## Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

#### Отчет

по лабораторной работе «Запросы на выборку и модификацию данных, представления и индексы в PostgreSQL»

по дисциплине «Базы данных»

Автор: Казарян Т.Г.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**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.

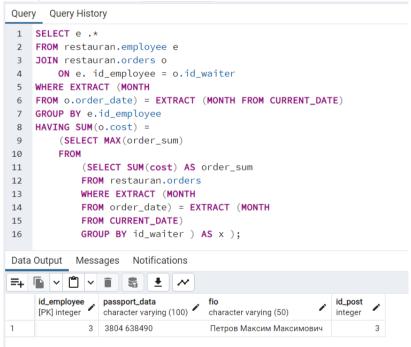
Программное обеспечение: СУБД PostgreSQL 1X, pgAdmin 4.

## Практическое задание:

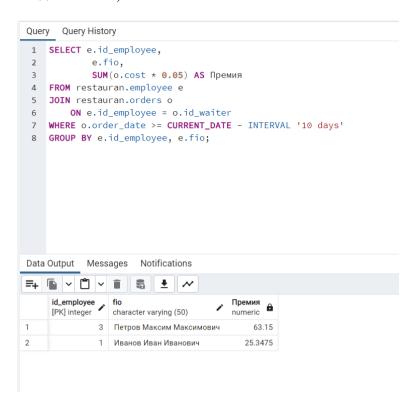
- 1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
- 2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
- 3. Изучить графическое представление запросов и просмотреть историю 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.

## Ход работы:

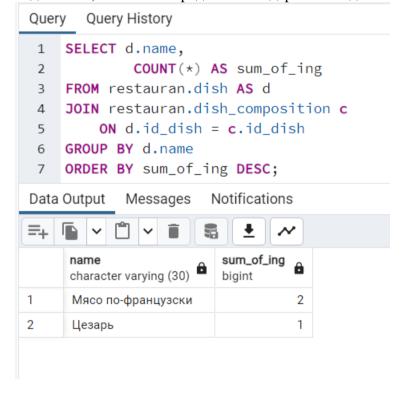
- 1. Выполнение запросов согласно индивидуальному заданию, часть 2.
- Вывести данные официанта, принявшего заказы на максимальную сумму за истекший месяц.



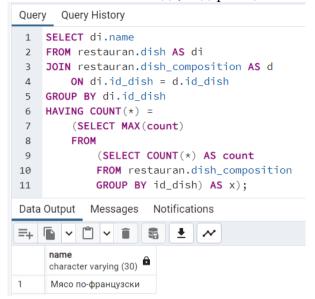
• Рассчитать премию каждого официанта за последние 10 дней (5% от стоимости каждого заказа).



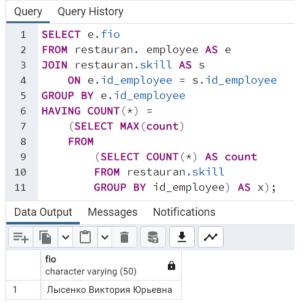
• Подсчитать, сколько ингредиентов содержит каждое блюдо.



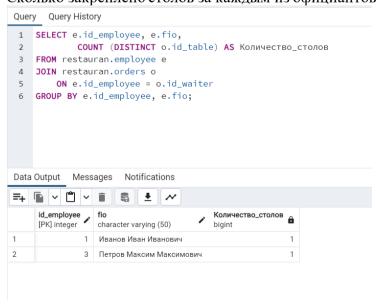
• Вывести название блюда, содержащее максимальное число ингредиентов.



• Какой повар может приготовить максимальное число видов блюд?



• Сколько закреплено столов за каждым из официантов?



• Какой из ингредиентов используется в максимальном количестве блюд?

```
1 SELECT i.name
   FROM restauran.ingredient AS i
2
3
   JOIN restauran.dish_composition AS d
       ON i.id_ingredient = d.id_ingredient
 5 GROUP BY i.id_ingredient
 6 HAVING COUNT(*) =
7
        (SELECT MAX(count)
8
9
            (SELECT COUNT(*) AS count
10
            FROM restauran.dish_composition
            GROUP BY id_ingredient) AS x);
11
Data Output Messages Notifications
                    character varying (40)
    Говядина
```

- 2. Выполнение запросов на создание представлений согласно индивидуальному заданию, часть 3.
- для расчета стоимости ингредиентов для заданного блюда;

SELECT d.name AS dish\_name,

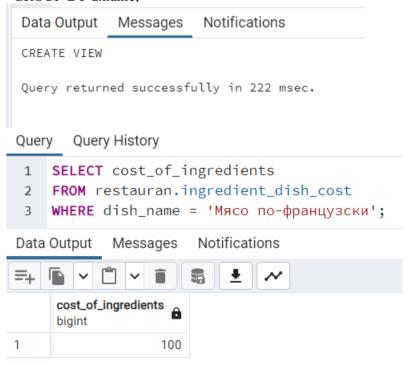
sum(co.ingredient\_volume \* pu.ingredient\_price) AS cost\_of\_ingredients

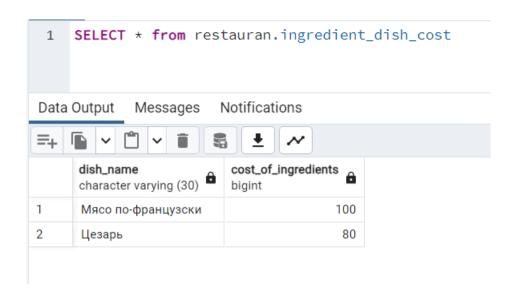
FROM restauran.dish d

JOIN restauran.dish\_composition co ON d.id\_dish = co.id\_dish

JOIN restauran.purchase pu ON co.id\_ingredient = pu.id\_ingredient

#### GROUP BY d.name;





• для всех поваров количество приготовленных блюд по каждому блюду за определенную дату.

CREATE OR REPLACE VIEW chef\_dish\_counts AS

**SELECT** 

e.id\_employee AS chef\_id,

d.name AS dish\_name,

o.order\_date::date AS preparation\_date,

COUNT(\*) AS dish\_count

## **FROM**

restauran.employee e

JOIN restauran.skill s ON e.id\_employee = s.id\_employee

JOIN restauran.dish d ON s.id\_dish = d.id\_dish

JOIN restauran.order\_composition oc ON d.id\_dish = oc.id\_dish

JOIN restauran.orders o ON oc.id\_order = o.id\_order

## **GROUP BY**

e.id\_employee, d.name, o.order\_date::date;

	chef_id integer	ê	dish_name character varying (30)	<pre>preparation_date date</pre>	dish_count bigint
1		2	Торт "Наполеон"	2023-12-15	1
2		2	Мясо по-французски	2023-12-15	1
3		4	Мясо по-французски	2023-12-15	1

# 3. Выполнение запросов на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.

## **INSERT**

INSERT into restauran.skill (id\_employee, id\_dish) values (
(SELECT id\_employee)

FROM restauran.employee

WHERE fio = Меньшин Александр Юрьевич), 2);

До:

	id_employee integer	id_dish integer	skill_id [PK] integer
1	2	2	1
2	2	1	2
3	2	3	3
4	4	1	4

## После:

	id_employee integer	id_dish integer	skill_id [PK] integer
1	2	2	1
2	2	1	2
3	2	3	3
4	4	1	4
5	4	2	5

## **UPDATE**

UPDATE restauran.ingredient SET quantity\_in\_stock = quantity\_in\_stock - 1

WHERE id\_ingredient IN (SELECT id\_ingredient

FROM restauran.dish\_composition WHERE ingredient\_volume > 1);

г —

П	$^{\prime}$	•
$\mu$	ļυ	٠

=+	<u> </u>					
	id_ingredient [PK] integer	name character varying (40)	product_type character varying (20)	required_stock double precision	quantity_in_stock double precision	caloric_content integer
1	1	Лук	Овощи	5	3.5	40
2	2	Говядина	Мясо	8	7	187
3	3	Яйцо куриное	Яйца	10	9	143

## После:

	id_ingredient [PK] integer	name character varying (40)	product_type character varying (20)	required_stock double precision	quantity_in_stock double precision	caloric_content integer
1	1	Лук	Овощи	5	3.5	40
2	2	Говядина	Мясо	8	7	187
3	3	Яйцо куриное	Яйца	10	8	143

## **DELETE**

DELETE FROM restauran.employee WHERE id\_employee NOT IN (SELECT id\_employee FROM restauran.skill) AND id\_post = 2;

## До:

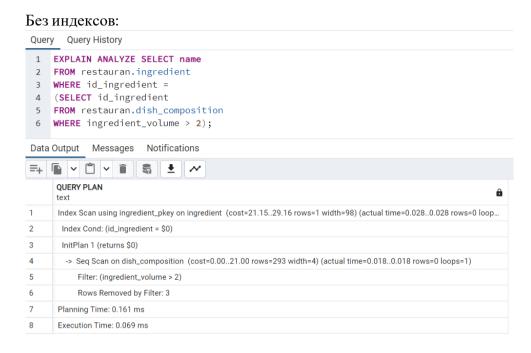
=+	<b>▶</b>   <b>□</b>   <b>∨</b>			
	id_employee [PK] integer	passport_data character varying (100)	fio character varying (50)	id_post integer
1	1	4018 998445	Иванов Иван Иванович	;
2	2	5334 584036	Лысенко Виктория Юрьевна	
3	3	3804 638490	Петров Максим Максимович	,
4	4	4567 377988	Меньшин Александр Юрьевич	
5	5	6546 646416	Козловский Марк Григорьевич	
6	6	4882 457235	Кулаков Никита Сергеевич	
7	7	1623 698877	Исаев Матвей Семёнович	

## После:

	id_employee [PK] integer	passport_data character varying (100)	fio character varying (50)	id_post integer	j	
1	1	4018 998445	Иванов Иван Иванович		3	
2	2	5334 584036	Лысенко Виктория Юрьевна		1	
3	3	3804 638490	Петров Максим Максимович		3	
4	4	4567 377988	Меньшин Александр Юрьевич		1	

### 4. Создание индексов.

## 1 Пример:



## Создание индекса:

create index NIndex on restauran.ingredient(name);

## С индексом:



## 2 Пример:

#### Без индекса:

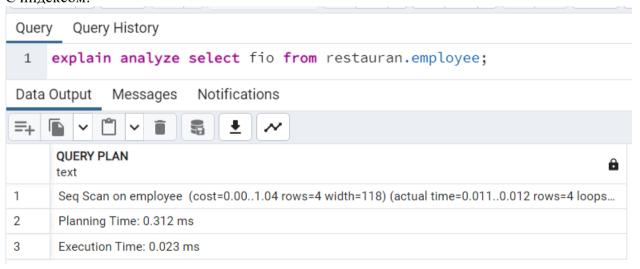
```
Query Query History
  1 EXPLAIN ANALYZE
  2 SELECT fio
         FROM restauran.employee
  3
  4 WHERE id_employee = ANY (
                   SELECT id_employee
                   FROM restauran.skill
  6
                    WHERE id_dish = 1
  8
        );
  9
Data Output Messages Notifications

    □
    √
    □
    □
    □
    □
    ✓
    □
    □
    ✓
    □
    ✓
    □
    ✓
    □
    ✓
    □
    ✓
    □
    ✓
    □
    ✓
    □
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓
    ✓</
            QUERY PLAN
            Hash Semi Join (cost=35.62..48.39 rows=10 width=118) (actual time=0.058..0.059 rows=2 loops=1)
2
             Hash Cond: (employee.id_employee = skill.id_employee)
              -> Seq Scan on employee (cost=0.00..12.10 rows=210 width=122) (actual time=0.018..0.018 rows=4...
3
4
              -> Hash (cost=35.50..35.50 rows=10 width=4) (actual time=0.017..0.017 rows=2 loops=1)
5
                    Buckets: 1024 Batches: 1 Memory Usage: 9kB
6
                    -> Seq Scan on skill (cost=0.00..35.50 rows=10 width=4) (actual time=0.008..0.009 rows=2 loops...
7
                         Filter: (id_dish = 1)
8
                         Rows Removed by Filter: 3
9
            Planning Time: 0.183 ms
10
            Execution Time: 0.091 ms
```

### Создание индекса:

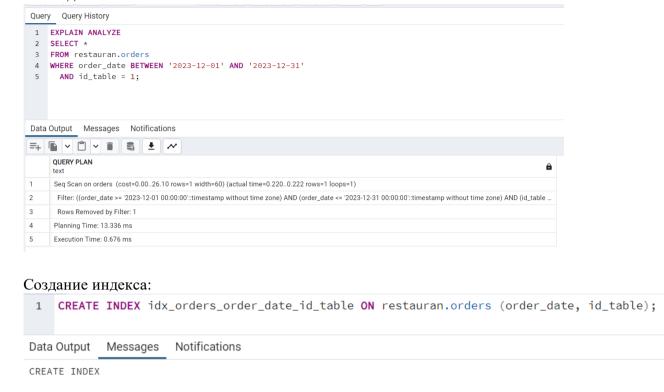
1 create index FIOIndex on restauran.employee(fio);

## С индексом:



## 3 Пример:

#### Без индекса:



## С индексом:

Query returned successfully in 68 msec.



После создания индексов запросы выполняются быстрее.

• Индексы помогают ускорить выполнение запросов, особенно тех, которые включают условия фильтрации или сортировки по индексированным столбцам. Однако, создание индексов также может повлечь проблемы при вставке, обновлении или удалении данных, так как индексы должны быть поддержаны и обновлены.

## Вывод

В ходе лабораторной работы я научился создавать представления и запросы на выборку данных к базе данных PostgreSQL, используя подзапросы при модификации данных и индексов. По моему мнению, эти знания полезны и часто применяются на практике.