

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4**

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

**по дисциплине «Проектирование и реализация баз данных»**

**Обучающийся Камалов Руслан Олегович**

**Факультет прикладной информатики**

**Группа К3241**

**Направление подготовки 09.03.03 Прикладная информатика**

**Образовательная программа Мобильные и сетевые технологии 2023**

**Преподаватель Говорова Марина Михайловна**

Санкт-Петербург  
2024/2025

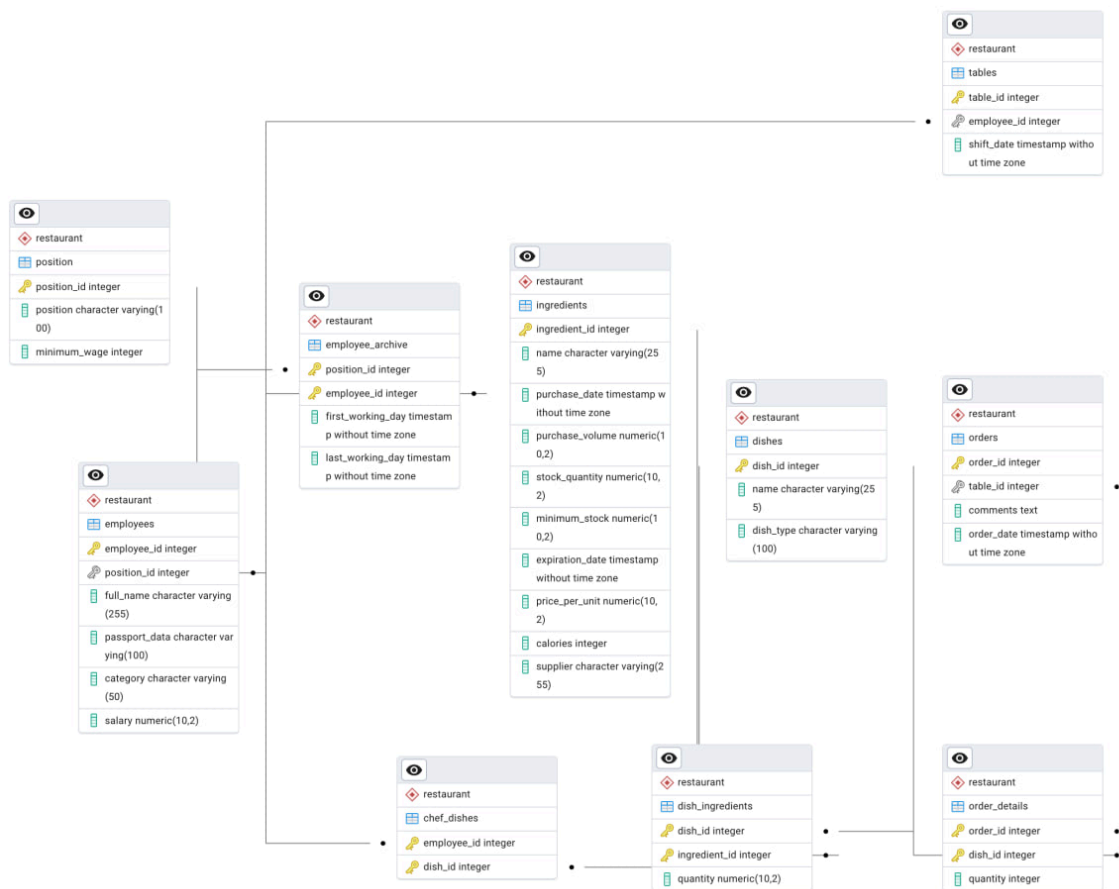
## 1. Цель работы:

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

## 2. Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию лабораторной работы №2, 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.

## 3. Схема базы данных(ЛР 3)



## 4. Выполне:

1. Вывести данные официанта, принявшего заказы на максимальную сумму за

Query Query History

```
1 with sales as (select e.employee_id, e.full_name, sum(dp.price * od.quantity) as total_sales
2 from restaurant.employees e
3 join restaurant.tables t on e.employee_id = t.employee_id
4 join restaurant.orders o on t.table_id = o.table_id
5 join restaurant.order_details od on o.order_id = od.order_id
6 join restaurant.dishes d on od.dish_id = d.dish_id
7 join restaurant.dish_price dp on d.dish_id = dp.dish_id
8 where e.position_id = (select position_id from restaurant."position" where position = 'Официант')
9 and o.order_date between date_trunc('month', current_date - interval '1 month') and date_trunc('month', current_date)
10 group by e.employee_id)
11 select employee_id, full_name, total_sales
12 from sales
13 where total_sales = (select max(total_sales) from sales);
```

Data Output Messages Explain X Notifications

Graphical Analysis Statistics

Total rows: 1 Query complete 00:00:00.039 LF Ln 1, Col 45

истекший месяц.

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

restaurant/postgres@pin.db

Query Query History

```
1 select e.employee_id, e.full_name, sum(od.quantity * dp.price * 0.05) as cash
2 from restaurant.employees as e
3 join restaurant."tables" t on e.employee_id = t.employee_id
4 join restaurant.orders o on t.table_id = o.table_id
5 join restaurant.order_details od on od.order_id = o.order_id
6 join restaurant.dish_price dp on dp.dish_id = od.dish_id
7 where
8     e.position_id = (select position_id from restaurant."position" where position = 'Официант')
9     and o.order_date >= current_date - interval '10 days'
10 group by e.employee_id
```

Data Output Messages Explain X Notifications

Graphical Analysis Statistics

The diagram illustrates the execution plan for the query. It starts with the 'orders' table, which is joined with the 'employees' table via a 'Nested Loop Inner Join'. This is followed by another 'Nested Loop Inner Join' with the 'tables' table, and a third 'Nested Loop Inner Join' with the 'order\_details' table. The final step is an 'Aggregation' operation, which is also joined with the 'position' table via a 'Nested Loop Inner Join'. The plan shows a sequential flow of joins leading to the final aggregation.

Total rows: 1 Query complete 00:00:00.100 LF Ln 10, Col 23

### 3. Подсчитать, сколько ингредиентов содержит каждое блюдо.

restaurant/postgres@pin.db

Query Query History

```
1 select d.dish_id, d."name", count(di.ingredient_id) as ingredients_count
2 from restaurant.dishes d
3 left join restaurant.dish_ingredients di on di.dish_id = d.dish_id
4 group by d.dish_id
```

Data Output Messages Explain X Notifications

Graphical Analysis Statistics

```
graph LR
    A[dish_ingredients] --> C[Hash Right Join]
    B[dishes] --> C
    C --> D[Aggregate]
```

Total rows: 1 Query complete 00:00:00.059 LF Ln 4, Col 19

#### 4. Вывести название блюда, содержащее максимальное число ингредиентов.

restaurant/postgres@pin.db

Query Query History

```
1 with dd as (select d.dish_id, d."name", count(di.ingredient_id) as ingredients_count
2 from restaurant.dishes d
3 left join restaurant.dish_ingredients di on di.dish_id = d.dish_id
4 group by d.dish_id)
5 select dish_id, name, ingredients_count
6 from dd
7 where ingredients_count = (select max(ingredients_count) from dd)
```

Data Output Messages Explain X Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	dish_id [PK] integer	name character varying (255)	ingredients_count bigint
1	1	Борщ	2

Total rows: 1 Query complete 00:00:01.160 LF Ln 7, Col 6

## 5. Какой повар может приготовить максимальное число видов блюд?

restaurant/postgres@pin.db

Query Query History

```
1 with meow as (select e.employee_id, e.full_name, count(cd.dish_id) as dishes_count
2 from restaurant.employees e
3 join restaurant.chef_dishes cd on e.employee_id = cd.employee_id
4 where e.position_id = (select position_id from restaurant.position where position = 'Повар')
5 group by e.employee_id)
6 select employee_id, full_name, dishes_count
7 from meow
8 where dishes_count = (select max(dishes_count) from meow)
```

Loading...

Data Output Messages Explain X Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	employee_id [PK] integer	full_name character varying (255)	dishes_count bigint
1	2	Петров Петр Петрович	2

Total rows: 1 Query complete 00:00:01.091 LF Ln 8, Col 5

6. Сколько закреплено столов за каждым из официантов за сегодняшний день?

restaurant/postgres@pin.db

Query Query History

```
1 select e.employee_id, e.full_name, count(t.table_id) as table_count
2 from restaurant.employees e
3 left join restaurant.tables t on e.employee_id = t.employee_id
4 where e.position_id = (select position_id from restaurant.position where position = 'Официант') and t.shift_date = c
5 group by e.employee_id
```

Data Output Messages Explain X Notifications

Graphical Analysis Statistics

The diagram illustrates the execution plan for the provided SQL query. It starts with a table icon representing the 'employees' table. An arrow leads to a join symbol (two lines meeting at a point). From the join, an arrow points to another table icon representing the 'tables' table. A second arrow from the join points to a subquery icon (a circle with a smaller circle inside). This subquery icon points to a table icon representing the 'position' table. Finally, an arrow from the join points to an 'Aggregate' icon (a grid with a plus sign), which represents the final aggregation step. The status bar at the bottom indicates 'Total rows: 1', 'Query complete 00:00:00.050', and 'Ln 5, Col 23'.

Total rows: 1 Query complete 00:00:00.050 LF Ln 5, Col 23



## 7. Какой из ингредиентов используется в максимальном количестве блюд?

restaurant/postgres@pin.db

Query Query History

```
1 with ogo as (select i.ingredient_id, i.name, count(di.dish_id) as dishes_count
2 from restaurant.ingredients i
3 join restaurant.dish_ingredients di on i.ingredient_id = di.ingredient_id
4 group by i.ingredient_id, i.name)
5 select ingredient_id, name, dishes_count
6 from ogo
7 where dishes_count = (select max(dishes_count) from ogo)
8
```

Data Output Messages Explain X Notifications

Showing rows: 1 to 5 Page No: 1 of 1

	ingredient_id [PK] integer	name character varying (255)	dishes_count bigint
1	4	Сахар	1
2	2	Мука	1
3	3	Яйца	1
4	1	Молоко	1
5	5	Соль	1

Total rows: 5 Query complete 00:00:01.620 LF Ln 7, Col 5

## 8. Создать представление для расчета стоимости ингредиентов для заданного блюда

Query Query History

```
1 create or replace view restaurant.dish_ingredients_cost as
2 select d.dish_id, d.name as dish_name, i.ingredient_id, i.name as ingredient_name, di.quantity, i.price_per_unit, (di.quantity * i.price_per_unit) as ingredient_cost
3 from restaurant.dishes d
4 join restaurant.dish_ingredients di on di.dish_id = d.dish_id
5 join restaurant.ingredients i on di.ingredient_id = i.ingredient_id;
6 select * from restaurant.dish_ingredients_cost;
```

Data Output Messages Explain X Notifications

Showing rows: 1 to 5 Page No: 1 of 1

	dish_id integer	dish_name character varying (255)	ingredient_id integer	ingredient_name character varying (255)	quantity numeric (10,2)	price_per_unit numeric (10,2)	ingredient_cost numeric
1	1	Борщ	1	Молоко	0.50	80.50	40.2500
2	1	Борщ	2	Мука	0.30	30.00	9.0000
3	2	Стейк	3	Яйца	2.00	120.00	240.0000
4	3	Салат "Цезарь"	4	Сахар	0.10	50.00	5.0000
5	4	Тирамису	5	Соль	0.05	20.00	1.0000

Total rows: 5 Query complete 00:00:00.078 LF Ln 6, Col 4

restaurant/postgres@pin.db
No limit

---

**Query**   Query History

```

1 create or replace view restaurant.chef_dishes_prepared as
2 select e.employee_id, e.full_name as chef_name, date(o.order_date), count(d.dish_id) as dishes_prepared
3 from restaurant.employees e
4 join restaurant.chef_dishes cd on e.employee_id = cd.employee_id
5 join restaurant.dishes d on cd.dish_id = d.dish_id
6 join restaurant.order_details od on od.dish_id = d.dish_id
7 join restaurant.orders o on o.order_id = od.order_id
8 where e.position_id = (select position_id from restaurant.position where position = 'Повар')
9 group by e.employee_id, date(o.order_date);
10 select * from restaurant.chef_dishes_prepared;
```

---

**Data Output**   Messages   Explain X   Notifications

SQL
Showing rows: 1 to 1
Page No: 1 of 1

	employee_id <small>integer</small>	chef_name <small>character varying (255)</small>	date <small>date</small>	dishes_prepared <small>bigint</small>
1	2	Петров Петр Петрович	2025-05-19	2

Total rows: 1
Query complete 00:00:00.058
LF   Ln 10, Col 4

10. Insert

restaurant/postgres@pin.db

Query History

```
1 insert into restaurant.orders (order_id, table_id, "comments", order_date)
2 select
3     (select max(order_id) + 1 from restaurant.orders),
4     (select table_id from restaurant."tables" where table_id = 3),
5     'Без лактозы',
6     CURRENT_TIMESTAMP;
7 select * from restaurant.orders;
```

Data Output Messages Notifications

Showing rows: 1 to 19 Page No: 1 of 1

	order_id [PK] integer	table_id integer	comments text	order_date timestamp without time zone
8	8	3	Без лактозы	2025-05-19 16:18:42.398373
9	9	3	Без лактозы	2025-05-19 16:21:30.622457
10	10	3	Без лактозы	2025-05-19 16:21:40.469818
11	11	3	Без лактозы	2025-05-19 16:24:58.619243
12	12	3	Без лактозы	2025-05-19 16:25:10.405032
13	13	3	Без лактозы	2025-05-19 16:25:56.557553
14	14	3	Без лактозы	2025-05-19 16:26:54.560145
15	15	3	Без лактозы	2025-05-19 16:27:06.903631
16	16	3	Без лактозы	2025-05-19 16:32:04.942628
17	17	3	Без лактозы	2025-05-19 16:34:14.50664
18	18	3	Без лактозы	2025-05-19 16:34:20.154957
19	19	3	Без лактозы	2025-05-28 18:45:21.455964

Total rows: 19 Query complete 00:00:00.158 LF Ln 8, Co

Graphical Analysis Statistics

ORDERS\_pkey Limit Result Insert TABLES\_pkey Result

Total rows: 1 Query complete 00:00:00.042 LF Ln 6, Co

11. Update

QueryQuery History

Scratch Pad

```
1 update restaurant.employees
2 set salary = salary * 1.15
3 where position_id = (select position_id from restaurant."position" where position = 'Официант');
4 select * from restaurant.employees;
5
```

Data OutputMessagesNotifications

Showing rows: 1 to 5Page No: 1of 1

	employee_id [PK] integer	position_id integer	full_name character varying (255)	passport_data character varying (100)	category character varying (50)	salary numeric (10,2)
1		2	Петров Петр Петрович	9876 543210	Senior	55000.00
2		3	Сидорова Анна Михайловна	4567 890123	Manager	65000.00
3		5	Смирнов Алексей Дмитриевич	3210 987654	Cleaner	27000.00
4		1	Иванов Иван Иванович	1234 567890	Junior	73128.96
5		4	Кузнецова Елена Сергеевна	6543 210987	Middle	79984.80

Total rows: 5Query complete 00:00:00.053

GraphicalAnalysisStatistics

position

employees

Update

Successfully run. Total query runtime: 41 msec. 1 rows affected.

Total rows: 1Query complete 00:00:00.041

12. Delete

QueryQuery History

Scratch Pad X

```
1 delete from restaurant.ingredients
2 where ingredient_id not in (
3     select ingredient_id from restaurant.dish_ingredients
4 );
5 select * from restaurant.ingredients;
```

Data OutputMessagesNotifications

Showing rows: 1 to 5Page No: 1 of 1

	ingredient_id [PK] integer	name character varying (255)	purchase_date timestamp without time zone	purchase_volume numeric (10,2)	stock_quantity numeric (10,2)	minimum_stock numeric (10,2)	expiration_date timestamp without time zone	price_per_unit numeric (10,2)	calories integer	supplier character varying (255)
1	1	Молоко	2025-05-01 00:00:00	10.50	8.20	5.00	2025-06-01 00:00:00	80.50	42	Ферма "Белая корова"
2	2	Мука	2025-04-15 00:00:00	25.00	15.00	10.00	2025-10-15 00:00:00	30.00	364	ЗАО "Зернопродукт"
3	3	Яйца	2025-05-10 00:00:00	50.00	30.00	20.00	2025-06-10 00:00:00	120.00	155	Птицефабрика "Солнечная"
4	4	Сахар	2025-03-20 00:00:00	15.00	5.00	3.00	2025-09-20 00:00:00	50.00	387	ООО "Сладкий мир"
5	5	Соль	2025-01-10 00:00:00	5.00	2.50	1.00	2026-01-10 00:00:00	20.00	0	Компания "Соль Земли"

Total rows: 5Query complete 00:00:00.053LF Ln 5, Col

GraphicalAnalysisStatistics

SearchZoom InZoom OutDownload

dish\_ingredients

ingredients

Delete

Total rows: 1Query complete 00:00:00.067LF Ln 4, Col

### 13. Query history

restaurant/postgres@pin.db

No limit

Query

Query History

Show queries generated internally by pgAdmin?

Remove

Remove All

28.05.2025 19:13:40

5

53 msec

Date

Rows affected

Duration

Copy

Copy to Query Editor

```

delete from restaurant.ingredients where ingredient_id not in (
  select ingredient_id from restaurant.dish_ingredients
);
select * from restaurant.ingredients;

```

Messages

Successfully run. Total query runtime: 53 msec. 5 rows affected.

Today - 28.05.2025

▶ delete from restaurant.ingredients where ingredient\_id not in (

19:20:23

▶ update restaurant.employees set salary = salary \* 1.15 where

19:20:00

▶ insert into restaurant.orders (order\_id, table\_id, "comment...")

19:19:19

▶ delete from restaurant.ingredients where ingredient\_id not in (

19:13:40

▶ insert into restaurant.ingredients (ingredient\_id, name, pu...

19:09:43

▶ update restaurant.employees set salary = salary \* 1.15 where

18:59:14

▶ insert into restaurant.orders (order\_id, table\_id, "comment...")

18:45:21

Data Output

Messages

Explain

Notifications

Successfully run. Total query runtime: 67 msec.

1 rows affected.

Total rows: 1

Query complete 00:00:00.067

Scratch Pad

LF

Ln 4, Col 1

#### 14. Запрос без индекса

Query

Query History

1

▼

explain analyze

2

select \* from restaurant.orders where order\_date between '2025-05-01' and '2025-05-31';

Data Output

Messages

Explain ×

Notifications

≡

📄

▼

🗑️

📄

📄

📄

📄

SQL

Showing rows: 1 to 4

Page No: 1

of 1

⏪

⏩

	QUERY PLAN	
	text	🔒
1	Seq Scan on orders (cost=0.00..26.05 rows=5 width=48) (actual time=0.052..0.062 rows=21 loops=1)	
2	Filter: ((order_date >= '2025-05-01 00:00:00'::timestamp without time zone) AND (order_date <= '2025-05-31 00:00:00'::timestamp without time zon...	
3	Planning Time: 0.193 ms	
4	Execution Time: 0.089 ms	

Total rows: 4

Query complete 00:00:00.058

LF

Ln 2, Col 8

## 15. Запрос с индексом

restaurant/postgres@pin.db

**Query** Query History

```

1 explain analyze
2 select * from restaurant.orders where order_date between '2025-05-01' and '2025-05-31';

```

---

**Data Output** Messages Explain X Notifications

Showing rows: 1 to 4 Page No: 1 of 1

	QUERY PLAN text
1	Seq Scan on orders (cost=0.00..1.31 rows=1 width=48) (actual time=0.017..0.022 rows=21 loops=1)
2	Filter: ((order_date >= '2025-05-01 00:00:00':timestamp without time zone) AND (order_date <= '2025-05-31 00:00:00':timestamp without time zon...
3	Planning Time: 0.949 ms
4	Execution Time: 0.061 ms

Total rows: 4    Query complete 00:00:00.040    LF    Ln 2, Col 8

## **5. Выводы:**

В ходе работы в pgAdmin была успешно спроектирована и реализована база данных "Ресторан". Созданы все необходимые таблицы (EMPLOYEES, DISHES, ORDERS и др.) с корректными связями и ограничениями (PRIMARY KEY, FOREIGN KEY, CHECK). Данные заполнены и проверены на целостность. Ошибки исправлены. В том числе я овладел практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.