# Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования

# «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

**Отчет**

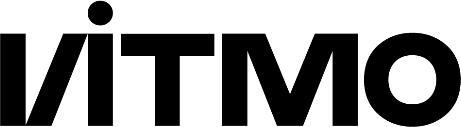
по лабораторной работе №4 «Запросы на выборку и модификацию данных. Представления. Работа с индексами»

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

Автор: Андрей Федак

Факультет: ИКТ Группа: К3239

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

# Оглавление

[Цель работы и практическое задание 3](#_bookmark0)

1. [Запросы к БД 4](#_bookmark1)
2. [Создание представлений 9](#_bookmark2)
3. [Запросы на модификацию данных 12](#_bookmark3)

[Вывод 16](#_bookmark4)

# Цель работы и практическое задание

**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.

1. Запросы к БД
   * Определить расчетное время полета по всем маршрутам. Листинг:

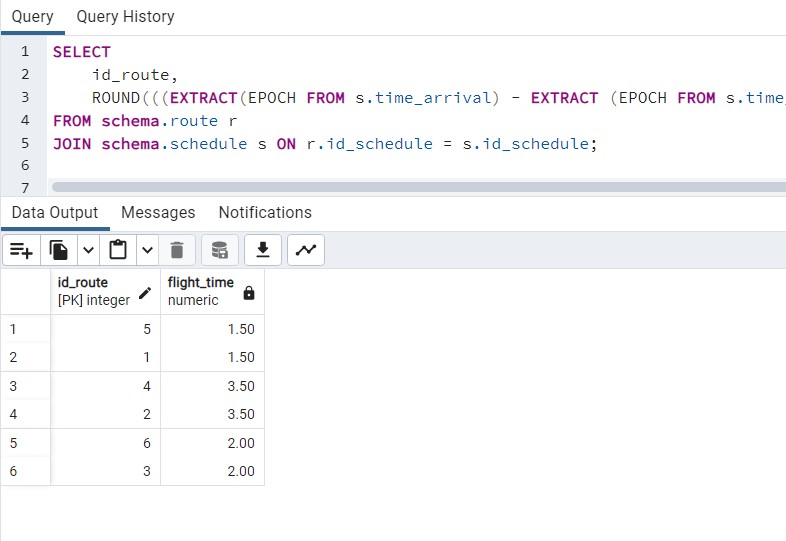
SELECT

id\_route,

ROUND(((EXTRACT(EPOCH FROM s.time\_arrival) - EXTRACT (EPOCH FROM

s.time\_departure))/3600),2) AS flight\_time FROM schema.route r

JOIN schema.schedule s ON r.id\_schedule = s.id\_schedule;



* + Определить расход топлива по всем маршрутам. Листинг:

SELECT r.id\_route,

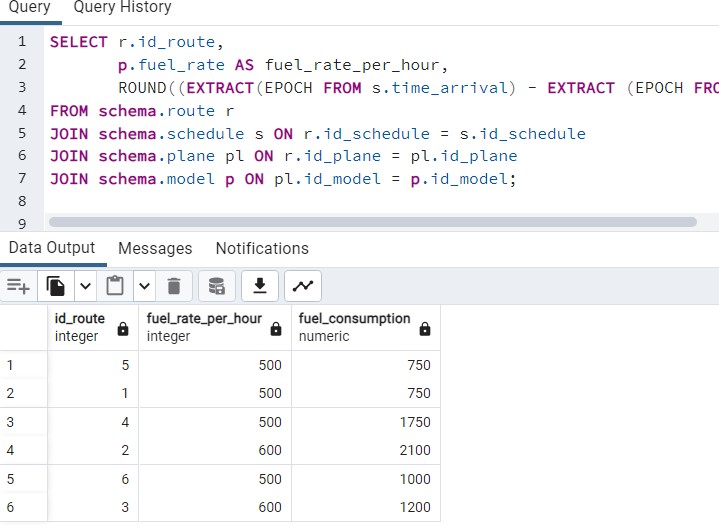
p.fuel\_rate AS fuel\_rate\_per\_hour,

ROUND((EXTRACT(EPOCH FROM s.time\_arrival) - EXTRACT (EPOCH FROM

s.time\_departure))/3600 \* p.fuel\_rate,0) AS fuel\_consumption FROM schema.route r

JOIN schema.schedule s ON r.id\_schedule = s.id\_schedule JOIN schema.plane pl ON r.id\_plane = pl.id\_plane

JOIN schema.model p ON pl.id\_model = p.id\_model;



* + Вывести данные о том, сколько свободных мест оставалось в самолетах, совершавших полет по заданному маршрутe за вчерашний день.

Листинг:

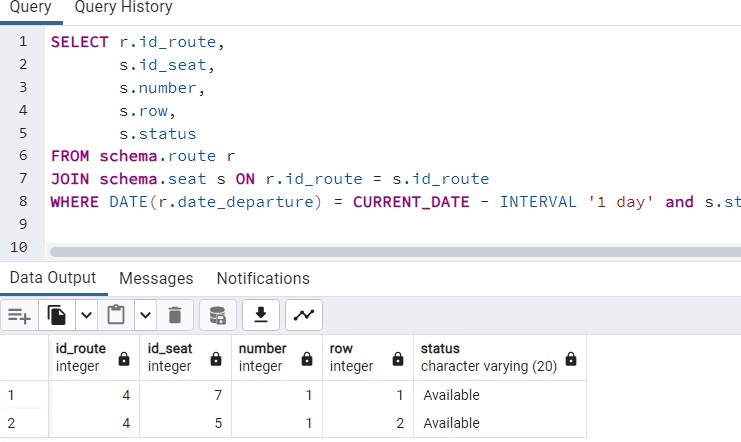
SELECT r.id\_route, s.id\_seat, s.number, s.row,

s.status

FROM schema.route r

JOIN schema.seat s ON r.id\_route = s.id\_route

WHERE DATE(r.date\_departure) = CURRENT\_DATE - INTERVAL '1 day' and s.status = 'Available' AND r.id\_schedule = 2;



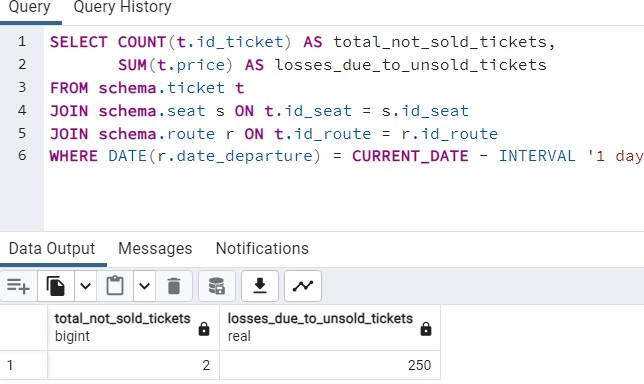
* + Рассчитать убытки компании за счет непроданных билетов за вчерашний день.

SELECT COUNT(t.id\_ticket) AS total\_not\_sold\_tickets, SUM(t.price) AS losses\_due\_to\_unsold\_tickets

FROM schema.ticket t

JOIN schema.seat s ON t.id\_seat = s.id\_seat JOIN schema.route r ON t.id\_route = r.id\_route

WHERE DATE(r.date\_departure) = CURRENT\_DATE - INTERVAL '1 day' and t.status = 'Available';



* + Определить, какой тип самолетов чаще всего летал в заданный аэропорт назначения. Листинг:

SELECT \* FROM (

SELECT m.type\_of\_plane as type\_of\_plane, COUNT(\*) AS flight\_count

FROM schema.route r

JOIN schema.schedule sc ON r.id\_schedule = sc.id\_schedule JOIN schema.plane p ON r.id\_plane = p.id\_plane

JOIN schema.model m ON p.id\_model = m.id\_model WHERE sc.id\_airport\_arrival = 1

GROUP BY type\_of\_plane ORDER BY flight\_count DESC

)

WHERE flight\_count in (SELECT MAX(flight\_count) FROM ( SELECT m.type\_of\_plane as type\_of\_plane,

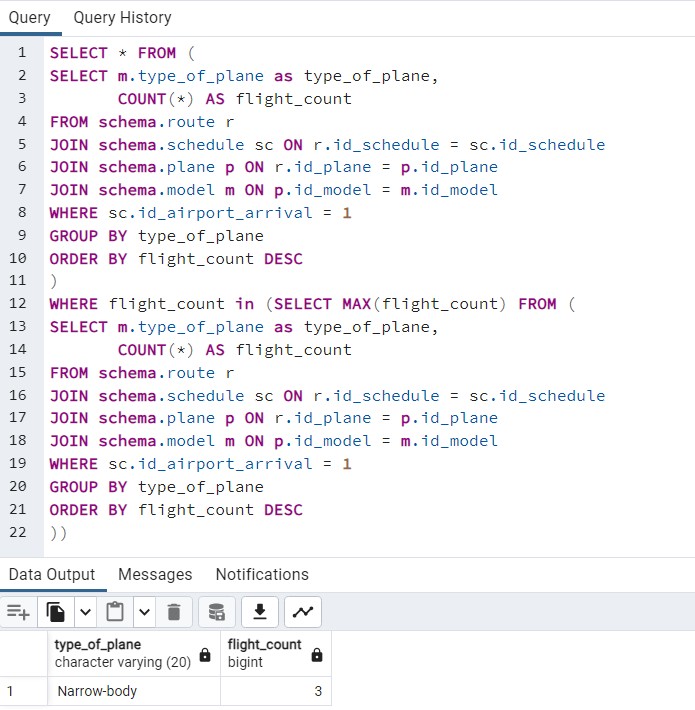
COUNT(\*) AS flight\_count FROM schema.route r

JOIN schema.schedule sc ON r.id\_schedule = sc.id\_schedule JOIN schema.plane p ON r.id\_plane = p.id\_plane

JOIN schema.model m ON p.id\_model = m.id\_model WHERE sc.id\_airport\_arrival = 1

GROUP BY type\_of\_plane ORDER BY flight\_count DESC

))



* + Вывести список самолетов, “возраст” которых превышает средний “возраст” самолетов этого типа.

Листинг:

SELECT \* FROM (

SELECT p.id\_plane, p.tail\_number,

AGE(CURRENT\_DATE, p.date\_last\_repair) AS plane\_age,

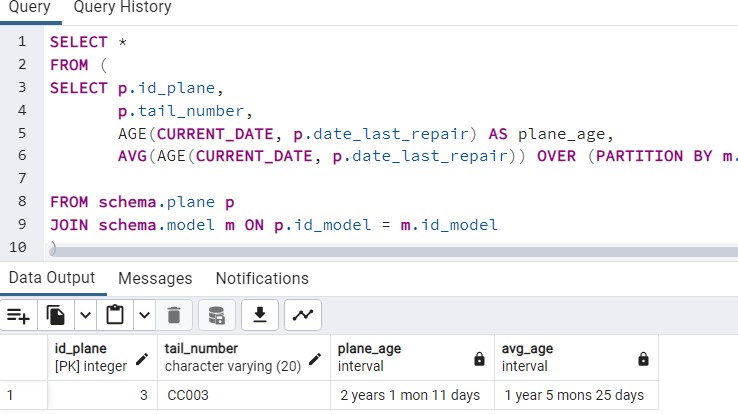
AVG(AGE(CURRENT\_DATE, p.date\_last\_repair)) OVER (PARTITION BY m.type\_of\_plane) AS avg\_age

FROM schema.plane p

JOIN schema.model m ON p.id\_model = m.id\_model

)

WHERE plane\_age > avg\_age



1. Создание представлений
   * для пассажиров авиакомпании о рейсах в Москву на ближайшую неделю; Листинг:

CREATE VIEW schema.passenger\_flights\_to\_moscow AS SELECT DISTINCT

r.id\_route, r.date\_departure, r.date\_arrival

FROM

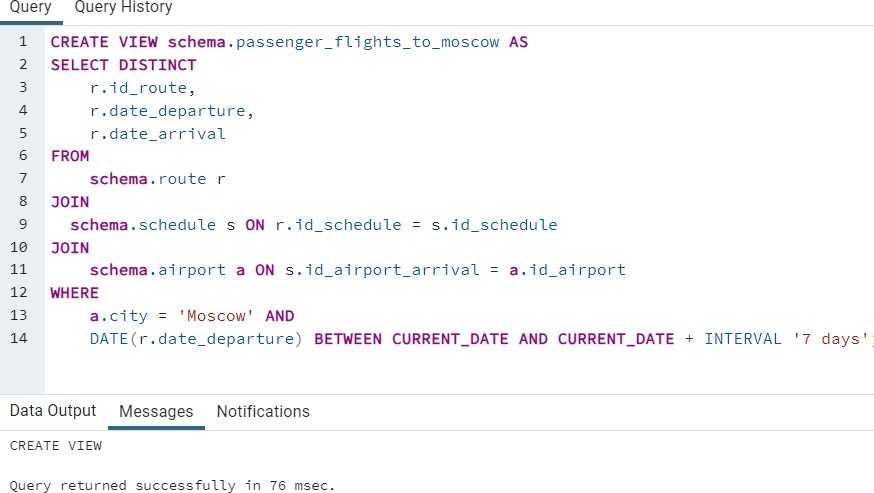
schema.route r JOIN

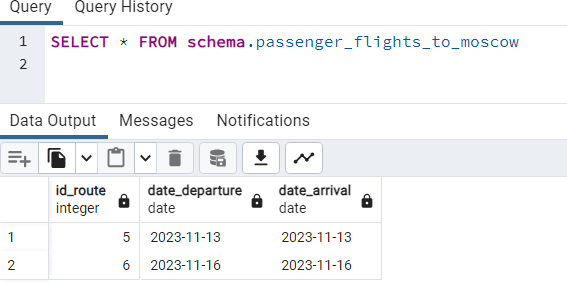
schema.schedule s ON r.id\_schedule = s.id\_schedule JOIN

schema.airport a ON s.id\_airport\_arrival = a.id\_airport WHERE

a.city = 'Moscow' AND

DATE(r.date\_departure) BETWEEN CURRENT\_DATE AND CURRENT\_DATE + INTERVAL '7 days';





* + количество самолетов каждого типа, летавшими за последний месяц. Листинг:

CREATE VIEW schema.aircraft\_count\_by\_type AS SELECT

m.type\_of\_plane,

COUNT(r.id\_plane) AS airplane\_count FROM

schema.route r

JOIN

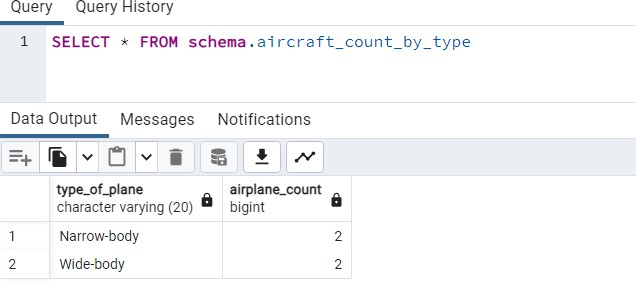
schema.plane p ON p.id\_plane = r.id\_plane JOIN

schema.model m ON p.id\_model = m.id\_model WHERE

DATE(r.date\_departure) BETWEEN CURRENT\_DATE - INTERVAL '1 month' AND CURRENT\_DATE

GROUP BY

m.type\_of\_plane;



1. Запросы на модификацию данных
   * INSERT – вставить в таблицу с кассами кассы, находящиеся в аэропортах Листинг:

INSERT INTO schema.ticket\_office(country, city, address) SELECT country, city, name

FROM schema.airport;



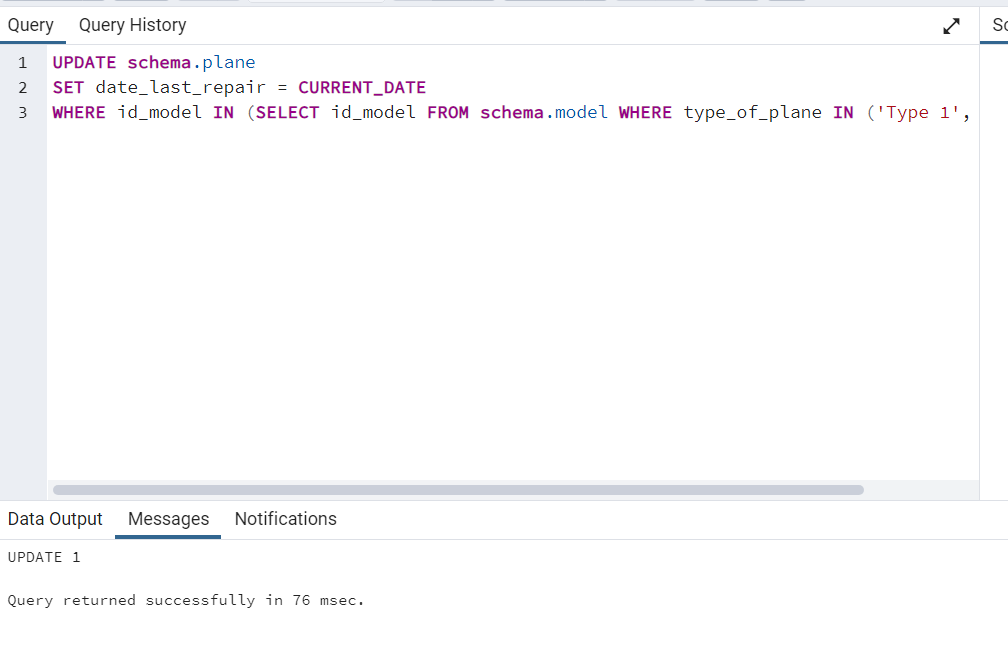
* + UPDATE – обновить дату последнего ремонта на сегодняшнюю для всех самолетов типа Type 1 или Type 3

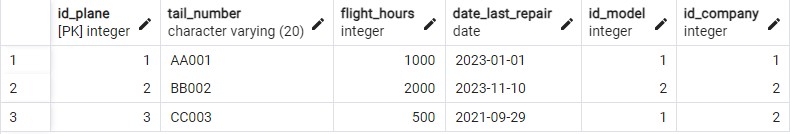
Листинг:

UPDATE schema.plane

SET date\_last\_repair = CURRENT\_DATE

WHERE id\_model IN (SELECT id\_model FROM schema.model WHERE type\_of\_plane IN ('Type 1', 'Type 3'));





* + DELETE – удалить все самолеты, принадлежащие компаниям определенной страны Листинг:

DELETE FROM schema.plane

WHERE id\_company IN (SELECT id\_company FROM schema.company WHERE country = 'Country 3');



1. Создание индексов Листинг:

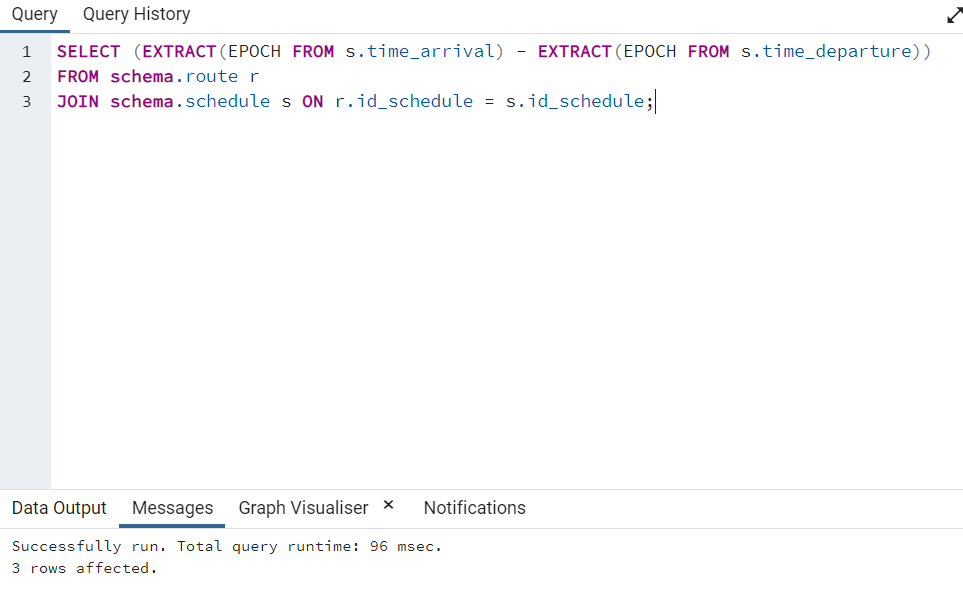
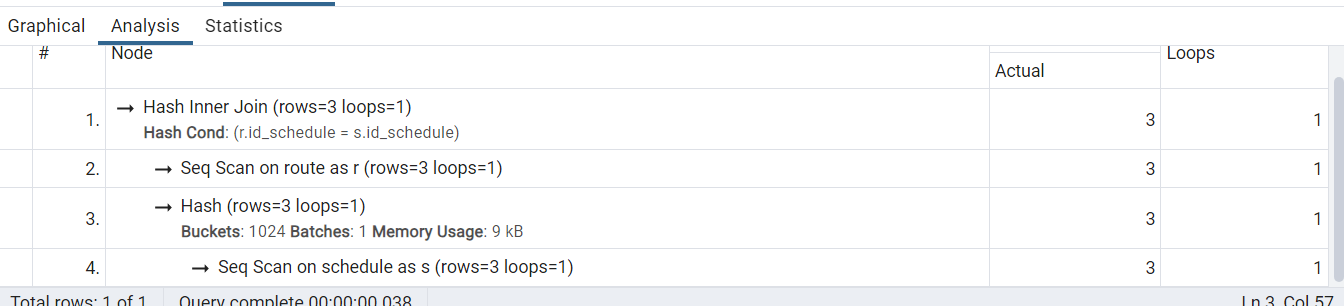
CREATE INDEX idx\_plane ON schema.route (id\_plane);

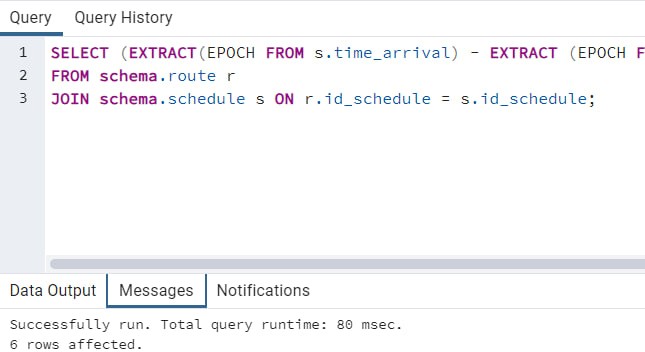
CREATE INDEX idx\_fuel ON schema.plane (flight\_hours, id\_model); Без индексов:

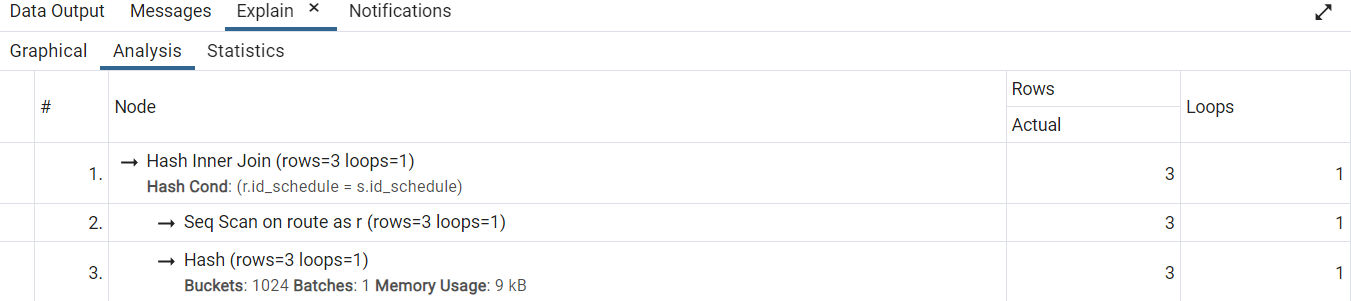
SELECT (EXTRACT(EPOCH FROM s.time\_arrival) - EXTRACT(EPOCH FROM s.time\_departure))

FROM schema.route r

JOIN schema.schedule s ON r.id\_schedule = s.id\_schedule;



С индексами:



# Вывод

В ходе лабораторной работы я освоила работу с различными SQL-запросами к базе данных, также создание представлений и индексов. Также сравнил время работы SELECT запросов с индексами и без. Разумеется, с индексами время меньше.