

**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**

**Тема задания:** «Запросы на выборку и модификацию данных.  
Представления. Работа с индексами»

**Обучающийся:** Анисимов Владислав Андреевич, группа К3240

**Проверяющий:** Говорова М.М., преподаватель

Санкт – Петербург,  
2025

## СОДЕРЖАНИЕ

Цель работы.....	3
Практическое задание.....	4
Выполнение.....	5
1.1 Создание запросов.....	6
1.2 Создание представлений.....	10
2 Запросы на (INSERT, UPDATE, DELETE).....	11
3 Создание индексов.....	13
Выводы.....	14

## **Цель работы**

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

## **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.

## Выполнение

Схема базы данных:

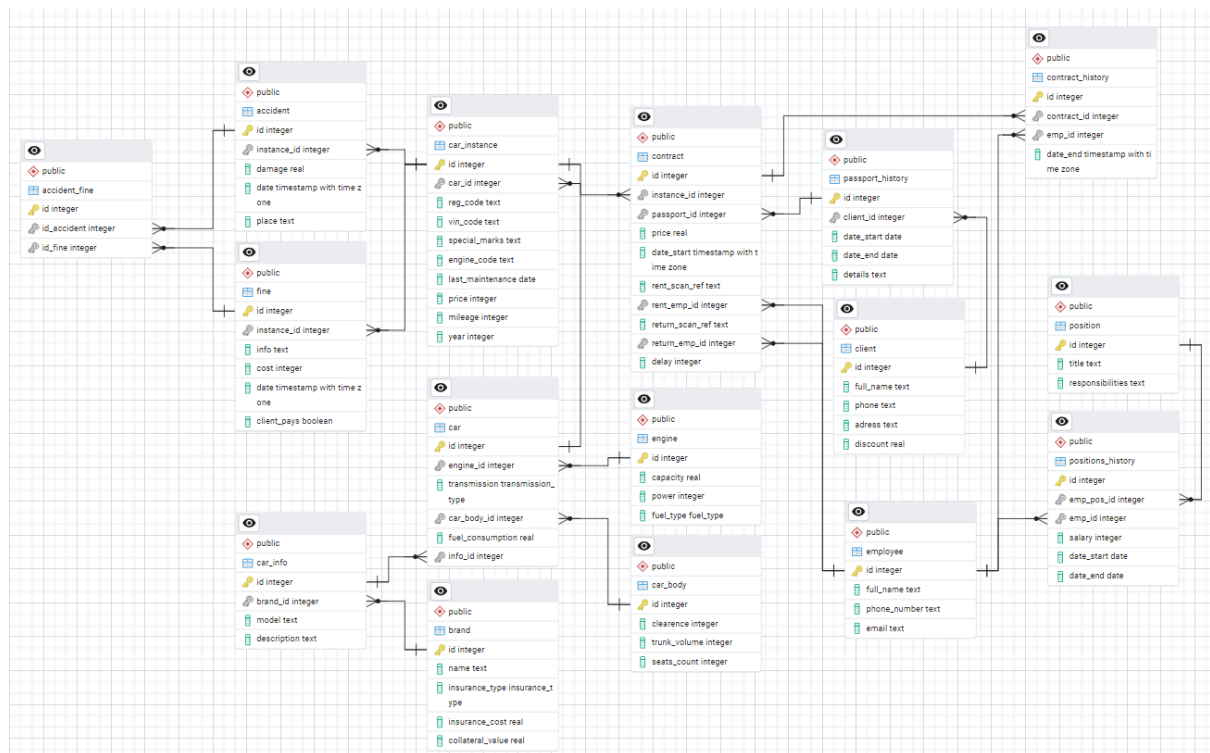


Рис 1 - Схема базы данных

## 1.1 Создание запросов

- Рассчитать выручку компании за последний календарный месяц.

```
SELECT SUM(earned) as profit FROM
  (SELECT contract.id,
    contract.date_start::date,
    ROUND (
      (SELECT EXTRACT(epoch
        FROM ( (COALESCE (
          (SELECT MIN(contract_history.date_end)
            FROM contract_history
            WHERE contract_history.contract_id=contract.id
            and contract_history.date_end >= '2025-03-01'),
          (SELECT MAX(contract_history.date_end)
            FROM contract_history
            WHERE contract_history.contract_id=contract.id
            and contract_history.date_end < '2025-03-01')
        + interval '1' HOUR * contract.delay)) -
      (CASE
        WHEN contract.date_start < '2025-02-01'
          THEN (SELECT MIN(contract_history.date_end)
            FROM contract_history
            WHERE contract_history.contract_id=contract.id)
        WHEN contract.date_start >= '2025-02-01'
          THEN contract.date_start
        END))) / 3600)
      * (contract.price / 24.0
      * (1-COALESCE(client.discount, 0)))
    ) as earned
  FROM contract
  JOIN contract_history
ON(contract_history.contract_id=contract.id)
  JOIN passport_history ON(passport_history.id=passport_id)
  JOIN client ON(client.id=client_id)
WHERE contract_history.date_end::date >= '2025-02-01'
and contract.date_start::date < '2025-03-01'
GROUP BY contract.id, client.id)
```


	profit double precision 
1	2130208

Рис 2 - результат выполнения запроса 1

- Автомобили какой марки чаще всего брались в прокат?

```
SELECT brand.name, COUNT(brand.name) as times
FROM brand
INNER JOIN car_info ON(car_info.brand_id=brand.id)
INNER JOIN car ON(car.info_id=car_info.id)
INNER JOIN car_instance ON(car_instance.car_id=car.id)
INNER JOIN contract ON(contract.instance_id=car_instance.id)
GROUP BY brand.name
ORDER BY times DESC
```

	name text	times bigint
1	BMW	246
2	Mercedes-Benz	212
3	Hyundai	141
4	Ford	119
5	Audi	119
6	Toyota	116
7	Chevrolet	92
8	Honda	92
9	Volkswagen	86

Рис 3 - результат выполнения запроса 2

- Определить убытки от простоя автомобилей за вчерашний день.

```
SELECT SUM(price / 1000 * 3) FROM car_instance
WHERE id NOT IN
(SELECT contract.instance_id
FROM contract
WHERE contract.date_start::date < NOW()::date
- interval '1 day' - interval '2 month'
and
(SELECT MAX(date_end)::date
FROM contract_history
WHERE contract_history.contract_id=contract.id)
>= NOW()::date - interval '2 month'
)
```

	sum bigint
1	49800

Рис 4 - результат выполнения запроса 3

- Вывести данные автомобиля, имеющего максимальный пробег.

```
(SELECT name, model, capacity, fuel_type, transmission, reg_code,
vin_code, last_maintenance, price, mileage, year
FROM car_instance
INNER JOIN car ON(car_id=car.id)
INNER JOIN car_info ON(info_id=car_info.id)
INNER JOIN engine ON(engine_id=engine.id)
INNER JOIN brand ON(brand_id=brand.id)
WHERE mileage=(SELECT MAX(mileage) FROM car_instance))
```

	name text	model text	capacity real	fuel_type fuel_type	transmission transmission_type	reg_code text	vin_code text	last_maintenance date	price integer	mileage integer	year integer
1	Volkswagen	Golf	1.4	AI-95	manual	X345XE78	XDL321098L0123456	2024-11-15	800000	148524	2015

Рис 5 - результат выполнения запроса 4

- Какой автомобиль суммарно находился в прокате дольше всех.

```
SELECT name, model, capacity, fuel_type, transmission, reg_code,
vin_code, last_maintenance, price, mileage, year, in_rent FROM
(SELECT instance_id, SUM(period) as in_rent
FROM
(SELECT car_instance.id as instance_id,
MAX(date_end) - contract.date_start as period
FROM contract
INNER JOIN contract_history
ON(contract_history.contract_id=contract.id)
INNER JOIN car_instance
ON(contract.instance_id=car_instance.id)
GROUP BY contract.id, car_instance.id
ORDER BY car_instance.id)
GROUP BY instance_id)
INNER JOIN car_instance ON(instance_id=car_instance.id)
INNER JOIN car ON(car_id=car.id)
INNER JOIN car_info ON(info_id=car_info.id)
```



```
INNER JOIN engine ON(engine_id=engine.id)
INNER JOIN brand ON(brand_id=brand.id)
ORDER BY in_rent DESC LIMIT 1
```

	name text	model text	capacity real	fuel_type fuel_type	transmission transmission_type	reg_code text	vin_code text	last_maintenance date	price integer	mileage integer	year integer	in_rent interval
1	Hyundai	Tucson	2	AI-95	auto	H789H078	XVF987654F4567890	2024-10-30	1850000	55000	2019	1039 days 1340:01:00

Рис 6 - результат выполнения запроса 5

- Определить, каким количеством автомобилей каждой марки и модели владеет компания.

```
SELECT name as brand, model, COUNT(*) FROM car_instance
INNER JOIN car ON(car_id=car.id)
INNER JOIN car_info ON(info_id=car_info.id)
INNER JOIN engine ON(engine_id=engine.id)
INNER JOIN brand ON(brand_id=brand.id)
GROUP BY name, model
```

	brand text	model text	count bigint
1	Toyota	Camry	2
2	Mercedes-Benz	E-Class	2
3	Audi	A4	1
4	Volkswagen	Golf	3
5	Honda	Civic	1
6	Hyundai	Tucson	2
7	Chevrolet	Camaro	2
8	Ford	F-150	1
9	BMW	X5	3

Рис 7 - результат выполнения запроса 6

- Определить средний “возраст” автомобилей компании.

```
SELECT ROUND(AVG(year), 2) as avg_year FROM car_instance
INNER JOIN car ON(car_id=car.id)
INNER JOIN car_info ON(info_id=car_info.id)
INNER JOIN engine ON(engine_id=engine.id)
INNER JOIN brand ON(brand_id=brand.id)
```

	avg_year numeric
1	2017.88

Рис 8 - результат выполнения запроса 7

## 1.2 Создание представлений

- Какой автомобиль ни разу не был в прокате?

```
CREATE OR REPLACE VIEW new_cars AS
SELECT car_instance.id, name, model, capacity, fuel_type,
transmission, reg_code, vin_code, last_maintenance, price,
mileage, year
FROM car_instance
INNER JOIN car ON(car_id=car.id)
INNER JOIN car_info ON(info_id=car_info.id)
INNER JOIN engine ON(engine_id=engine.id)
INNER JOIN brand ON(brand_id=brand.id)
WHERE car_instance.id NOT IN (SELECT DISTINCT car_instance.id
FROM car_instance
INNER JOIN contract ON(instance_id=car_instance.id));

SELECT * FROM new_cars
```

	id integer	name text	model text	capacity real	fuel_type fuel_type	transmission transmission_type	reg_code text	vin_code text	last_maintenance date	price integer	mileage integer	year integer
1	17	Toyota	Camry	2.5	AI-95	auto	M8460A178	JN8AZ28R59T103357	2025-03-20	3000000	72000	2021

Рис 9 - результат выполнения запроса 8

- Вывести данные клиентов, не вернувших автомобиль вовремя.

```
CREATE OR REPLACE VIEW clients_with_delay AS
SELECT DISTINCT client.* FROM contract
INNER JOIN passport_history ON(passport_id=passport_history.id)
INNER JOIN client ON(client.id=passport_history.client_id)
WHERE delay > 0
ORDER BY client.id;

SELECT * FROM clients_with_delay
```

Data Output Messages Notifications								
Showing rows: 1 to 67 Page No: 1 of 1								
	id	full_name	phone	address	discount			
	integer	text	text	text	real			
59	88	Valma Weathers	+380 (724) 768-1979	69960 Bluestem Avenue	[null]			
60	90	Charita McKee	+63 (222) 304-2295	5 Stone Corner Way	[null]			
61	91	Codee Keely	+58 (393) 812-6728	963 Sachtjen Terrace	[null]			
62	92	Gilligan Kittless	+1 (904) 142-3576	4671 Chinook Lane	[null]			
63	94	Herta Ewebank	+7 (822) 116-2580	3 Leroy Road	0.05			
64	96	Chrissy Jantet	+52 (774) 165-3141	024 Kipling Road	[null]			
65	97	Yulma Seabrocke	+86 (242) 335-6544	9 Mockingbird Circle	[null]			
66	98	Ynes Lorimer	+86 (340) 676-4313	76313 Susan Parkway	[null]			
67	100	Rubetta Marsham	+598 (580) 340-7423	03 Artisan Parkway	[null]			
Total rows: 67 Query complete 00:00:00.092							LF Ln 6, Col 20	

Рис 10 - результат выполнения запроса 9

## 2 Запросы на (INSERT, UPDATE, DELETE)

### 1. INSERT

Вставим данные о новом штрафе на авто с гос. номером “X345XE78”

```
INSERT INTO fine (id, instance_id, info, cost, date, client_pays)
SELECT (SELECT MAX(id)+1 FROM fine), id, '12.9', 500, '2025-02-17
10:31:17+03', true
FROM car_instance
WHERE reg_code='X345XE78'
```

	id	instance_id	info	cost	date	client_pays	id	car_id	reg_code	vin_code	special_marks	engine_code
	integer	integer	text	integer	timestamp with time zone	boolean	integer	integer	text	text	text	text
1	1001	13	12.9	500	2025-02-17 10:31:17+03	true	13	16	X345XE78	XDL321098L0123456	[null]	VWEA2114TSI34567
2	413	13	12.6	1500	2025-01-07 14:56:53+03	true	13	16	X345XE78	XDL321098L0123456	[null]	VWEA2114TSI34567
3	244	13	12.19	3000	2025-01-03 05:56:17+03	true	13	16	X345XE78	XDL321098L0123456	[null]	VWEA2114TSI34567
4	202	13	12.6	500	2024-11-08 22:17:10+03	true	13	16	X345XE78	XDL321098L0123456	[null]	VWEA2114TSI34567
5	903	13	12.6	1500	2024-10-31 14:25:40+03	true	13	16	X345XE78	XDL321098L0123456	[null]	VWEA2114TSI34567

Рис 11 - результат выполнения запроса INSERT

### 2. UPDATE

Понизим скидку (если она есть) на 3% клиентам, которые целый год ничего не арендовали.

```
UPDATE client
SET discount=discount-0.03
WHERE discount IS NOT NULL and id NOT IN
(
    SELECT DISTINCT client.id
    FROM contract
    JOIN passport_history ON passport_id=passport_history.id
    JOIN client ON client.id=client_id
    WHERE contract.date_start > '2024-01-01'
)
```

	id [PK] integer	discount real
1	36	0.06

Рис 12 - до UPDATE

	id [PK] integer	discount real
1	36	0.03

Рис 13 - после UPDATE

### 3. DELETE

Удалим все авто, которых нет в компании по прокату.

```
DELETE FROM car
WHERE id NOT IN
(
    SELECT DISTINCT car.id
    FROM car
    JOIN car_instance ON car_instance.car_id=car.id
)
```

	id [PK] integer	engine_id integer	transmission transmission_type	car_body_id integer	fuel_consumption real	info_id integer
22	22	22	auto	20	12.3	8
23	23	23	auto	21	9.8	8
24	24	24	robotic	22	15	8
25	25	25	auto	23	7.4	9
26	26	26	manual	24	6.9	9
27	27	27	robotic	25	7.8	9
28	28	28	manual	26	6.1	10
29	29	29	auto	27	6.5	10
30	30	30	variator	28	5.8	10

Рис 14 - до DELETE

	id [PK] integer	engine_id integer	transmission transmission_type	car_body_id integer	fuel_consumption real	info_id integer
1	1	1	auto	1	7.8	1
2	3	3	variator	2	7.1	1
3	4	4	variator	3	6.2	2
4	7	7	auto	5	9.5	3
5	10	10	auto	8	6.8	4
6	13	13	auto	11	6.5	5
7	16	16	manual	14	5.9	6
8	19	19	auto	17	13.5	7
9	22	22	auto	20	12.3	8

Рис 15 - после DELETE

### 3 Создание индексов

```
EXPLAIN ANALYZE
SELECT * FROM contract
WHERE delay != 0;

CREATE INDEX indx_for_delay ON contract(delay);
```

Рассмотрим разницу в выполнении запросов с индексами и без.

	QUERY PLAN text
1	Seq Scan on contract (cost=0.00..35.29 rows=116 width=100) (actual time=0.017..0.168 rows=116 loops=...
2	Filter: (delay <> 0)
3	Rows Removed by Filter: 1107
4	Planning Time: 0.066 ms
5	Execution Time: 0.185 ms

Рис 16 - до создания INDEX

	QUERY PLAN text
1	Seq Scan on contract (cost=0.00..35.29 rows=116 width=100) (actual time=0.025..0.389 rows=116 loops=...
2	Filter: (delay <> 0)
3	Rows Removed by Filter: 1107
4	Planning Time: 0.081 ms
5	Execution Time: 0.409 ms

Рис 17 - после создания INDEX

## **Выводы**

В ходе выполнения данной лабораторной работы было создано множество различных запросов для базы данных, а также скорректированы данные. Кроме того, были написаны представления и индексы. Я узнал, что индексы не всегда ускоряют выполнение запросов и использовать их нужно только в некоторых случаях.