**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №3.2 «Создание таблиц базы данных PostgreSQL. Заполнение таблиц рабочими данными»

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

Автор: Корниенко М.Ю.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**Оглавление**

[Цель работы 3](#_Toc148889485)

[Практическое задание 3](#_Toc148889487)

[Вариант 12. БД «Прокат автомобилей» 3](#_Toc148889495)

[Рисунок 1 – Схема логической модели базы данных. 4](#_Toc148889496)

[Листинг дампа 5](#_Toc148889497)

[Вывод 19](#_Toc148889498)

**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.
5. Запросы к БД.

Какой автомобиль находился в прокате максимальное количество часов?

Листинг:

**WITH** TotalHours **AS** (

**SELECT**

r.car\_id,

m.brand **AS** model\_brand,

**SUM**(**EXTRACT**(EPOCH **FROM** (r.end\_date - r.start\_date)) / 3600) **AS** total\_hours

**FROM** contracts r

**INNER** **JOIN** cars **c** **ON** r.car\_id = **c**.id

**INNER** **JOIN** models m **ON** **c**.model\_id = m.id

**GROUP** **BY** r.car\_id, m.brand

)

**SELECT** th.car\_id, th.model\_brand, th.total\_hours

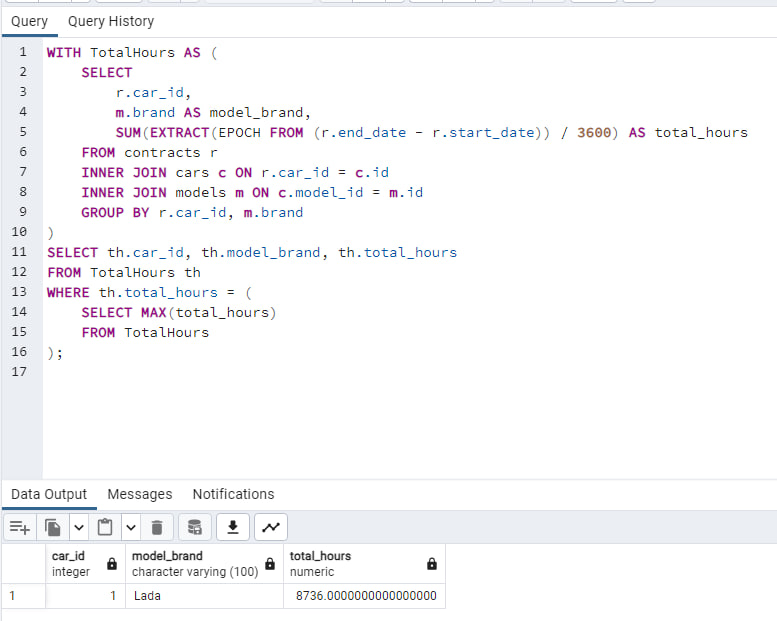
**FROM** TotalHours th

**WHERE** th.total\_hours = (

**SELECT** **MAX**(total\_hours)

**FROM** TotalHours

);



Автомобили какой марки чаще всего брались в прокат?

Листинг:

**WITH** RentalCounts **AS** (

**SELECT**

m.brand **AS** model\_brand,

**COUNT**(**c**.id) **AS** rental\_count

**FROM** contracts ct

**INNER** **JOIN** cars **c** **ON** ct.car\_id = **c**.id

**INNER** **JOIN** models m **ON** **c**.model\_id = m.id

**GROUP** **BY** m.brand

)

**SELECT** rc.model\_brand, rc.rental\_count

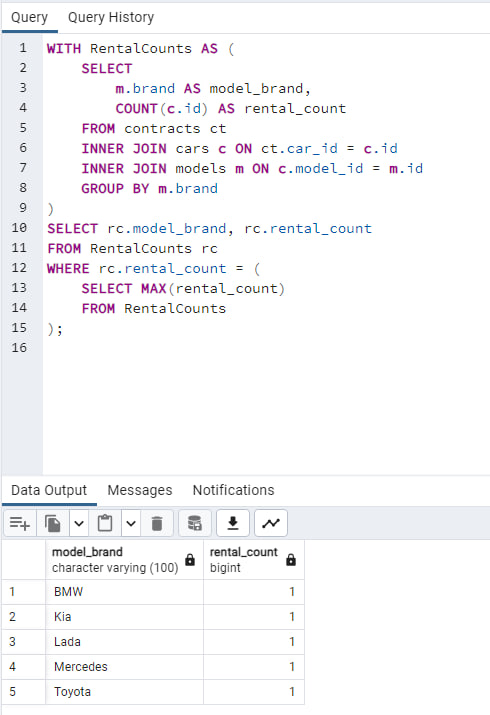
**FROM** RentalCounts rc

**WHERE** rc.rental\_count = (

**SELECT** **MAX**(rental\_count)

**FROM** RentalCounts

);



Определить убытки от простоя автомобилей за вчерашний день.

Листинг:

**WITH** DowntimePeriods **AS** (

**SELECT**

car.id **AS** car\_id,

models.id **AS** model\_id,

models.price **AS** price,

GREATEST(contracts.end\_date, NOW() - INTERVAL '1 day') **AS** end\_date,

COALESCE(

LEAD(contracts.start\_date) OVER (PARTITION **BY** car.id **ORDER** **BY** contracts.start\_date),

NOW()

) **AS** next\_start\_date

**FROM** cars car

**LEFT** **JOIN** contracts **ON** car.id = contracts.car\_id

**LEFT** **JOIN** models **ON** car.model\_id = models.id

),

DowntimeCalculations **AS** (

**SELECT**

dp.car\_id,

dp.model\_id,

dp.price,

**EXTRACT**(EPOCH **FROM** (dp.next\_start\_date - dp.end\_date)) / 3600 **AS** downtime\_hours

**FROM** DowntimePeriods dp

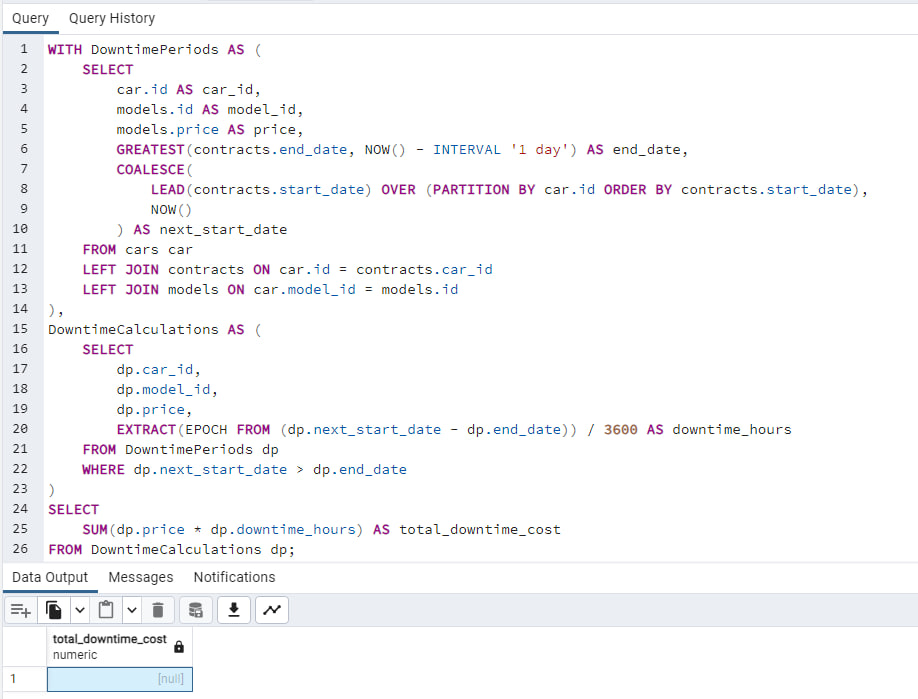
**WHERE** dp.next\_start\_date > dp.end\_date

)

**SELECT**

**SUM**(dp.price \* dp.downtime\_hours) **AS** total\_downtime\_cost

**FROM** DowntimeCalculations dp;



Вывести данные автомобиля, имеющего максимальный пробег.

Листинг:

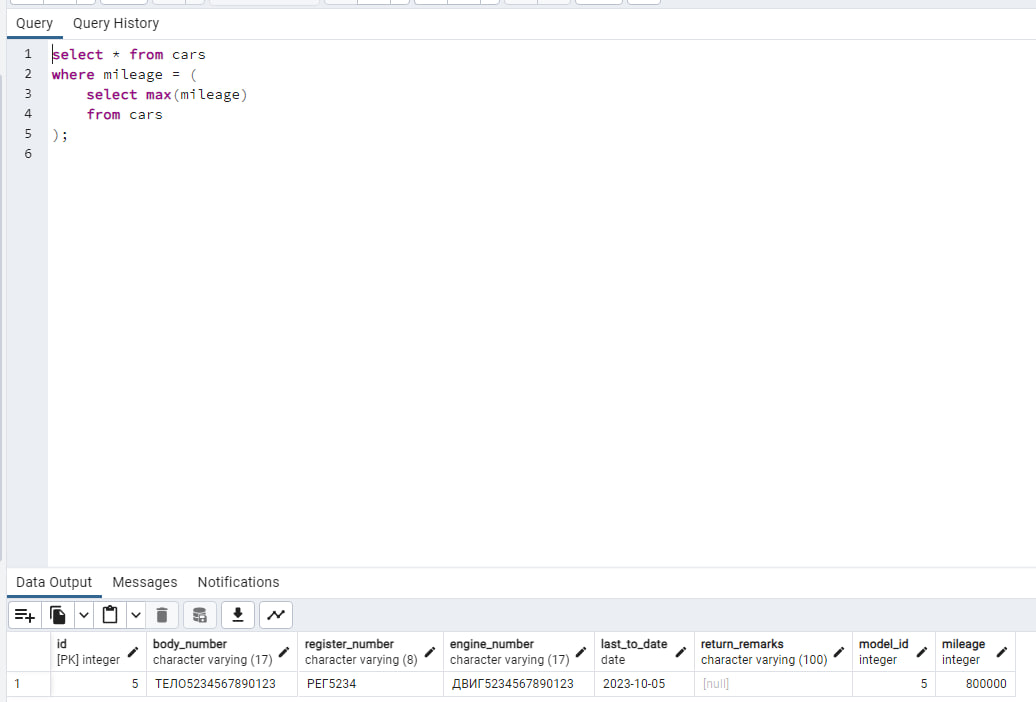
**select** \* **from** cars

**where** mileage = (

**select** **max**(mileage)

**from** cars

);



Какой автомобиль суммарно находился в прокате дольше всех.

Листинг:

**select** cars.\*

**from** contracts

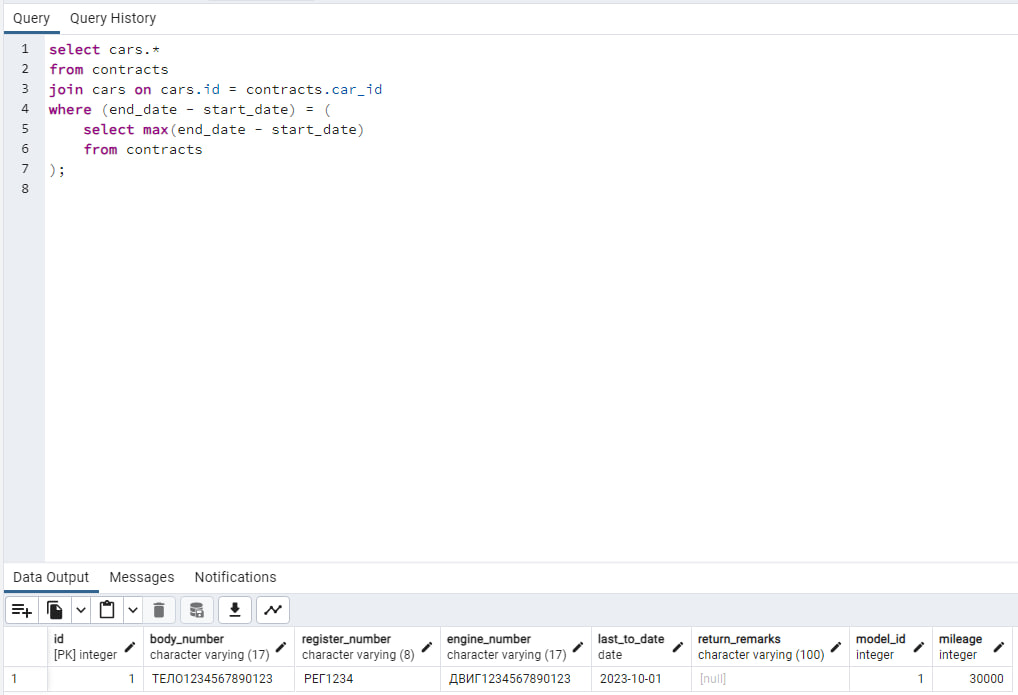
**join** cars **on** cars.id = contracts.car\_id

**where** (end\_date - start\_date) = (

**select** **max**(end\_date - start\_date)

**from** contracts

);



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

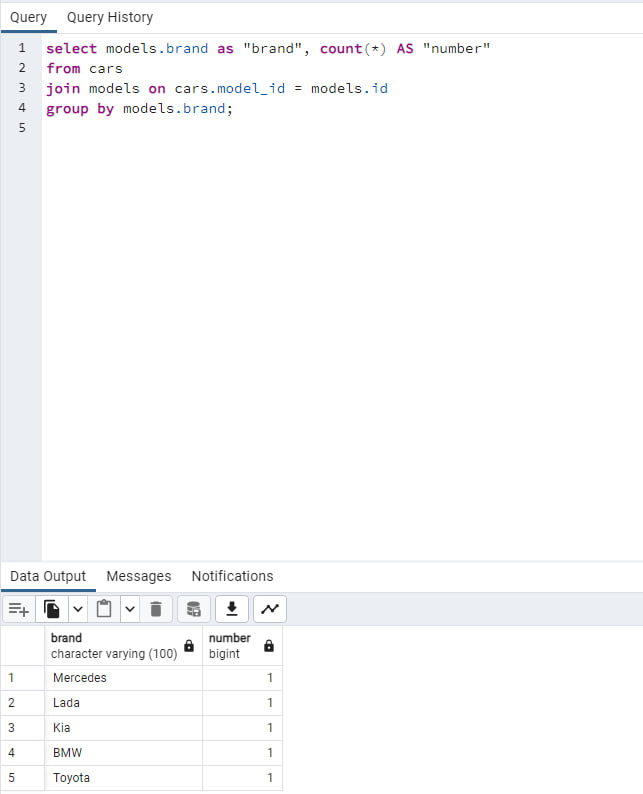
Листинг:

**select** models.brand **as** "brand", **count**(\*) **AS** "number"

**from** cars

**join** models **on** cars.model\_id = models.id

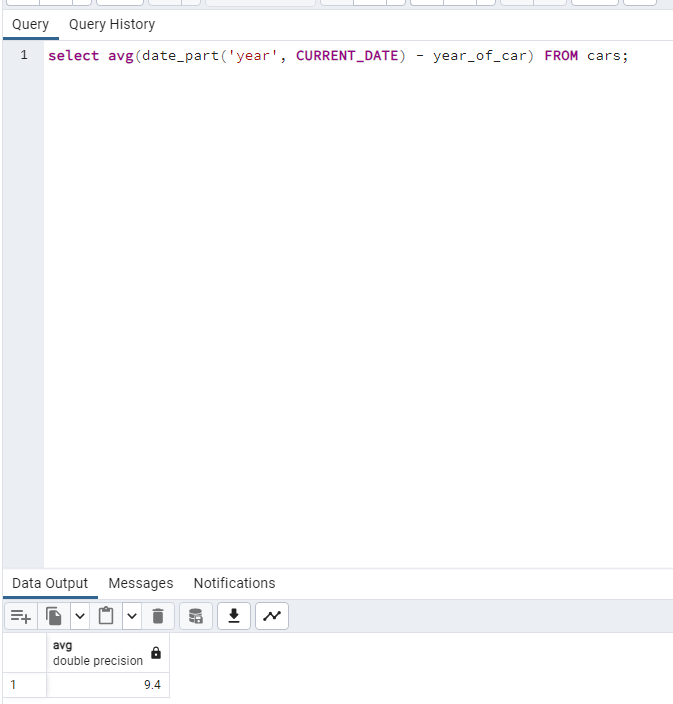
**group** **by** models.brand;



Определить средний “возраст” автомобилей компании.

Листинг:

**select** **avg**(date\_part('year', **CURRENT\_DATE**) - year\_of\_car) **FROM** cars;



1. Создание представлений:

Какой автомобиль ни разу не был в прокате?

Листинг:

**CREATE** **VIEW** CarsNotInRent **AS**

**SELECT** **c**.\*

**FROM** cars **c**

**WHERE** **NOT** **EXISTS** (

**SELECT** 1

**FROM** contracts contr

**WHERE** contr.car\_id = **c**.id

);



Вывести данные клиентов, не вернувших автомобиль вовремя.

Листинг:

**CREATE** **VIEW** ExpiredClients **AS**

**SELECT**

cl.id **AS** client\_id,

cl.mobile\_phone **AS** phone,

ct.end\_date **AS** enddate

**FROM**

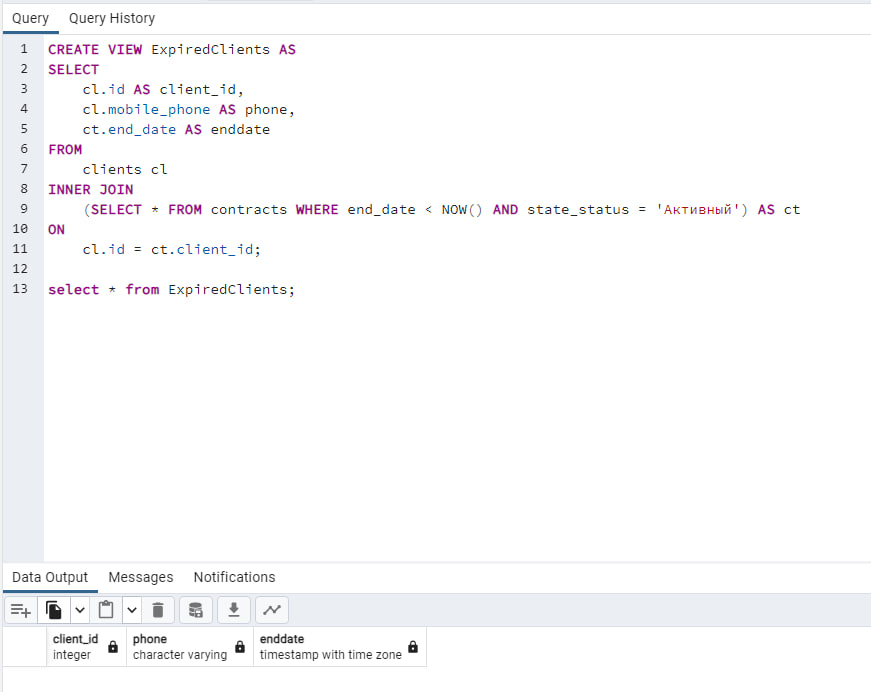
clients cl

**INNER** **JOIN**

(**SELECT** \* **FROM** contracts **WHERE** end\_date < NOW() **AND** state\_status = 'Активный') **AS** ct

**ON**

cl.id = ct.client\_id;



1. Запросы на модификацию данных:

Добавление работника на должность с наибольшим количеством работников

Листинг:

**INSERT** **INTO** workers (id, name, passport\_data, position\_id)

**SELECT** 6, 'Виталий Витальев', 12345678, (

**SELECT** p.id

**FROM** positions p

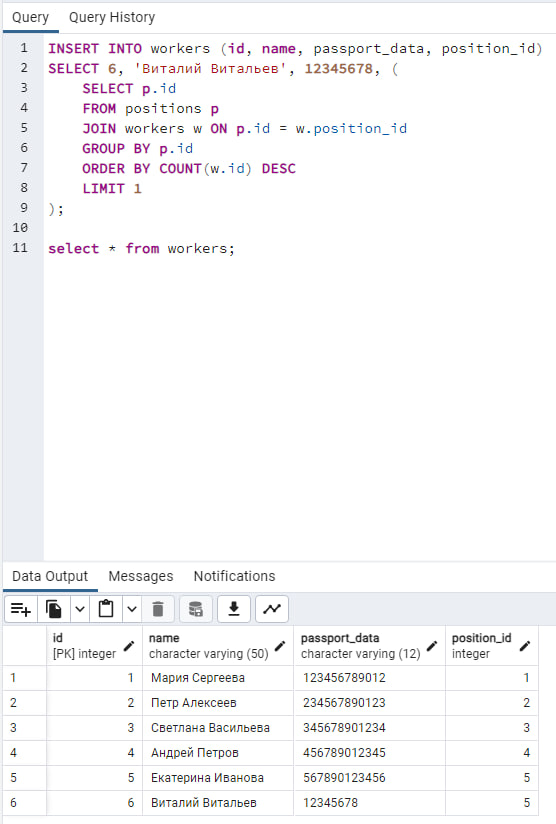
**JOIN** workers w **ON** p.id = w.position\_id

**GROUP** **BY** p.id

**ORDER** **BY** **COUNT**(w.id) **DESC**

**LIMIT** 1

);



Повышение зарплаты для должности, на которой суммарно получают меньше всего

Листинг:

**UPDATE** positions

**SET** salary = salary \* 1.05

**WHERE** id = (

**SELECT** p.id

**FROM** positions p

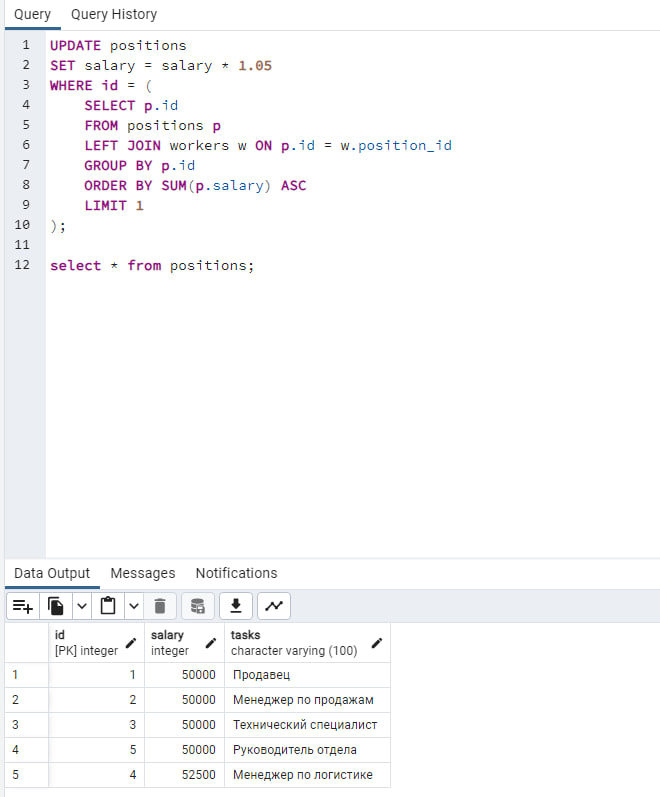
**LEFT** **JOIN** workers w **ON** p.id = w.position\_id

**GROUP** **BY** p.id

**ORDER** **BY** **SUM**(p.salary) **ASC**

**LIMIT** 1

);



Удаление неактивных контрактов с конца которых прошло более 1 года

**DELETE** **FROM** contracts

**WHERE** client\_id **IN** (

**SELECT** cl.id

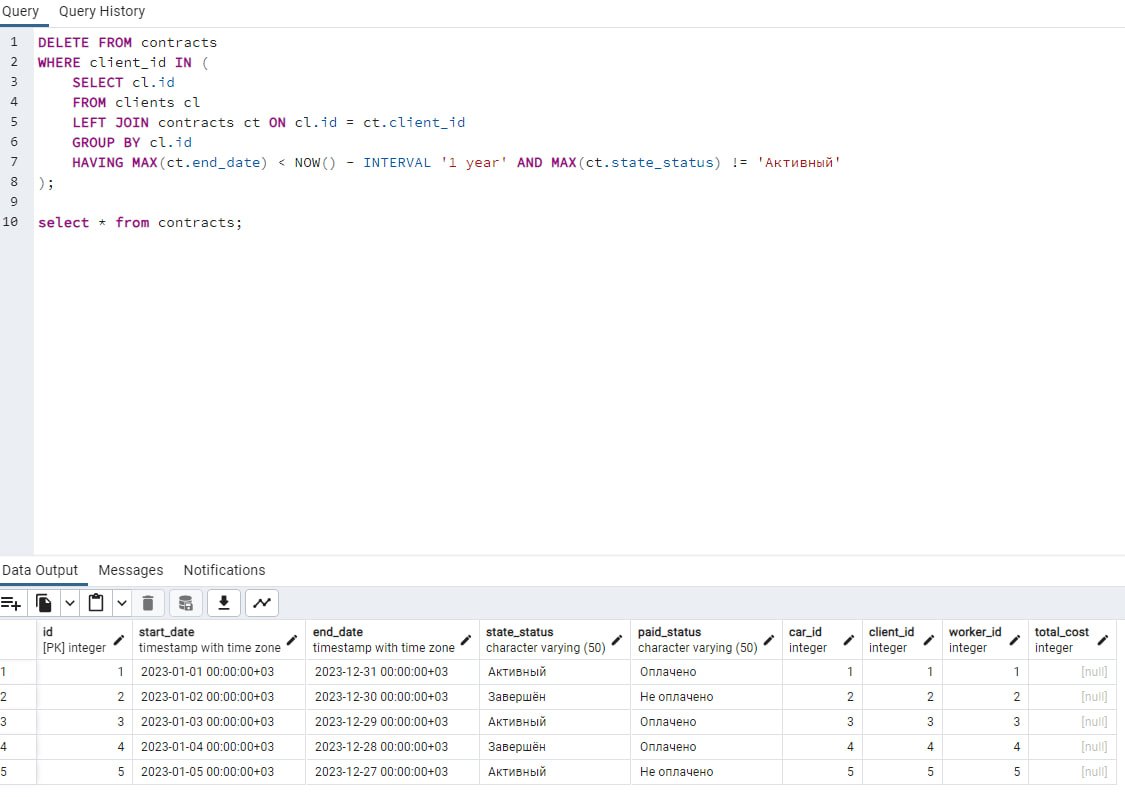
**FROM** clients cl

**LEFT** **JOIN** contracts ct **ON** cl.id = ct.client\_id

**GROUP** **BY** cl.id

**HAVING** **MAX**(ct.end\_date) < NOW() - INTERVAL '1 year' **AND** **MAX**(ct.state\_status) != 'Активный'

);

Создание индексов

Листинг:

**CREATE** **INDEX** idx\_cars\_model\_id **ON** cars(model\_id);

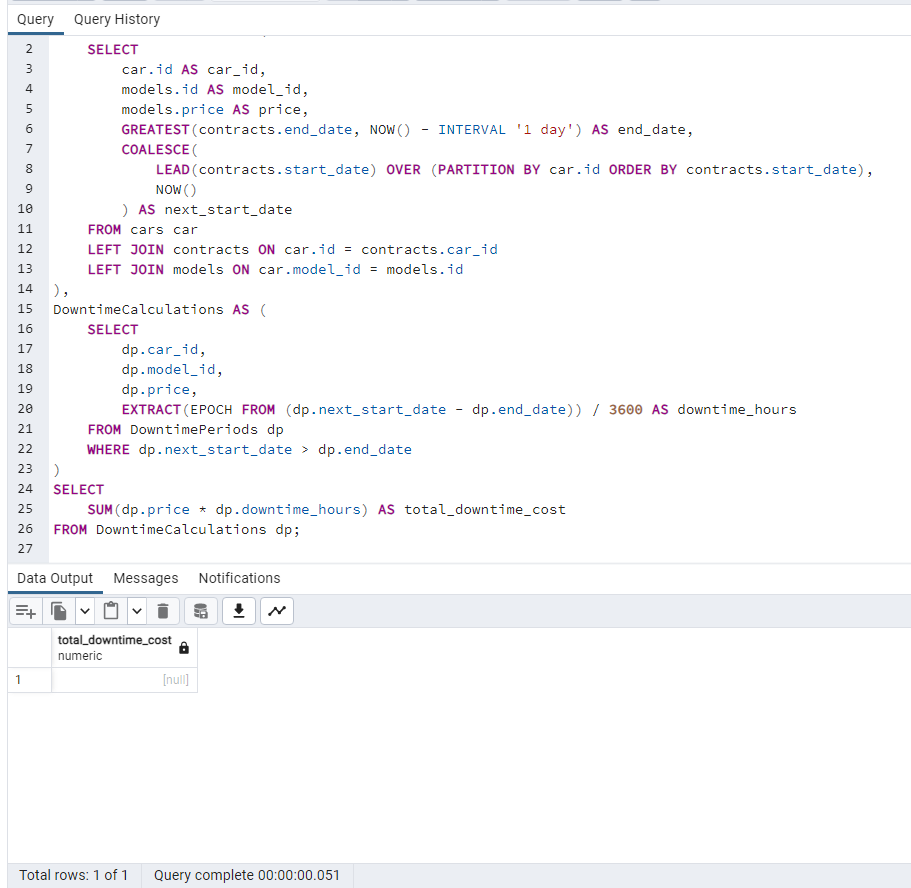
**CREATE** **INDEX** idx\_contracts\_id\_end\_date **ON** contracts(id, end\_date);

**CREATE** **INDEX** idx\_workers\_name **ON** workers(name);

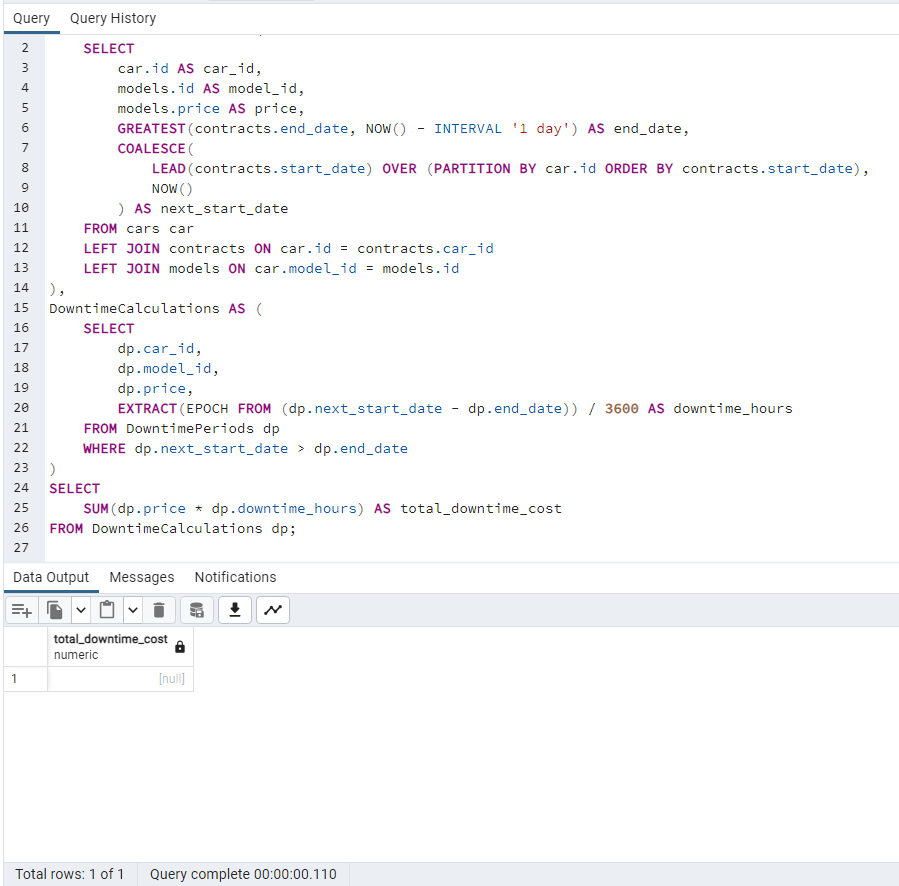
**CREATE** **INDEX** idx\_workers\_name\_position\_id **ON** workers(name, position\_id);

Сравнение времени работы с индексами и без:

1. С индексами



1. Без индексов



# Вывод

Во время лабораторной работы я научился использовать разнообразные SQL-запросы для взаимодействия с базой данных, а также узнал, как создавать представления и индексы. Кроме того, я провёл сравнение времени выполнения SELECT-запросов с использованием индексов и без них. Как оказалось, наличие индексов сокращает время выполнения запросов.