

**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»

Автор: Цыпандин А.П.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

## Оглавление

ЦЕЛЬ РАБОТЫ .....	3
ПРАКТИЧЕСКОЕ ЗАДАНИЕ .....	3
ВАРИАНТ 14. БД «СЛУЖБА ЗАКАЗА ТАКСИ».....	3
ВЫПОЛНЕНИЕ.....	3
Схема логической модели данных.....	5
Запросы к БД.....	5
ВЫВОД .....	7

## Цель работы

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

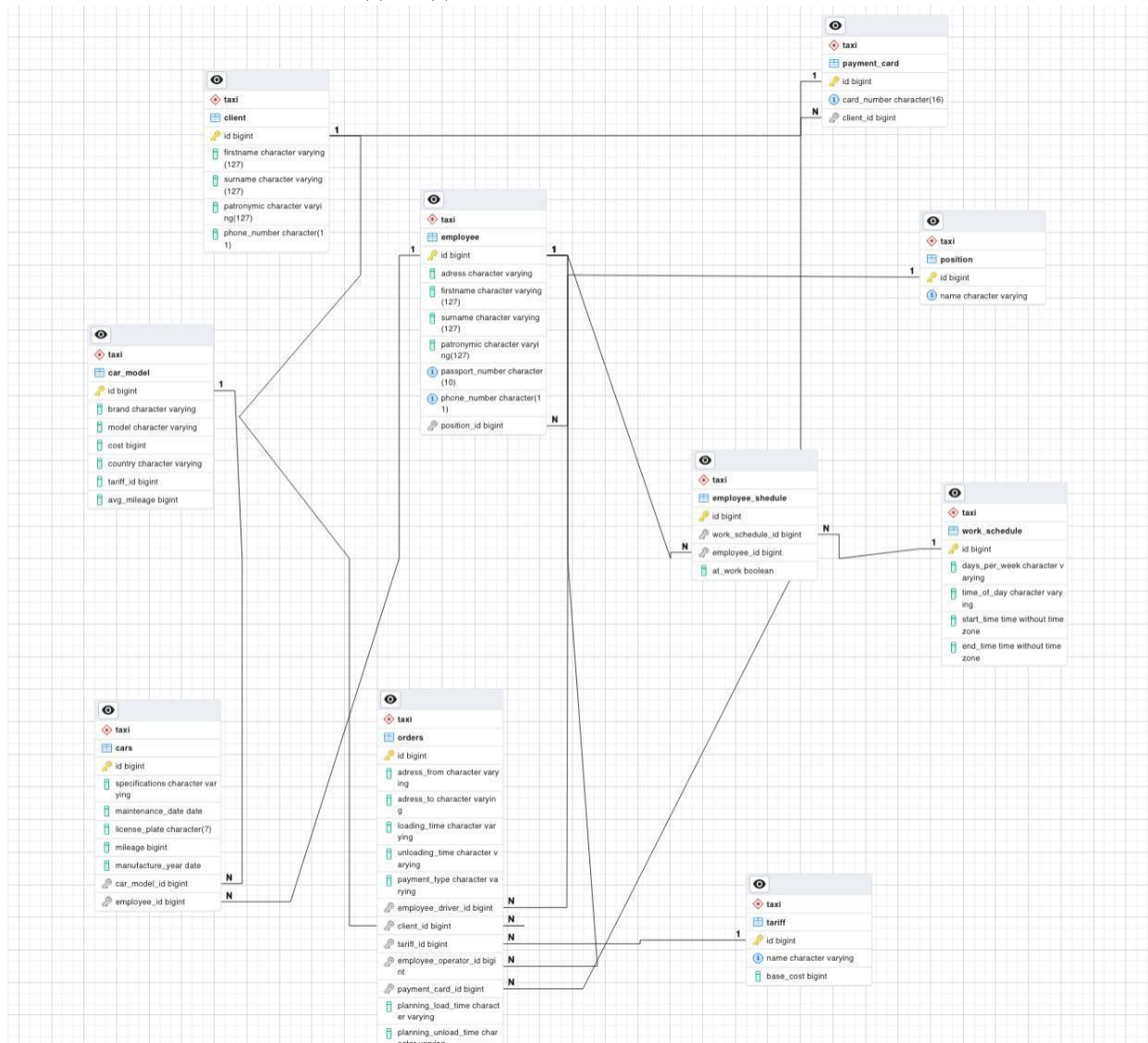
## Практическое задание

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, 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 работы

Название БД: Сервис для заказа такси.

Схема логической модели данных:



Запросы к базе данных:

1) Вывести данные о водителе, который чаще всех доставляет пассажиров на заданную улицу.

```
SELECT e.id, e.surname, e.firstname, e.passport_number, COUNT(o.id)
AS orders_count
FROM taxi.employee e
JOIN taxi.orders o ON o.employee_driver_id =
e.id WHERE o.address_to = 'ул. Тукая, д. 2'
GROUP BY e.id
HAVING COUNT(o.id) = (
    SELECT COUNT(o_inner.id)
    FROM taxi.employee e_inner
    JOIN taxi.orders o_inner ON o_inner.employee_driver_id =
e_inner.id WHERE o_inner.address_to = 'ул. Тукая, д. 2' GROUP BY
e_inner.id
ORDER BY COUNT(o_inner.id) DESC
LIMIT 1
)
ORDER BY orders_count DESC;
```

	id [PK] bigint	surname character varying (127)	firstname character varying (127)	passport_number character	orders_count bigint
1	4	Рыбаков	Юлий	2004250492	3
2	30	Моисеев	Твердислав	9543666304	3

2) Вывести данные об автомобилях, которые имеют пробег более 250 тысяч. километров и которые не проходили ТО в текущем году.

```
SELECT c.specifications, c.manufacture_year,
c.license_plate, c.maintenance_date, c.employee_id FROM
taxi.cars c
WHERE c.maintenance_date <= CURRENT_DATE - interval '1 year' AND c.mileage
> 250000;
```

	specifications character varying	manufacture_year date	license_plate character	maintenance_date date	employee_id bigint
1	Motor 1.8; Manual	2001-10-01	Я497ЧЗ	2020-10-09	1
2	Motor 2.2; Manual	2021-08-29	Я216ВЩ	2021-08-09	19
3	Motor 2.2; Manual	2005-06-15	Х687КИ	2021-11-24	22
4	Motor 2.0; Manual	2003-06-30	Щ959АЙ	2021-05-10	24

3) Сколько раз каждый пассажир воспользовался услугами таксопарка?

```
SELECT c.id, c.firstname, c.surname, COUNT(o.client_id)
AS orders_count FROM taxi.client c
LEFT JOIN taxi.orders o ON o.client_id = c.id
GROUP BY c.id
ORDER BY orders_count DESC;
```

	id [PK] bigint	firstname character varying (127)	surname character varying (127)	orders_count bigint
1	4	Марфа	Аркадьевна	3
2	100	Агап	Витальевич	3
3	37	Шарапова	Синклитикия	2
4	65	Клавдия	Николаевна	2
5	33	Аким	Демьянович	2
6	60	Кира	Архиповна	2
7	14	Лукьян	Власович	2
8	8	Кузнецов	Спиридон	2
9	2	Силина	Октябрина	2
10	105	Капустина	Синклитикия	1
11	45	Рогов	Капитон	1
12	107	Станислав	Вилорович	1
13	40	Елисей	Матвеевич	1
14	44	Антонина	Михайловна	1
15	30	Ксения	Никифоровна	1
16	24	Вышеслав	Филатович	1
17	39	Евламкий	Измаилович	1
18	153	Иван	Иванов	1
19	114	Григорьев	Эрнест	1
20	36	Юдина	Варвара	1
21	31	Жданова	Ульяна	1
22	17	Родионова	Валерия	1

4) Вывести данные пассажира, который воспользовался услугами таксопарка максимальное число раз.

```

SELECT c.id, c.firstname, c.surname, COUNT(o.client_id) AS
orders_count FROM taxi.client c
LEFT JOIN taxi.orders o ON o.client_id =
c.id GROUP BY c.id, c.firstname, c.surname
HAVING COUNT(o.client_id) = (
    SELECT COUNT(o1.client_id)
    FROM taxi.client c1
    LEFT JOIN taxi.orders o1 ON o1.client_id =
c1.id GROUP BY c1.id
    ORDER BY COUNT(o1.client_id) DESC
    LIMIT 1
)
ORDER BY orders_count DESC;

```

	id [PK] bigint	firstname character varying (127)	surname character varying (127)	orders_count bigint
1	4	Марфа	Аркадьевна	3
2	100	Агап	Витальевич	3

5) Вывести данные о водителе, который ездит на самом дорогом автомобиле

```
SELECT e.id, e.firstname, e.surname, e.phone_number, cm.cost AS
max_cost FROM taxi.employee e
JOIN taxi.cars c ON e.id = c.employee_id
JOIN taxi.car_model cm ON cm.id = c.car_model_id
GROUP BY e.id, cm.id
HAVING cm.cost = (
    SELECT MAX(cm1.cost)
    FROM taxi.car_model cm1
    JOIN taxi.cars c1 ON c1.car_model_id =
    cm1.id ORDER BY MAX(cm1.cost) DESC
    LIMIT 1
);
```

	id bigint	firstname character varying (127)	surname character varying (127)	phone_number character	max_cost bigint
1	19	Януарий	Елисеев	78484115254	4500000
2	30	Твердислав	Моисеев	71474032276	4500000

6) Вывести данные пассажира, который всегда ездит с одним и тем же водителем.

```
SELECT c.id, c.firstname, c.surname
FROM taxi.client c
JOIN taxi.orders o ON o.client_id = c.id
GROUP BY c.id, c.firstname, c.surname
HAVING COUNT(DISTINCT o.employee_driver_id) = 1;
```

	id [PK] bigint	firstname character varying (127)	surname character varying (127)
1	1	Богданов	Аверкий
2	4	Марфа	Аркадьевна
3	7	Ия	Викторовна
4	15	Таисия	Петровна
5	17	Родионова	Валерия
6	20	Станислав	Жоресович
7	23	Герасимова	Юлия
8	24	Вышеслав	Филатович
9	27	Пестов	Иннокентий
10	28	Орехова	Евдокия
11	30	Ксения	Никифоровна
12	31	Жданова	Ульяна
13	36	Юдина	Варвара
14	39	Евламий	Измаилович
15	40	Елисей	Матвеевич
16	44	Антонина	Михайловна
17	45	Рогов	Капитон
18	56	Жуков	Герман
19	57	Гедеон	Артемьевич

7) Какие автомобили имеют пробег больше среднего пробега для своей марки.

```
SELECT c.id, c.license_plate, c.specifications, c.employee_id
FROM taxi.cars c
JOIN taxi.car_model cm ON cm.id = c.car_model_id
WHERE c.mileage > cm.avg_mileage * EXTRACT(YEAR FROM
AGE(NOW(), c.maintenance_date));
```

	id [PK] bigint	license_plate character	specifications character varying	employee_id bigint
1	1	Я497ЧЗ	Motor 1.8l; Manual	1
2	2	Ш433ЖР	Motor 2.2l; Manual	2
3	3	П745ЛИ	Motor 2.2l; Manual	3
4	4	Р970СЖ	Motor 2.5l; Manual	4
5	6	К495ЫД	Motor 2.2l; Manual	20
6	7	А448ИХ	Motor 1.5l; Manual	21
7	8	Х687КИ	Motor 2.2l; Manual	22
8	9	Е768ЧЯ	Motor 1.6l; Manual	24
9	10	Ы540ЩШ	Motor 1.5l; Manual	23
10	11	Щ959АЙ	Motor 2.0l; Manual	24
11	12	Ж133ЧЭ	Motor 1.9l; Manual	25
12	13	Р970ЧО	Motor 2.1l; Manual	26
13	14	К161ВЭ	Motor 1.5l; Manual	27
14	15	Ш577РК	Motor 2.2l; Manual	28
15	17	Е648ХЗ	Motor 1.7l; Manual	30
16	16	З751ГО	Motor 1.9l; Manual	29
17	18	О184ГО	Motor 1.5, auto	32
18	5	Я216ВЩ	Motor 2.2l; Manual	19



Представления:

1) Содержащее сведения о незанятых на данный момент водителях

```
CREATE VIEW find_free_drivers AS
SELECT DISTINCT e.id, e.firstname, e.surname, e.phone_number
FROM taxi.employee e
JOIN taxi.employee_schedule es ON e.id = es.employee_id
JOIN taxi.orders o ON o.employee_driver_id = e.id
WHERE es.at_work IS TRUE AND o.status = 'в процессе';
```

	id [PK] bigint	firstname character varying (127)	surname character varying (127)	phone_number character
1	1	Любовь	Потапова	77539137931
2	2	Лаврентий	Колобов	77332695886

2) Зарплата всех водителей за вчерашний день.

```
CREATE VIEW find_drivers_salary AS
SELECT e.id,
       e.firstname,
       e.surname,
       e.phone_number,
       p.salary_per_hour * EXTRACT(EPOCH FROM (end_time - start_time)) /
3600 AS salary
FROM taxi.employee e
JOIN taxi.employee_working_shift ews ON ews.driver_id = e.id
JOIN taxi.position p ON p.id = e.position_id
GROUP BY e.id, p.id, ews.id
HAVING p.salary_per_hour * EXTRACT(EPOCH FROM (end_time - start_time)) /
3600 IS NOT NULL
ORDER BY salary DESC;
```

	id [PK] bigint	firstname character varying (127)	surname character varying (127)	phone_number character	salary numeric
1	29	Август	Потапова	79840212470	3200.0000000000000000
2	28	Лука	Киселев	79651988842	2880.0000000000000000
3	4	Юлий	Рыбаков	71117947185	2880.0000000000000000
4	23	Осип	Поляков	79127460157	2880.0000000000000000
5	26	Эрнст	Егорова	77531443655	2880.0000000000000000
6	22	Вышеслав	Меркушев	73799329146	2880.0000000000000000
7	25	Евфросиния	Самойлова	79266220398	2880.0000000000000000
8	1	Любовь	Потапова	77539137931	2880.0000000000000000
9	24	Эдуард	Наумова	78255276627	2880.0000000000000000
10	2	Лаврентий	Колобов	77332695886	2880.0000000000000000
11	27	Владлен	Колобова	75110255942	2880.0000000000000000



Подзапросы:

1) Запрос, который удаляет тех клиентов, у которых не привязана карта оплаты, нет ни одного заказа или заказы которые оплачены наличным расчётом.

```
DELETE FROM taxi.client c
WHERE NOT EXISTS (
    SELECT 1
    FROM taxi.orders o
    WHERE o.client_id = c.id OR o.payment_card_id IS null
)
AND NOT EXISTS (
    SELECT 1
    FROM taxi.payment_card pc
    WHERE pc.client_id = c.id
);
```

Клиенты которые подходят под описание под индексом 150, 151, 152:

144	147	Щербаков	Всеслав	Валерьянович	76423468110
145	148	Афанасьев	Евгений	Валентинович	72154751678
146	149	Носкова	Эмилия	Эльдаровна	77232086108
147	150	Василий	Удаленко	Иосифович	89124238712
148	151	Петр	Удаленков	Валерьевич	89124238412
149	152	Петр	Удаленков	Петрович	89124271632
150	153	Иван	Иванов	Кириллов	89124295544

После выполнения запроса:

142	145	Алина	Федоровна	Рыбакова	76359694742
143	146	Королев	Никандр	Власович	76914077824
144	147	Щербаков	Всеслав	Валерьянович	76423468110
145	148	Афанасьев	Евгений	Валентинович	72154751678
146	149	Носкова	Эмилия	Эльдаровна	77232086108
147	153	Иван	Иванов	Кириллов	89124295544

2) Запрос, который добавляет для сотрудника – водителя автомобиль по заданным параметрам

```
WITH inserted_car_model AS (
    INSERT INTO taxi.car_model (brand, model, cost, country, tariff_id,
    avg_mileage)
    VALUES ('Honda', 'Fit', '450000', 'Japan', 1, 15000)
    RETURNING id
)
```

```
INSERT INTO taxi.cars (id, specifications, maintenance_date,
license_plate, mileage, manufacture_year, car_model_id, employee_id)
```

```
SELECT
    18,
    'Motor 1.5, auto',
    '2023-11-15',
    'A413ГД',
    111034,
    '2017-02-15',
    icm.id,
```

```

        e.id
FROM taxi.employee e
JOIN inserted_car_model icm ON 1 = 1
WHERE
        e.firstname || ' ' || e.surname || ' ' || e.patronymic = 'Василий
Петров Петрович'
        AND passport_number = '9818765455'
        AND e.position_id = 1;

```

Таблица cars и car\_model до запроса:

13	13	Motor 2.1; Manual	2022-11-20	P97040	192667	2000-05-13	6	26
14	14	Motor 1.5; Manual	2021-08-18	K161B9	105459	2011-01-19	7	27
15	15	Motor 2.2; Manual	2021-08-25	Ш577PK	244624	2012-05-17	7	28
16	16	Motor 1.9; Manual	2020-03-25	3751ГО	229424	2017-11-26	8	29
17	17	Motor 1.7; Manual	2021-08-22	E648X3	248532	2000-01-05	9	30
5	5	VAZ	21083	240000	Russia	1	10000	
6	6	toyota	celica	650000	Japan	1	16000	
7	7	Nissan	SkyLine	450000	Japan	1	15000	
8	8	peugot	206	600000	France	1	18000	
9	9	mercedes	s-class	4500000	Germany	2	14500	

Таблица cars и car\_model после запроса, а также таблица employee:

16	16	Motor 1.9; Manual	2020-03-25	3751ГО	229424	2017-11-26	8	29
17	17	Motor 1.7; Manual	2021-08-22	E648X3	248532	2000-01-05	9	30
18	18	Motor 1.5, auto	2023-11-15	A413ГД	111034	2017-02-15	10	32

8	8	peugot	206	600000	France	1	18000
9	9	mercedes	s-class	4500000	Germany	2	14500
10	10	Honda	Fit	450000	Japan	1	15000

27	27	наб. Лазо, д. 7/7 стр. 5/8, 50	Владлен	Колобова	Геннадиевич	2497237661	75110255942	1
28	28	алл. Шевченко, д. 4 стр. 67, 17	Лука	Киселев	Кузьминична	6079770925	79651988842	1
29	29	пер. Первомайский, д. 60 стр. 707, 92	Август	Потапова	Горбачева	7553613986	79840212470	2
30	30	наб. Заовражная, д. 3 стр. 3/3, 63	Твердислав	Моисеев	Павлова	9543666304	71474032276	2
31	32	Краснодыбская 1/2, кв.7	Василий	Петров	Петрович	9818765455	78645632451	1

Total rows: 31 of 31 Query complete 00:00:00 137

3) Запрос, который обновляет дату последнего ТО на новое, для автомобиля, который привязан к сотруднику с заданными данными

```

UPDATE taxi.cars
SET maintenance_date = '2024.01.10', license_plate =
'O184ГО' WHERE employee_id = (
        SELECT id
        FROM taxi.employee e
        WHERE CONCAT(e.firstname, ' ', e.surname, ' ', e.patronymic) =
'Василий Петров Петрович'
        AND e.passport_number = '9818765455')
AND car_model_id = (
        SELECT id
        FROM taxi.car_model cm
        WHERE cm.brand = 'Honda'

```

```
AND cm.model = 'Fit');
```

Таблица **cars** до запроса:

15	15	Motor 2.2i; Manual	2021-08-25	Ш577PK	244624	2012-05-17	7	28
16	16	Motor 1.9i; Manual	2020-03-25	3751ГO	229424	2017-11-26	8	29
17	17	Motor 1.7i; Manual	2021-08-22	E648X3	248532	2000-01-05	9	30
18	18	Motor 1.5, auto	2023-11-15	A413ГД	111034	2017-02-15	10	32

Таблица **cars** и **car\_model** после запроса:

16	16	Motor 1.9i; Manual	2020-03-25	3751ГO	229424	2017-11-26	8	29
17	17	Motor 1.7i; Manual	2021-08-22	E648X3	248532	2000-01-05	9	30
18	18	Motor 1.5, auto	2024-01-10	O184ГO	111034	2017-02-15	10	32

Индексы:

Тестовый запрос и план запроса:

```
EXPLAIN ANALYZE SELECT
```

```
o.id,  
o.adress_to,  
o.adress_from,  
o.loading_time,  
o.unloading_time,  
c.firstname,  
c.surname,  
e.firstname,  
e.surname,  
t.name
```

```
FROM taxi.orders o
```

```
JOIN taxi.client c ON o.client_id = c.id
```

```
JOIN taxi.tariff t ON o.tariff_id = t.id
```

```
JOIN taxi.employee e ON o.employee_driver_id =  
e.id WHERE e.position_id = 1;
```

	QUERY PLAN	
	text	
1	Hash Join (cost=7.56..13.18 rows=23 width=242) (actual time=0.237..0.368 rows=31 loops=1)	
2	Hash Cond: (o.tariff_id = t.id)	
3	-> Hash Join (cost=6.52..11.94 rows=23 width=218) (actual time=0.207..0.323 rows=31 loops=1)	
4	Hash Cond: (c.id = o.client_id)	
5	-> Seq Scan on client c (cost=0.00..4.46 rows=146 width=39) (actual time=0.012..0.041 rows=147 loops=1)	
6	-> Hash (cost=6.23..6.23 rows=23 width=195) (actual time=0.146..0.148 rows=31 loops=1)	
7	Buckets: 1024 Batches: 1 Memory Usage: 14kB	
8	-> Hash Join (cost=2.56..6.23 rows=23 width=195) (actual time=0.070..0.113 rows=31 loops=1)	
9	Hash Cond: (o.employee_driver_id = e.id)	
10	-> Seq Scan on orders o (cost=0.00..3.51 rows=51 width=174) (actual time=0.013..0.037 rows=35 loops=1)	
11	-> Hash (cost=2.39..2.39 rows=14 width=37) (actual time=0.033..0.034 rows=15 loops=1)	
12	Buckets: 1024 Batches: 1 Memory Usage: 10kB	
13	-> Seq Scan on employee e (cost=0.00..2.39 rows=14 width=37) (actual time=0.007..0.019 rows=15 loop...	
14	Filter: (position_id = 1)	
15	Rows Removed by Filter: 16	
16	-> Hash (cost=1.02..1.02 rows=2 width=40) (actual time=0.022..0.022 rows=2 loops=1)	
17	Buckets: 1024 Batches: 1 Memory Usage: 9kB	
18	-> Seq Scan on tariff t (cost=0.00..1.02 rows=2 width=40) (actual time=0.015..0.017 rows=2 loops=1)	
19	Planning Time: 2.469 ms	
20	Execution Time: 0.463 ms	

### Создание индексов и план запроса с индексами:

```
CREATE INDEX idx_firstname ON taxi.employee(firstname);  
CREATE INDEX idx_surname ON taxi.employee(surname);  
CREATE INDEX idx_phone_number ON taxi.employee(phone);
```

### Составные индексы:


#### Тестовый запрос и план запроса:

```
EXPLAIN ANALYZE SELECT o.id, o.adress_from, o.adress_to  
FROM taxi.orders o  
JOIN taxi.employee e ON o.employee_driver_id = o.id  
JOIN taxi.client c ON c.id = o.client_id  
WHERE e.position_id = 1  
GROUP BY o.id;
```

	QUERY PLAN	
	text	
1	HashAggregate (cost=10.22..10.23 rows=1 width=83) (actual time=0.054..0.055 rows=1 loops=1)	
2	Group Key: o.id	
3	Batches: 1 Memory Usage: 24kB	
4	-> Nested Loop (cost=0.14..10.18 rows=14 width=83) (actual time=0.033..0.045 rows=15 loops=1)	
5	-> Nested Loop (cost=0.14..7.65 rows=1 width=83) (actual time=0.027..0.030 rows=1 loops=1)	
6	-> Seq Scan on orders o (cost=0.00..3.44 rows=1 width=91) (actual time=0.021..0.023 rows=1 loops=1)	
7	Filter: (employee_driver_id = id)	
8	Rows Removed by Filter: 53	
9	-> Index Only Scan using client_pkey on client c (cost=0.14..4.16 rows=1 width=8) (actual time=0.005..0.005 rows=1 loop...	
10	Index Cond: (id = o.client_id)	
11	Heap Fetches: 1	
12	-> Seq Scan on employee e (cost=0.00..2.39 rows=14 width=0) (actual time=0.005..0.012 rows=15 loops=1)	
13	Filter: (position_id = 1)	
14	Rows Removed by Filter: 16	
15	Planning Time: 0.346 ms	
16	Execution Time: 0.110 ms	

#### Создание индексов и повторный запрос:

```
CREATE INDEX idx_employee_id_position_id ON taxi.employee(id,  
position_id); CREATE INDEX idx_client_id ON taxi.client(id);  
CREATE INDEX idx_orders_client_id ON taxi.orders (client_id);
```

	QUERY PLAN text	
1	HashAggregate (cost=11.06..11.07 rows=1 width=83) (actual time=0.045..0.046 rows=1 loops=1)	
2	Group Key: o.id	
3	Batches: 1 Memory Usage: 24kB	
4	-> Nested Loop (cost=3.45..11.02 rows=14 width=83) (actual time=0.023..0.042 rows=15 loops=1)	
5	-> Hash Join (cost=3.45..8.50 rows=1 width=83) (actual time=0.019..0.034 rows=1 loops=1)	
6	Hash Cond: (c.id = o.client_id)	
7	-> Seq Scan on client c (cost=0.00..4.48 rows=148 width=8) (actual time=0.004..0.011 rows=148 loops=1)	
8	-> Hash (cost=3.44..3.44 rows=1 width=91) (actual time=0.010..0.010 rows=1 loops=1)	
9	Buckets: 1024 Batches: 1 Memory Usage: 9kB	
10	-> Seq Scan on orders o (cost=0.00..3.44 rows=1 width=91) (actual time=0.007..0.008 rows=1 loops=1)	
11	Filter: (employee_driver_id = id)	
12	Rows Removed by Filter: 53	
13	-> Seq Scan on employee e (cost=0.00..2.39 rows=14 width=0) (actual time=0.003..0.006 rows=15 loops=1)	
14	Filter: (position_id = 1)	
15	Rows Removed by Filter: 16	
16	Planning Time: 0.163 ms	
17	Execution Time: 0.064 ms	

## Вывод

В ходе выполнения данной лабораторной работы мы рассмотрели использование запросов SELECT, UPDATE и DELETE с применением подзапросов в контексте базы данных с несколькими связанными таблицами. Также, в ходе лабораторной работы были рассмотрены индексы и их влияние на производительность запросов. Индексы были созданы для полей, используемых в условиях JOIN, чтобы ускорить поиск и сопоставление данных. В результате использования индексов мы получили улучшение производительности запросов. При работе с большими объемами данных, отличие было бы еще больше.