERE price < 50
15 );
16
```

Query returned successfully in 35 msec.

13. Запросы на вставку.



```
1 -- Пример вставки новой записи для поставщика
2 INSERT INTO provider (inn, provider_name, address)
3 VALUES (789012345, 'XYZ Suppliers', '789 Supplier St');
4
5 -- Пример вставки нового продукта
6 INSERT INTO product (place, provider_id, product_name, receipt_date, quantity, price, weight)
7 VALUES (1, 1, 'Widget', '2023-12-10', 100, 49.99, 2.5);
8
9 -- Пример вставки новой записи для юридического лица
10 INSERT INTO legal_person (inn, legal_customer_name, phone_number, email, delivery_address)
11 VALUES (987654321, 'ABC Corporation', 1234567890, 'abc@example.com', '456 Business Ave');
12
```

Query returned successfully in 32 msec.

14. Используя таблицу с персональными данными из своей БД или demo БД в PostgreSQL отобразите список сотрудников/персон (указав их Фамилию И. в одной колонке), которые в следующем месяце будут отмечать юбилей, с указанием возраста, даты рождения, даты юбилея. Заголовки должны соответствовать шаблону вывода данных.

По причине отсутствия информации о дате рождения клиентов, создадим таблицу “Client”

```
1  -- Создание таблицы
2  CREATE TABLE Client (
3      client_id SERIAL PRIMARY KEY,
4      second_name VARCHAR(255) NOT NULL,
5      name VARCHAR(255) NOT NULL,
6      date_of_birth DATE NOT NULL
7  );
```

Далее заполним её разнообразными данными

Query	Query History
1	-- Заполнение данными
2	INSERT INTO Client (second_name, name, date_of_birth) VALUES
3	('Иванов', 'Иван', '1955-12-19'),
4	('Петров', 'Петр', '1956-12-23'),
5	('Сидоров', 'Николай', '1957-12-09'),
6	('Кузнецов', 'Сергей', '1958-12-02'),
7	('Смирнов', 'Алексей', '1962-12-02'),
8	('Васильев', 'Василий', '1964-12-28'),
9	('Павлов', 'Павел', '1965-12-17'),
10	('Семенов', 'Семен', '1966-12-14'),
11	('Голубев', 'Глеб', '1967-12-11'),
12	('Виноградов', 'Виктор', '1968-12-24'),
13	('Богданов', 'Борис', '1970-12-05'),
14	('Воробьев', 'Валерий', '1971-12-21'),
15	('Федоров', 'Федор', '1972-12-18'),
16	('Михайлов', 'Михаил', '1973-12-07'),

Затем отобразим список сотрудников/персон (указав их Фамилию И. в одной колонке), которые в следующем месяце будут отмечать юбилей, с указанием возраста, даты рождения, даты юбилея

```
1  SELECT
2      CONCAT("second_name", ' ', LEFT("name", 1), '.') AS "ФИО",
3      DATE_PART('year', AGE(NOW(), "date_of_birth")) AS "Возраст",
4      "date_of_birth" AS "Дата рождения",
5      "date_of_birth" + INTERVAL '1 year' * DATE_PART('year', AGE(NOW(), "date_of_birth")) AS "Дата юбилея"
6  FROM Client
7  WHERE
8      DATE_PART('month', "date_of_birth") = DATE_PART('month', NOW() + INTERVAL '1 MONTH')
9      AND DATE_PART('year', AGE(NOW(), "date_of_birth"))::integer % 5 = 4;
```