

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«Национальный исследовательский университет ИТМО»  
(Университет ИТМО)**

**Факультет прикладной информатики**

**Образовательная программа Мобильные и сетевые технологии**

**Направление подготовки 09.03.03 Мобильные и сетевые технологии**

**О Т Ч Е Т  
ЛАБОРАТОРНАЯ РАБОТА №4**

**"ЗАПРОСЫ НА ВЫБОРКУ И МОДИФИКАЦИЮ ДАННЫХ.  
ПРЕДСТАВЛЕНИЯ. РАБОТА С ИНДЕКСАМИ"**

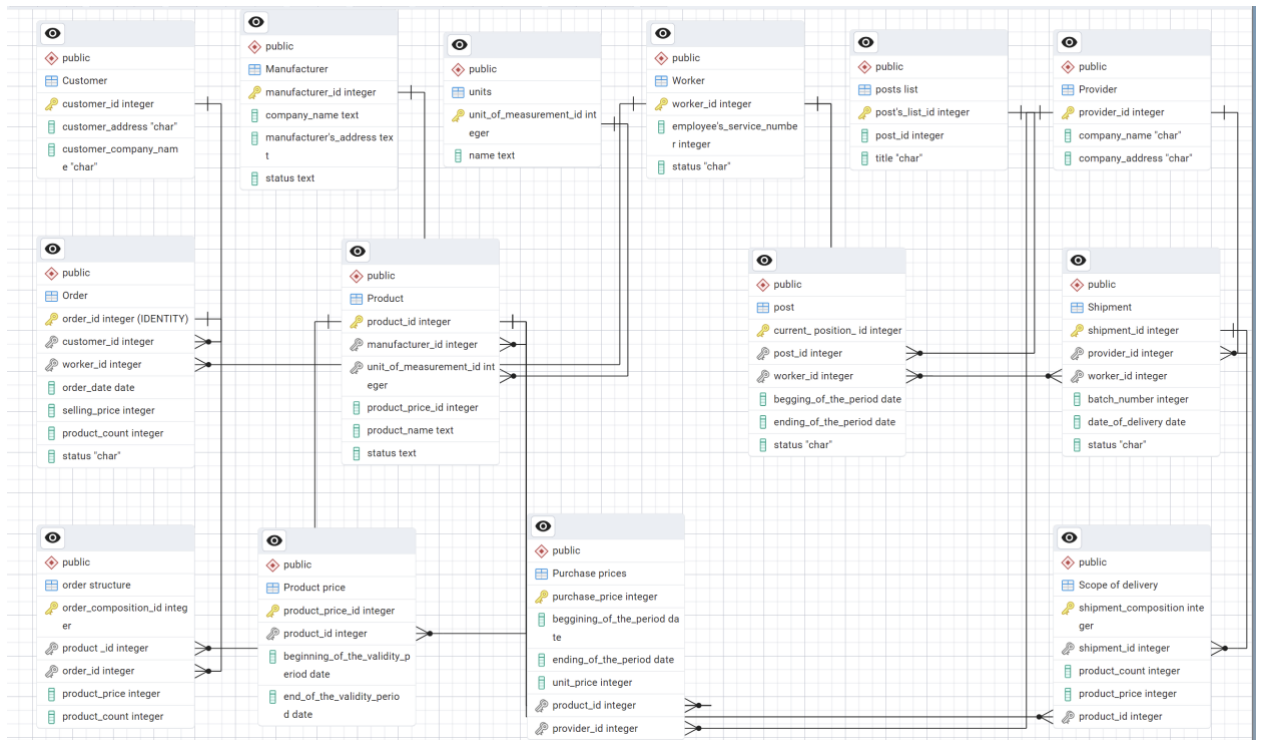
**Обучающийся:** Макаров Егор 3240

**Преподаватель:** Говорова Марина Михайловна

Санкт-  
Петербург,  
2025

**1. Цель работы:** Овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

## 2. Схема базы данных (ЛР 3).



## 3. Выполнение:

### 3.1 Запросы к базе данных.

В рамках выполнения лабораторной работы были составлены и выполнены SQL-запросы в соответствии с индивидуальным заданием (часть 2). Каждый запрос формировался исходя из требований задания, отражающих конкретные задачи по выборке данных из базы.

## Запрос 1: список всех товаров с ценой, производителем и единицей измерения товаров с ценой, производителем

Запрос История запросов

```
1 SELECT
2     "Product"."product_name",
3     "Product price"."beginning_of_the_validity_period",
4     "Product price"."end_of_the_validity_period",
5     "Manufacturer"."company_name" AS "manufacturer",
6     "units"."name" AS "unit"
7 FROM
8     "Product"
9 JOIN
10    "Product price" ON "Product"."product_price_id" = "Product price"."product_price_id"
11 JOIN
12    "Manufacturer" ON "Product"."manufacturer_id" = "Manufacturer"."manufacturer_id"
13 JOIN
14    "units" ON "Product"."unit_of_measurement_id" = "units"."unit_of_measurement_id";
```

Data Output Сообщения Notifications

Showing row

	product_name text	beginning_of_the_validity_period date	end_of_the_validity_period date	manufacturer text	unit text
1	Product 1	2025-01-01	2025-12-31	Company A	шт
2	Product 2	2025-01-01	2025-12-31	Company B	кг

## Запрос 2: Заказы, где количество товара больше 1

Запрос История запросов

```
1 SELECT
2     "Order"."order_id", -- ID заказа
3     "Customer"."customer_company_name", -- название компании заказчика
4     "Order"."order_date", -- дата заказа
5     "Order"."product_count" -- общее количество товаров в заказе
6 FROM
7     "Order"
8 JOIN
9     "Customer" ON "Order"."customer_id" = "Customer"."customer_id"
10 -- соединяем заказ с покупателем
11 WHERE
12     "Order"."product_count" > 1; -- фильтруем по количеству товара
```

Data Output Сообщения Notifications

Showing

	order_id integer	customer_company_name 'char'	order_date date	product_count integer
1	1	\320	2025-04-04	5
2	2	\320	2025-04-05	3

### Запрос 3: Количество поставок по каждому поставщику

Запрос История запросов

```
1 SELECT
2     "Provider"."company_name",          -- название поставщика
3     COUNT("Shipment"."shipment_id") AS "shipment_count" -- количество поставок
4 FROM
5     "Provider"
6 JOIN
7     "Shipment" ON "Provider"."provider_id" = "Shipment"."provider_id"
8     -- связываем поставщика с поставками
9 GROUP BY
10    "Provider"."company_name";          -- группируем по имени поставщика
11
```

Data Output Сообщения Notifications

	company_name char	shipment_count bigint
1	\320	2

### Запрос 4: Товары с закупочной ценой выше 100

Запрос История запросов

```
1 SELECT
2     "Product"."product_name",          -- название товара
3     "Purchase prices"."unit_price"      -- цена закупки
4 FROM
5     "Product"
6 JOIN
7     "Purchase prices" ON "Product"."product_id" = "Purchase prices"."product_id"
8     -- соединяем товары с таблицей закупочных цен
9 WHERE
10    "Purchase prices"."unit_price" > 100; -- фильтр по цене
11
```

Data Output Сообщения Notifications

	product_name text	unit_price integer
1	Product 2	150

### Запрос 5: Активные сотрудники и их должности

ЗапросИстория запросов

1

SELECT

"Worker"."worker\_id",  
"Worker"."employee's\_service\_number",  
"posts list"."title" AS "position\_title",  
"post"."begging\_of\_the\_period",  
"post"."ending\_of\_the\_period"

-- ID сотрудника  
-- табельный номер  
-- название должности  
-- начало периода  
-- конец периода

7

FROM

"Worker"

9

JOIN

"post" ON "Worker"."worker\_id" = "post"."worker\_id"

-- соединяем сотрудника с его текущей должностью

12

JOIN

"posts list" ON "post"."post\_id" = "posts list"."post\_id"

-- получаем название должности из справочника

15

WHERE

"post"."ending\_of\_the\_period" IS NULL  
OR "post"."ending\_of\_the\_period" > CURRENT\_DATE;

-- фильтруем только активные должности

Data Output

Сообщения

Notifications

SQL

Showing

	worker_id integer	employee's_service_number integer	position_title "char"	begging_of_the_period date	ending_of_the_period date
1	1	12345	\320	2025-04-01	2025-12-31
2	2	67890	\320	2025-04-01	2025-12-31

## 3.2 Представления

### Представление 1: Информация о товарах и производителях

ЗапросИстория запросов

1

CREATE VIEW "product\_info\_view" AS

SELECT

"Product"."product\_name",  
"Product price"."beginning\_of\_the\_validity\_period",  
"Product price"."end\_of\_the\_validity\_period",  
"Manufacturer"."company\_name" AS "manufacturer",  
"units"."name" AS "unit"

-- название товара  
-- начало действия цены  
-- конец действия цены  
-- производитель  
-- единица измерения

8

FROM

"Product"

10

JOIN

"Product price" ON "Product"."product\_price\_id" = "Product price"."product\_price\_id"

12

JOIN

"Manufacturer" ON "Product"."manufacturer\_id" = "Manufacturer"."manufacturer\_id"

14

JOIN

"units" ON "Product"."unit\_of\_measurement\_id" = "units"."unit\_of\_measurement\_id";

Data Output

Сообщения

Notifications

CREATE VIEW

Запрос завершен успешно, время выполнения: 132 мсек.

## Представление 2: Крупные заказы

Запрос История запросов

```
1 CREATE VIEW "big_orders_view" AS
2 SELECT
3     "Order"."order_id",
4     "Customer"."customer_company_name",
5     "Order"."order_date",
6     "Order"."product_count"
7 FROM
8     "Order"
9 JOIN
10    "Customer" ON "Order"."customer_id" = "Customer"."customer_id"
11 WHERE
12    "Order"."product_count" > 10;
```

Data Output Сообщения Notifications

ERROR: relation "big\_orders\_view" already exists

SQL-состояние: 42P07

## Представление 3: Активные сотрудники и их должности

Запрос История запросов

```
1 CREATE VIEW "active_employees_view" AS
2 SELECT
3     "Worker"."worker_id",
4     "Worker"."employee's_service_number",
5     "posts list"."title" AS "position_title",
6     "post"."beginning_of_the_period",
7     "post"."ending_of_the_period"
8 FROM
9     "Worker"
10 JOIN
11    "post" ON "Worker"."worker_id" = "post"."worker_id"
12 JOIN
13    "posts list" ON "post"."post_id" = "posts list"."post_id"
14 WHERE
15    "post"."ending_of_the_period" IS NULL
16    OR "post"."ending_of_the_period" > CURRENT_DATE;
```

Data Output Сообщения Notifications

CREATE VIEW

Запрос завершён успешно, время выполнения: 172 msec.

## Представление 4: Количество поставок по каждому поставщику

```
Запрос История запросов
1 CREATE VIEW "provider_shipments_view" AS
2 SELECT
3     "Provider"."company_name", -- название поставщика
4     COUNT("Shipment"."shipment_id") AS "shipment_count" -- общее количество поставок
5 FROM
6     "Provider"
7 JOIN
8     "Shipment" ON "Provider"."provider_id" = "Shipment"."provider_id"
9 GROUP BY
10     "Provider"."company_name";
11
```

Data Output Сообщения Notifications

CREATE VIEW

Запрос завершён успешно, время выполнения: 99 msec.

## Представление 5: Товары с закупочной ценой выше 100

```
Запрос История запросов
1 CREATE VIEW "expensive_purchases_view" AS
2 SELECT
3     "Product"."product_name", -- название товара
4     "Purchase prices"."unit_price" -- закупочная цена
5 FROM
6     "Product"
7 JOIN
8     "Purchase prices" ON "Product"."product_id" = "Purchase prices"."product_id"
9 WHERE
10     "Purchase prices"."unit_price" > 100;
11
```

Data Output Сообщения Notifications

CREATE VIEW

Запрос завершён успешно, время выполнения: 95 msec.

## 3.3 Запросы на модификацию данных

Выполнение запросов на модификацию данных (INSERT, UPDATE, DELETE) с подзапросами

В ходе работы были составлены и успешно выполнены три сложных запроса на модификацию данных:

1. INSERT — добавление нового заказа для клиента, найденного по названию компании с помощью подзапроса.
2. UPDATE — обновление статуса самого последнего заказа клиента на «completed», используя вложенные подзапросы для выбора нужного заказа.
3. DELETE — удаление поставки с минимальной ценой у заданного поставщика, с использованием подзапросов для поиска нужной записи.

Задание: Вставить новый заказ для клиента, выбрав клиента по названию компании. Пример INSERT с подзапросом

```
Запрос  История запросов
1  INSERT INTO "Order" (
2      "customer_id",
3      "worker_id",
4      "order_date",
5      "selling_price",
6      "product_count",
7      "status"
8  )
9  VALUES (
10     (SELECT "customer_id"
11        FROM "Customer"
12       WHERE "customer_company_name" = 'ООО Пример'
13       ORDER BY "customer_id"
14       LIMIT 1),
15     1,
16     CURRENT_DATE,
17     15000,
18     25,
19     'active'
20 );
21
22
```

Data Output Сообщения Notifications

INSERT 0 1

Запрос завершён успешно, время выполнения: 126 мсек.

Задание: обновить статус заказа на 'completed' для самого последнего заказа клиента 'ООО Пример'.

Запрос UPDATE с подзапросом:

```
Запрос  История запросов
1  UPDATE "Order"
2  SET "status" = 'completed'          -- меняем статус на 'completed'
3  WHERE "order_id" = (
4      SELECT "order_id"
5      FROM "Order"
6      WHERE "customer_id" = (
7          SELECT "customer_id"
8          FROM "Customer"
9          WHERE "customer_company_name" = 'ООО Пример'
10         ORDER BY "customer_id"
11         LIMIT 1
12     )
13     ORDER BY "order_date" DESC      -- берём самый последний заказ
14     LIMIT 1
15 );
16
```

Data Output Сообщения Notifications

UPDATE 1

Запрос завершён успешно, время выполнения: 106 мсек.



Запрос История запросов

```

1  -- Вывести заказы клиента "ООО Пример"
2  SELECT * FROM "Order"
3  WHERE "customer_id" = (
4      SELECT "customer_id"
5      FROM "Customer"
6      WHERE "customer_company_name" = 'ООО Пример'
7      LIMIT 1
8  )
9  ORDER BY "order_date" DESC;
10

```

Data Output Сообщения Notifications

Showing results

	order_id [PK] integer	customer_id integer	worker_id integer	order_date date	selling_price integer	product_count integer	status "char"
1	3	1	1	2025-05-27	15000	25	c
2	1	1	1	2025-04-04	250	5	c

Задание: удалить поставку с наименьшей ценой за единицу товара у поставщика 'ООО Поставка'.

Запрос DELETE с подзапросом:

Запрос История запросов

```

1  DELETE FROM "Shipment"
2  WHERE "shipment_id" = (
3      SELECT "shipment_id"
4      FROM "Scope of delivery"
5      WHERE "shipment_id" IN (
6          SELECT "shipment_id"
7          FROM "Shipment"
8          WHERE "provider_id" = (
9              SELECT "provider_id"
10             FROM "Provider"
11             WHERE "company_name" = 'ООО Поставка'
12             ORDER BY "provider_id"
13             LIMIT 1
14         )
15     )
16     ORDER BY "product_price" ASC
17     LIMIT 1
18 );
19

```

Data Output Сообщения Notifications

DELETE 0

Запрос завершён успешно, время выполнения: 93 мсек.

### 3.4 Создание индексов

В данном пункте лабораторной работы были проведены следующие действия:

## 1. Выполнение тестовых запросов без индексов

Были выполнены два выбранных запроса к базе данных, и с помощью команды EXPLAIN ANALYZE получены планы их выполнения. Зафиксировано время выполнения и использованные методы доступа к данным (например, последовательное сканирование).

Выберем 2 запроса:

А: выбор товаров по производителю.

The screenshot shows a database query interface. At the top, there are tabs for 'Запрос' (Query) and 'История запросов' (Query History). The 'Запрос' tab is active, displaying a SQL query:

```
1 EXPLAIN ANALYZE
2 SELECT * FROM "Order"
3 WHERE "customer_id" = 2 AND "order_date" > '2024-01-01';
4
```

Below the query, there are tabs for 'Data Output', 'Сообщения' (Messages), and 'Notifications'. The 'Data Output' tab is active, showing a table with the query plan. The table has a header row with 'QUERY PLAN' and 'text'. The rows of the table are:

	QUERY PLAN	text
1	Bitmap Heap Scan on "Order" (cost=4.21..14.39 rows=3 width=25) (actual time=0.387..0.400 rows=1 loops=1)	
2	Recheck Cond: (customer_id = 2)	
3	Filter: (order_date > '2024-01-01'::date)	
4	Heap Blocks: exact=1	
5	-> Bitmap Index Scan on fki_customer_id (cost=0.00..4.21 rows=8 width=0) (actual time=0.298..0.302 rows=3 loops=1)	
6	Index Cond: (customer_id = 2)	
7	Planning Time: 0.650 ms	

Запрос В: фильтрация по клиенту и дате заказа

Запрос
История запросов

1
2
3
4

EXPLAIN ANALYZE  
SELECT \* FROM "Order"  
WHERE "customer\_id" = 2 AND "order\_date" > '2024-01-01';

Data Output
Сообщения
Notifications

Showing 1

QUERY PLAN  
text

1
2
3
4
5
6
7

Bitmap Heap Scan on "Order" (cost=4.21..14.39 rows=3 width=25) (actual time=0.387..0.400 rows=1 loops=1)  
Recheck Cond: (customer\_id = 2)  
Filter: (order\_date > '2024-01-01'::date)  
Heap Blocks: exact=1  
-> Bitmap Index Scan on fki\_customer\_id (cost=0.00..4.21 rows=8 width=0) (actual time=0.298..0.302 rows=3 loop\_...)  
Index Cond: (customer\_id = 2)  
Planning Time: 0.650 ms

## 2. Создание индексов

Для каждого из запросов были созданы:

- о простой индекс (по одному полю),

Запрос
История запросов

1
2
3

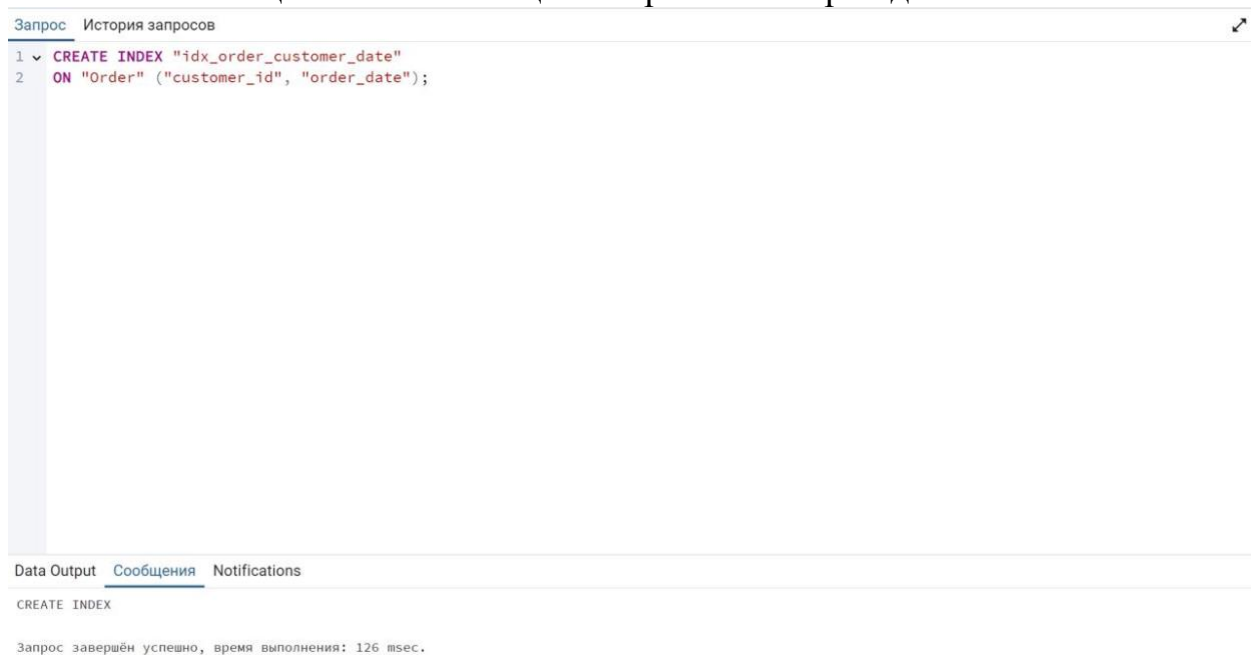
CREATE INDEX "idx\_product\_manufacturer"  
ON "Product" ("manufacturer\_id");

Data Output
Сообщения
Notifications

CREATE INDEX

Запрос завершён успешно, время выполнения: 147 мсек.

- составной индекс (по нескольким полям),  
с целью оптимизации скорости выборки данных



The screenshot shows a database query interface with a tab labeled 'Запрос' (Query) and 'История запросов' (Query History). The query text is as follows:

```
1 CREATE INDEX "idx_order_customer_date"
2 ON "Order" ("customer_id", "order_date");
```

Below the query editor, there are tabs for 'Data Output', 'Сообщения' (Messages), and 'Notifications'. The 'Сообщения' tab is selected, showing the message 'CREATE INDEX'. At the bottom, a status message reads: 'Запрос завершён успешно, время выполнения: 126 мсек.' (Query completed successfully, execution time: 126 ms).

### 3. Выполнение тех же запросов с индексами

Запросы были повторно выполнены, планы запросов с помощью EXPLAIN ANALYZE показали изменение стратегии доступа — в том числе использование индексного поиска (Index Scan или Bitmap Index Scan).

Время выполнения первого запроса составило около 0.16 мс.

Результаты после создания индексов:

В первом запросе СУБД использовала индексный поиск через Bitmap Index Scan, что обычно ускоряет доступ к данным.

Время выполнения увеличилось немного (до ~0.30 мс), что связано с небольшим объёмом данных и накладными расходами на использование индексов.

Использование индексов особенно эффективно при увеличении объёмов данных или более сложных условиях выборки.

```

1  EXPLAIN ANALYZE
2  SELECT * FROM "Product"
3  WHERE "manufacturer_id" = 1;
4

```



	QUERY PLAN	
	text	
1	Seq Scan on "Product" (cost=0.00..1.02 rows=1 width=80) (actual time=0.066..0.076 rows=1 loops=...	
2	Filter: (manufacturer_id = 1)	
3	Rows Removed by Filter: 1	
4	Planning Time: 3.969 ms	
5	Execution Time: 0.252 ms	

Запрос

История запросов

1

EXPLAIN ANALYZE

2

SELECT \* FROM "Order"

3

WHERE "customer\_id" = 2 AND "order\_date" > '2024-01-01';

4

Data Output

Сообщения

Notifications

≡+

▼

▼

SQL

QUERY PLAN

text

1

Seq Scan on "Order" (cost=0.00..1.04 rows=1 width=25) (actual time=0.053..0.058 rows=1 loops=...

2

Filter: ((order\_date > '2024-01-01'::date) AND (customer\_id = 2))

3

Rows Removed by Filter: 2

4

Planning Time: 2.760 ms

5

Execution Time: 0.162 ms

## Удаление индексов

В конце созданные индексы были удалены, чтобы вернуть базу к исходному состоянию.

## 4. Вывод по лабораторной работе:

В ходе выполнения лабораторной работы были приобретены практические навыки разработки и выполнения сложных SQL-запросов на выборку и модификацию данных в реляционной базе PostgreSQL.

