

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИТМО»**

Факультет инфокоммуникационных технологий

Дисциплина:

«Проектирование и реализация баз данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

**«Запросы на выборку и модификацию данных,
представления и индексы в PostgreSQL»**

Выполнил:

студент группы K33391

Черкес Артур Викторович

(подпись)

Проверил(а):

Говорова Марина Михайловна

(отметка о выполнении)

(подпись)

Санкт-Петербург 2023

г.

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

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, pgadmin 4.

Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
3. Изучить графическое представление запросов и просмотреть историю запросов.
4. Создать простой и составной индексы для двух произвольных запросов и сравнить время выполнения запросов без индексов и с индексами. Для получения плана запроса использовать команду EXPLAIN.

Вариант 13. БД «Ресторан»

Описание предметной области: Необходимо создать систему для обслуживания заказов клиентов в ресторане.

Сотрудники ресторана – повара и официанты.

За каждым официантом закреплены определенные столы за смену. Клиенты могут бронировать столы заранее.

Каждый повар может готовить определенный набор блюд.

Официант принимает заказ от стола и передает его на кухню. Шеф-повар распределяет блюда для приготовления между поварами. В одном заказе может быть несколько одинаковых или разных блюд.

Запас продуктов на складе не должен быть ниже заданного значения.

Цена заказа складывается из стоимости ингредиентов и наценки, которая составляет 40% стоимости ингредиентов.

БД должна содержать следующий минимальный набор сведений: Табельный номер сотрудника. ФИО сотрудника. Паспортные данные сотрудника. Категория сотрудника. Должность сотрудника. Оклад сотрудника. Наименование ингредиента. Код ингредиента. Дата закупки. Объем закупки. Количество продукта на складе. Необходимый запас продукта. Срок годности. Цена ингредиента. Калорийность (на 100г продукта). Поставщик. Наименование

блюда. Код блюда. Объем ингредиента. Номер стола. Дата заказа. Код заказа. Количество. Название блюда. Ингредиенты, входящие в блюдо. Тип ингредиента.

Выполнение:

Запросы:

- Посчитать кол-во ингредиентов в блюде.

```
1 SELECT d.name AS dish_name, COUNT(c.ingredients_id) AS ingredient_count
2 FROM restaurant.dishes d
3 LEFT JOIN restaurant.composion c ON d.id = c.dishes_id
4 GROUP BY d.name
5 ORDER BY d.name;
```

Data Output Messages Notifications

	dish_name character varying (60)	ingredient_count bigint
1	Блюдо 1	1
2	Блюдо 2	1
3	Блюдо 3	3

```
SELECT d.name AS dish_name, COUNT(c.ingredients_id) AS
ingredient_count
FROM restaurant.dishes d
LEFT JOIN restaurant.composion c ON d.id = c.dishes_id
GROUP BY d.name
ORDER BY d.name;
```

Узнать, какой повар готовит больше всего блюд

```

1 SELECT
2     s.full_name AS chef_name,
3     COUNT(skills.dishes_key) AS dish_count
4 FROM
5     restaurant.staff AS s
6 LEFT JOIN
7     restaurant.skills AS skills
8     ON s.personnel_id = skills.stuff_key
9 GROUP BY
10    s.full_name
11 ORDER BY
12    dish_count DESC;

```

Data Output Messages Notifications



	chef_name character varying (80)	dish_count bigint
1	Мария Сидорова	2
2	Петр Петров	1
3	Иван Иванов	1

SELECT

s.full_name AS chef_name,

COUNT(skills.dishes_key) AS dish_count

FROM

restaurant.staff AS s

LEFT JOIN

restaurant.skills AS skills

ON s.personnel_id = skills.stuff_key

GROUP BY

s.full_name

ORDER BY

dish_count DESC;

Какой ингредиент содержится в максимальном количестве блюд

Query Query History

```
1 SELECT
2   i.name AS ingredient_name,
3   COUNT(c.ingredients_id) AS dish_count
4 FROM
5   restaurant.ingredients AS i
6 JOIN
7   restaurant.composion AS c
8   ON i.id = c.ingredients_id
9 GROUP BY
10  i.name
11 ORDER BY
12  dish_count DESC
13 LIMIT 1;
```

Data Output Messages Notifications

	ingredient_name character varying (60)	dish_count bigint
1	Оленина	3

```
SELECT
    i.name AS ingredient_name,
COUNT(c.ingredients_id) AS dish_count
FROM
    restaurant.ingredients AS i
JOIN
    restaurant.composion AS c
    ON i.id = c.ingredients_id
GROUP BY
    i.name
ORDER BY
    dish_count DESC
LIMIT 1;
```

Количество блюд, заказанных в какой-то день:

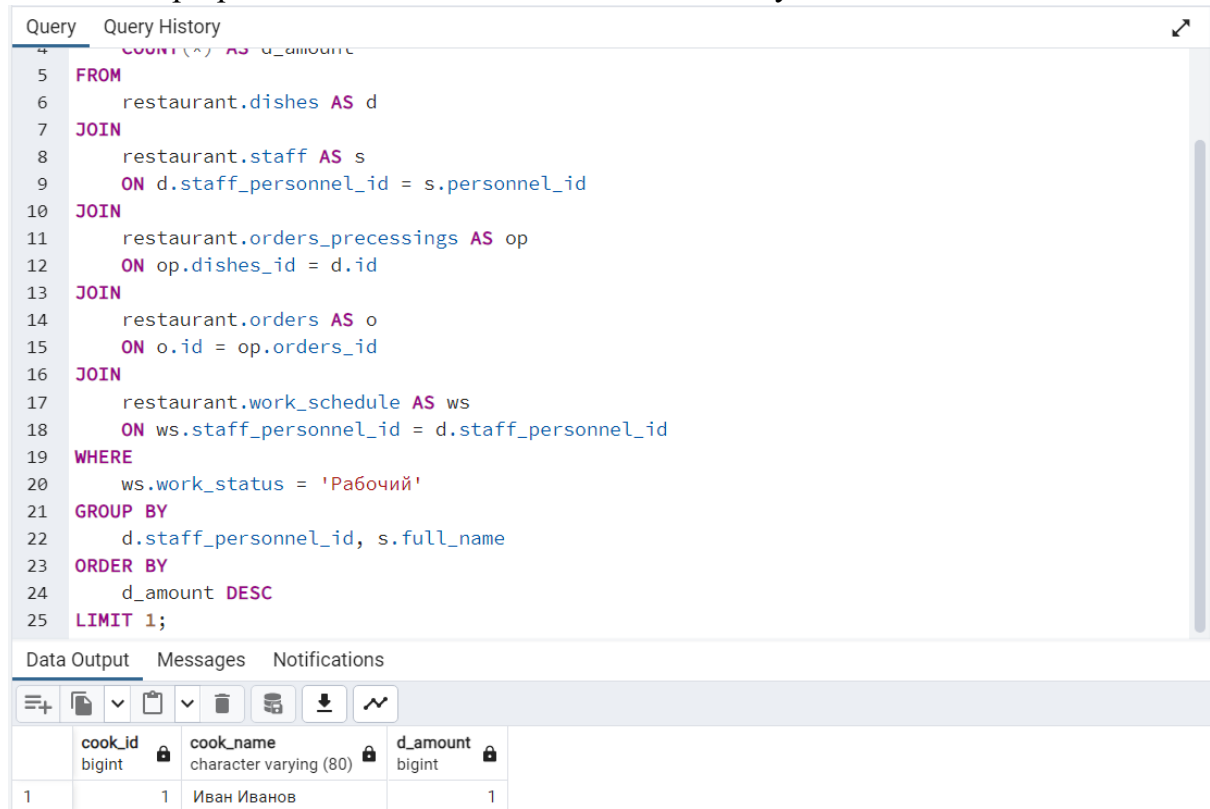
```
1 SELECT
2     d.name AS dish_name,
3     COUNT(*) AS dish_count
4 FROM
5     restaurant.orders AS o
6 JOIN
7     restaurant.orders_precessings AS op
8     ON o.id = op.orders_id
9 JOIN
10    restaurant.dishes AS d
11    ON op.dishes_id = d.id
12 WHERE
13     o.date = '2023-10-02'
14 GROUP BY
15     d.name;
```

Data Output Messages Notifications

	ingredient_name character varying (60)	dish_count bigint
1	Оленина	3

```
SELECT
    d.name AS dish_name,
    COUNT(*) AS dish_count
FROM
    restaurant.orders AS o
JOIN
    restaurant.orders_precessings AS op
    ON o.id = op.orders_id
JOIN
    restaurant.dishes AS d
    ON op.dishes_id = d.id
WHERE
    o.date = '2023-10-02'
GROUP BY
    d.name;
```

Какой повар приготовил больше всего блюд за смену



The screenshot shows a database query editor with a SQL query and its results. The query is as follows:

```
4  COUNT(*) AS d_amount
5  FROM
6    restaurant.dishes AS d
7  JOIN
8    restaurant.staff AS s
9    ON d.staff_personnel_id = s.personnel_id
10 JOIN
11    restaurant.orders_precessings AS op
12    ON op.dishes_id = d.id
13 JOIN
14    restaurant.orders AS o
15    ON o.id = op.orders_id
16 JOIN
17    restaurant.work_schedule AS ws
18    ON ws.staff_personnel_id = d.staff_personnel_id
19 WHERE
20    ws.work_status = 'Рабочий'
21 GROUP BY
22    d.staff_personnel_id, s.full_name
23 ORDER BY
24    d_amount DESC
25 LIMIT 1;
```

Below the query, there is a toolbar with icons for query execution, saving, and other functions. The results are displayed in a table with the following columns: cook_id, cook_name, and d_amount.

cook_id	cook_name	d_amount
1	Иван Иванов	1

```
SELECT
    d.staff_personnel_id AS cook_id,
    s.full_name AS cook_name,
    COUNT(*) AS d_amount
FROM
    restaurant.dishes AS d
JOIN
    restaurant.staff AS s
    ON d.staff_personnel_id = s.personnel_id
JOIN
    restaurant.orders_precessings AS op
    ON op.dishes_id = d.id
JOIN
    restaurant.orders AS o
    ON o.id = op.orders_id
JOIN
    restaurant.work_schedule AS ws
    ON ws.staff_personnel_id = d.staff_personnel_id
WHERE
    ws.work_status = 'Рабочий'
GROUP BY
    d.staff_personnel_id, s.full_name
```

ORDER BY
d_amount DESC
LIMIT 1;

Рассчитать премию каждого официанта за последние 10 дней (5% от стоимости каждого заказа).

Query Query History

```
1 SELECT
2     s.full_name AS waiter,
3     SUM(d.price * op.dishes_amount * 0.05) AS bonus
4 FROM
5     restaurant.staff AS s
6 JOIN
7     restaurant.orders AS o
8     ON s.personnel_id = o.staff_personnel_id
9 JOIN
10    restaurant.orders_precessings AS op
11    ON o.id = op.orders_id
12 JOIN
13    restaurant.dishes AS d
14    ON op.dishes_id = d.id
15 WHERE
16     o.date >= CURRENT_DATE - INTERVAL '10 days'
17 GROUP BY
18     s.full_name;
```

Data Output Messages Notifications

	waiter character varying (80)	bonus numeric
1	Мария Сидорова	55.00

Сколько закреплено столов за каждым из официантов?

Query Query History

```
1 SELECT
2     s.full_name AS официант,
3     COUNT(t.id) AS количество_столов
4 FROM
5     restaurant.tables AS t
6 INNER JOIN
7     restaurant.staff AS s
8     ON t.staff_personnel_id = s.personnel_id
9 INNER JOIN
10    restaurant.positions AS p
11    ON s.positions_id = p.id
12 WHERE
13     p.name = 'Официант'
14 GROUP BY
15     s.full_name;
```

Data Output Messages Notifications

	официант character varying (80)	количество_столов bigint
1	Мария Сидорова	2

Представления:

Список ингредиентов с калорийностью более 150 калорий на 100 грамм.

Query

Query History

↗

```
1
2 CREATE VIEW high_calorie_ingredients AS
3 SELECT
4     id,
5     product_type,
6     name,
7     calories_100g
8 FROM
9     restaurant.ingredients
10 WHERE
11     calories_100g > 150;
```

Data Output

Messages

Notifications

CREATE VIEW

Query returned successfully in 43 msec.

↗

Query

Query History

↗

```
1
2 SELECT * FROM high_calorie_ingredients;
3
```

Data Output

Messages

Notifications

≡

📄

▼

📋

▼

🗑

📦

⬇

📈

	id	product_type	name	calories_100g
	bigint	character varying (30)	character varying (60)	bigint
1	5	Рыба	Окунь	200

Представление с информацией о сотрудниках и их графике работы:

```
CREATE VIEW staff_work_schedule AS
SELECT
    s.full_name AS staff_name,
    ws.work_status AS work_status,
    ss.beginning AS shift_start,
    ss.ending AS shift_end
FROM
    staff s
JOIN
    work_schedule ws ON s.personnel_id = ws.staff_personnel_id
JOIN
    shifts_schedule ss ON ws.shift_id = ss.id;
```

Messages

Query returned successfully in 47 msec.

Query Query History

1 SELECT * FROM staff_work_schedule;

2 |

Data Output Messages Notifications

	staff_name character varying (80)	work_status character varying (20)	shift_start timestamp without time zone	shift_end timestamp without time zone
1	Иван Иванов	Рабочий	2023-10-02 00:00:00	2023-10-02 11:59:59
2	Петр Петров	Рабочий	2023-10-02 12:00:00	2023-10-02 23:59:59
3	Мария Сидорова	Отсутствует	2023-10-02 12:00:00	2023-10-02 23:59:59

Модификация данных

```
1 UPDATE restaurant.ingredients AS i
2 SET calories_100g = (
3     SELECT AVG(calories_100g)
4     FROM restaurant.ingredients
5     WHERE product_type = i.product_type
6 )
7 WHERE id = 4;
```

Data Output Messages Notifications

UPDATE 1

Query Query History

```
1 UPDATE restaurant.orders
2 SET date = '2023-10-05', status = 'Изменен', payment_state = 'Не оплачен'
3 WHERE id = (SELECT id FROM restaurant.orders WHERE payment_state = 'Не оплачен' LIMIT 1);
```

Data Output Messages Notifications

UPDATE 1

Удаление заказа, еду для которого приготовил повар с самой высокой ставкой

[Copy](#)[Copy to Query Editor](#)

```
DELETE FROM restaurant.orders_precessings
WHERE orders_id IN (
    SELECT id
    FROM restaurant.orders
    WHERE staff_personnel_id = (
        SELECT personnel_id
        FROM restaurant.staff
        WHERE positions_id = (
            SELECT positions_id
            FROM restaurant.positions
            ORDER BY working_rates DESC
            LIMIT 1)
        LIMIT 1)
    LIMIT 1));
```

Messages

Query returned successfully in 67 msec.

Создание индексов

Без индекса:

QueryQuery History

1

SELECT * FROM restaurant.dishes WHERE name = 'Блюдо 2';

2

Data OutputMessagesExplain xNotifications

	id [PK] bigint	staff_personnel_id bigint	recipe character varying (32000)	name character varying (60)	type character varying (30)
1	11	2	Рецепт блюда 2	Блюдо 2	Второе

CopyCopy to Query Editor

SELECT * FROM restaurant.dishes WHERE name = 'Блюдо 2';

Messages

Successfully run. Total query runtime: 60 msec. 1 rows affected.

Query

Query History

1

SELECT * FROM restaurant.dishes WHERE name = 'Блюдо 2';

2

Data Output

Messages

Explain

×

Notifications

Graphical

Analysis

Statistics

dishes

Query

Query History

1

SELECT * FROM restaurant.dishes WHERE name = 'Блюдо 2';

2

|

Data Output

Messages

Explain

×

Notifications

Graphical

Analysis

Statistics

#	Node
1.	→ Seq Scan on dishes as dishes Filter: ((name)::text = 'Блюдо 2')::text

Создание индекса:

Copy Copy to Query Editor

```
CREATE INDEX idx_dishes_name ON restaurant.dishes (name);
```

Messages

Query returned successfully in 39 msec.

Поиск с индексом:

```
SELECT *  
FROM restaurant.dishes  
WHERE name = 'Блюдо 2';
```

Messages

Successfully run. Total query runtime: 53 msec. 1 rows affected.

Оказался быстрее на 7 мс

Без индекса:

Query Query History

```
1 SELECT  
2   d.staff_personnel_id AS повар_id,  
3   s.full_name AS повар_имя,  
4   COUNT(*) AS количество_блюд  
5 FROM  
6   restaurant.dishes AS d  
7 JOIN  
8   restaurant.staff AS s  
9   ON d.staff_personnel_id = s.personnel_id  
10 JOIN  
11  restaurant.orders_precessings AS op  
12  ON op.dishes_id = d.id  
13 JOIN  
14  restaurant.orders AS o  
15  ON o.id = op.orders_id
```

Data Output Messages Explain x Notifications

+

📄

▼

🗑️

📦

⬇️

📶

	повар_id bigint	повар_имя character varying (80)	количество_блюд bigint
1	1	Иван Иванов	1

Query

Query History

1

CREATE INDEX idx_optimized_join

2

ON restaurant.orders_precessings (dishes_id, orders_id);

Data Output

Messages

Explain

×

Notifications

CREATE INDEX

Query returned successfully in 47 msec.

Поиск и индексом:

Query

Query History

6

restaurant.dishes AS d

7

JOIN

8

restaurant.staff AS s

9

ON d.staff_personnel_id = s.personnel_id

10

JOIN

11

restaurant.orders_precessings AS op

12

ON op.dishes_id = d.id

13

JOIN

14

restaurant.orders AS o

15

ON o.id = op.orders_id

16

JOIN

17

restaurant.work_schedule AS ws

18

ON ws.staff_personnel_id = d.staff_personnel_id

19

WHERE

20

ws.work_status = 'Рабочий'

21

GROUP BY

22

d.staff_personnel_id, s.full_name

23

ORDER BY

24

количество_блюд DESC

25

LIMIT 1;

Data Output

Messages

Explain

×

Notifications

Successfully run. Total query runtime: 45 msec.

1 rows affected.

Быстрее на 6 мс

Индексы помогают сократить время сложного запроса, но на примере простых запросов мы видим, что планировщик считает что лучше просканировать обычным способом

Выводы

SQL запросы позволяют изменять, добавлять или удалять данные, а также составлять различные выборки, подсчитывать числовые характеристики.

Сравнив время выполнения запросов с индексами и без, можно сделать вывод, что с индексами запросы выполнялись примерно столько же. Это связано с небольшим количеством данных в таблице.