**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №4 «Создание таблиц базы данных PostgreSQL. Заполнение таблиц рабочими данными»

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

Автор: Тюленев А.С.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**Оглавление**

[Цель работы **Ошибка! Закладка не определена.**](#_Toc148889485)

[Практическое задание **Ошибка! Закладка не определена.**](#_Toc148889487)

[Вариант 6. БД «Пассажир» **Ошибка! Закладка не определена.**](#_Toc148889495)

[Рисунок 1 – Схема логической модели базы данных. **Ошибка! Закладка не определена.**](#_Toc148889496)

[Листинг дампа **Ошибка! Закладка не определена.**](#_Toc148889497)

[Вывод 4](#_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.

1. Создать запросы:

Свободные места на все поезда, отправляющиеся с вокзала в течение следующих суток.

**SELECT**

t.id **AS** train\_id,

t.departure\_time,

w.id **AS** wagon\_id,

s.id **AS** seat\_id,

s.seat\_type,

s.occupancy\_status

**FROM**

train t

**JOIN** wagon w **ON** t.id = w.trip

**JOIN** seats s **ON** w.id = s.wagon

**JOIN** station st **ON** t.stdepid = st.id

**WHERE**

st.station\_name = 'Москва'

**AND** t.departure\_time **BETWEEN** **NOW**() **AND** **NOW**() + **INTERVAL** '1 day'

**AND** s.occupancy\_status = 'Свободно';

Список пассажиров, отправившихся в Москву всеми рейсами за прошедшие сутки.

**SELECT**

p.full\_name,

tr.departure\_time,

tr.arrival\_time

**FROM**

passenger p

**JOIN** ticket tk **ON** p.id = tk.passenger\_id

**JOIN** trip tr **ON** tk.id = tr.ticket\_id

**JOIN** station s **ON** tr.dest\_stationid = s.id

**WHERE**

s.station\_name = 'Москва'

**AND** tr.departure\_time **BETWEEN** **NOW**() - **INTERVAL** '1 day' **AND** **NOW**();

Номера поездов, на которые проданы все билеты на следующие сутки.

**SELECT**

tr.id **AS** train\_id

**FROM**

train tr

**JOIN** trip tp **ON** tr.id = tp.train

**JOIN** wagon w **ON** tr.id = w.trip

**JOIN** seats s **ON** w.id = s.wagon

**JOIN** ticket tk **ON** s.id = tk.seat\_id

**WHERE**

tp.departure\_date **BETWEEN** **CURRENT\_DATE** + **INTERVAL** '1 day' **AND** **CURRENT\_DATE** + **INTERVAL** '2 days'

**GROUP** **BY**

tr.id

**HAVING**

**COUNT**(**DISTINCT** s.id) = **COUNT**(tk.id);

Свободные места в купейные вагоны всех рейсов до Москвы на текущие сутки.

**SELECT**

tr.id **AS** train\_id,

w.wagon\_number,

s.id **AS** seat\_id,

s.seat\_type

**FROM**

train tr

**INNER** **JOIN** trip tp **ON** tr.id = tp.train

**INNER** **JOIN** wagon w **ON** tr.id = w.trip

**INNER** **JOIN** wtype wt **ON** w.wtype = wt.id

**INNER** **JOIN** seats s **ON** w.id = s.wagon

**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id **AND** tk.sale\_date = **CURRENT\_DATE**

**INNER** **JOIN** station st **ON** tp.dest\_stationid = st.id

**WHERE**

st.station\_name = 'Москва'

**AND** wt.wagon\_name = 'Купе'

**AND** tp.departure\_date = **CURRENT\_DATE**

**AND** tk.id **IS** **NULL**;

Выручка от продажи билетов на все поезда за прошедшие сутки.

**SELECT**

**SUM**(tk.ticket\_price) **AS** total\_revenue

**FROM**

ticket tk

**WHERE**

tk.sale\_date **BETWEEN** **CURRENT\_DATE** - **INTERVAL** '1 day' **AND** **CURRENT\_DATE**;

Общее количество билетов, проданных по всем направлениям в вагоны типа “СВ”.

**SELECT**

**COUNT**(\*) **AS** total\_sv\_tickets

**FROM**

ticket tk

**JOIN** seats s **ON** tk.seat\_id = s.id

**JOIN** wagon w **ON** s.wagon = w.id

**JOIN** wtype wt **ON** w.wtype = wt.id

**WHERE**

wt.wagon\_name = 'СВ';

Номера и названия поездов, все вагоны которых были заполнены менее чем наполовину за прошедшие сутки.

**SELECT**

tr.id **AS** train\_id,

tr.train\_name

**FROM**

train tr

**JOIN** wagon w **ON** tr.id = w.trip

**JOIN** wtype wt **ON** w.wtype = wt.id

**LEFT** **JOIN** (

**SELECT**

s.wagon,

**COUNT**(tk.id) **AS** tickets\_sold

**FROM**

seats s

**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id **AND** tk.sale\_date **BETWEEN** **CURRENT\_DATE** - **INTERVAL** '1 day' **AND** **CURRENT\_DATE**

**GROUP** **BY** s.wagon

) **AS** ts **ON** w.id = ts.wagon

**WHERE**

ts.tickets\_sold <= (wt.seats\_total / 2)

**GROUP** **BY**

tr.id,

tr.train\_name

**HAVING**

**COUNT**(w.id) = **COUNT**(ts.wagon);

Представления:

Для пассажиров о наличии свободных мест на заданный рейс.

**CREATE** **VIEW** available\_seats\_view **AS**

**SELECT**

tr.id **AS** train\_id,

tr.train\_name,

w.wagon\_number,

wt.wagon\_name,

s.id **AS** seat\_id,

s.seat\_type,

**CASE**

**WHEN** tk.id **IS** **NULL** **THEN** 'Свободно'

**ELSE** 'Занято'

**END** **AS** occupancy\_status

**FROM**

train tr

**JOIN** wagon w **ON** tr.id = w.trip

**JOIN** wtype wt **ON** w.wtype = wt.id

**JOIN** seats s **ON** w.id = s.wagon

**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id;

Количество непроданных билетов на все поезда, формирующиеся за прошедшие сутки (номер поезда, тип вагона, количество).

**CREATE** **VIEW** unsold\_tickets\_view **AS**

**SELECT**

tr.id **AS** train\_id,

wt.wagon\_name **AS** wagon\_type,

**COUNT**(\*) - **COUNT**(tk.id) **AS** unsold\_tickets\_count

**FROM**

train tr

**JOIN** trip tp **ON** tr.id = tp.train

**JOIN** wagon w **ON** tr.id = w.trip

**JOIN** wtype wt **ON** w.wtype = wt.id

**JOIN** seats s **ON** w.id = s.wagon

**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id **AND** tk.sale\_date **BETWEEN** **CURRENT\_DATE** - **INTERVAL** '1 day' **AND** **CURRENT\_DATE**

**WHERE**

tp.departure\_date **BETWEEN** **CURRENT\_DATE** - **INTERVAL** '1 day' **AND** **CURRENT\_DATE**

**GROUP** **BY**

tr.id, wt.wagon\_name;

2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.

INSERT

**INSERT** **INTO** ticket (passenger\_id, seat\_id, ticket\_price, sale\_date, ticket\_status, dep\_stationid, dest\_stationid)

**SELECT**

1,

(**SELECT** s.id **FROM** seats s

**INNER** **JOIN** wagon w **ON** s.wagon = w.id

**INNER** **JOIN** wtype wt **ON** w.wtype = wt.id

**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id **AND** tk.sale\_date = **CURRENT\_DATE**

**WHERE** wt.wagon\_name = 'Купе'

**AND** tk.id **IS** **NULL**

**LIMIT** 1),

5000,

**CURRENT\_DATE**,

'Выкуплен',

1,

2

**WHERE**

**EXISTS** (

**SELECT** s.id **FROM** seats s

**INNER** **JOIN** wagon w **ON** s.wagon = w.id

**INNER** **JOIN** wtype wt **ON** w.wtype = wt.id

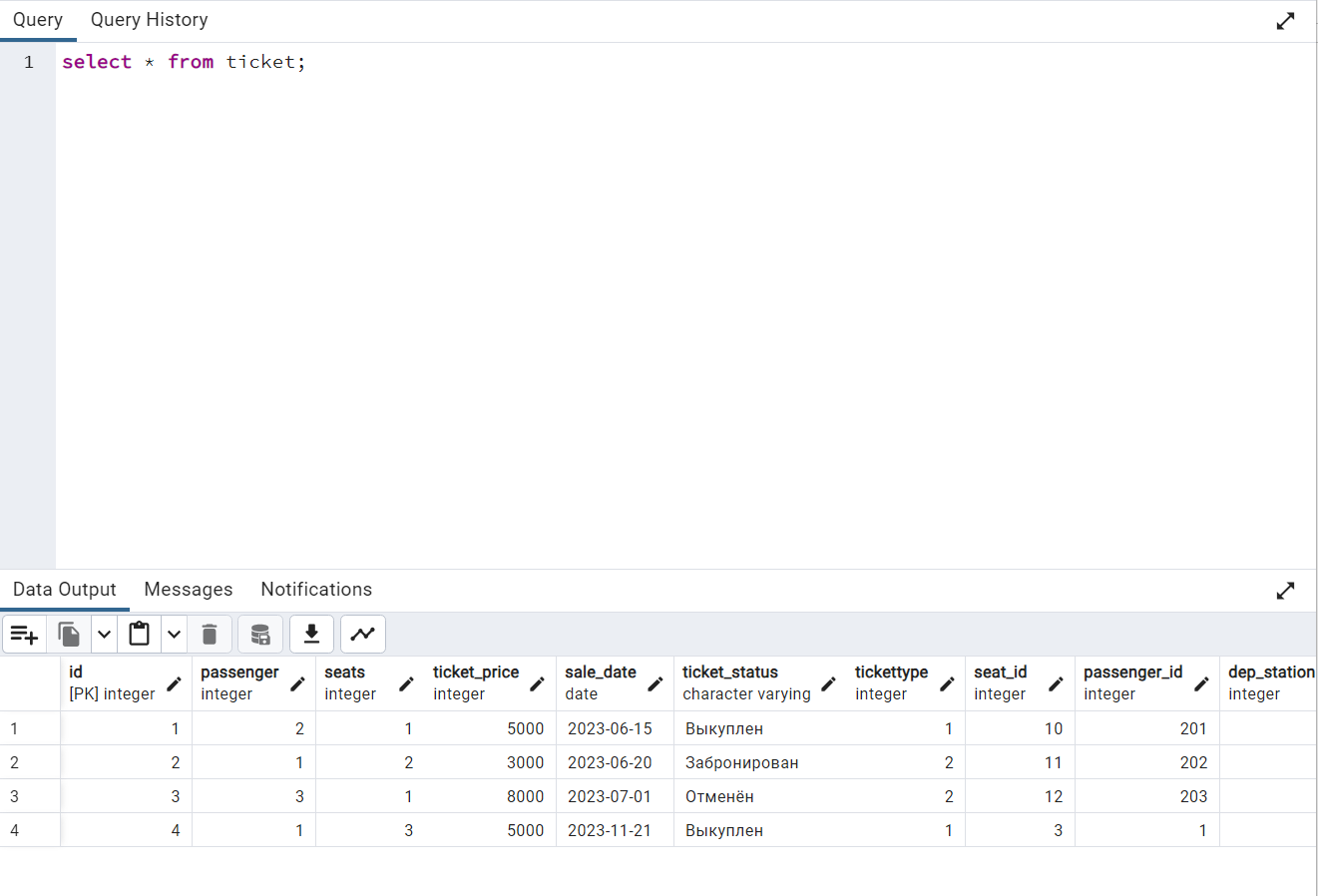
**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id **AND** tk.sale\_date = **CURRENT\_DATE**

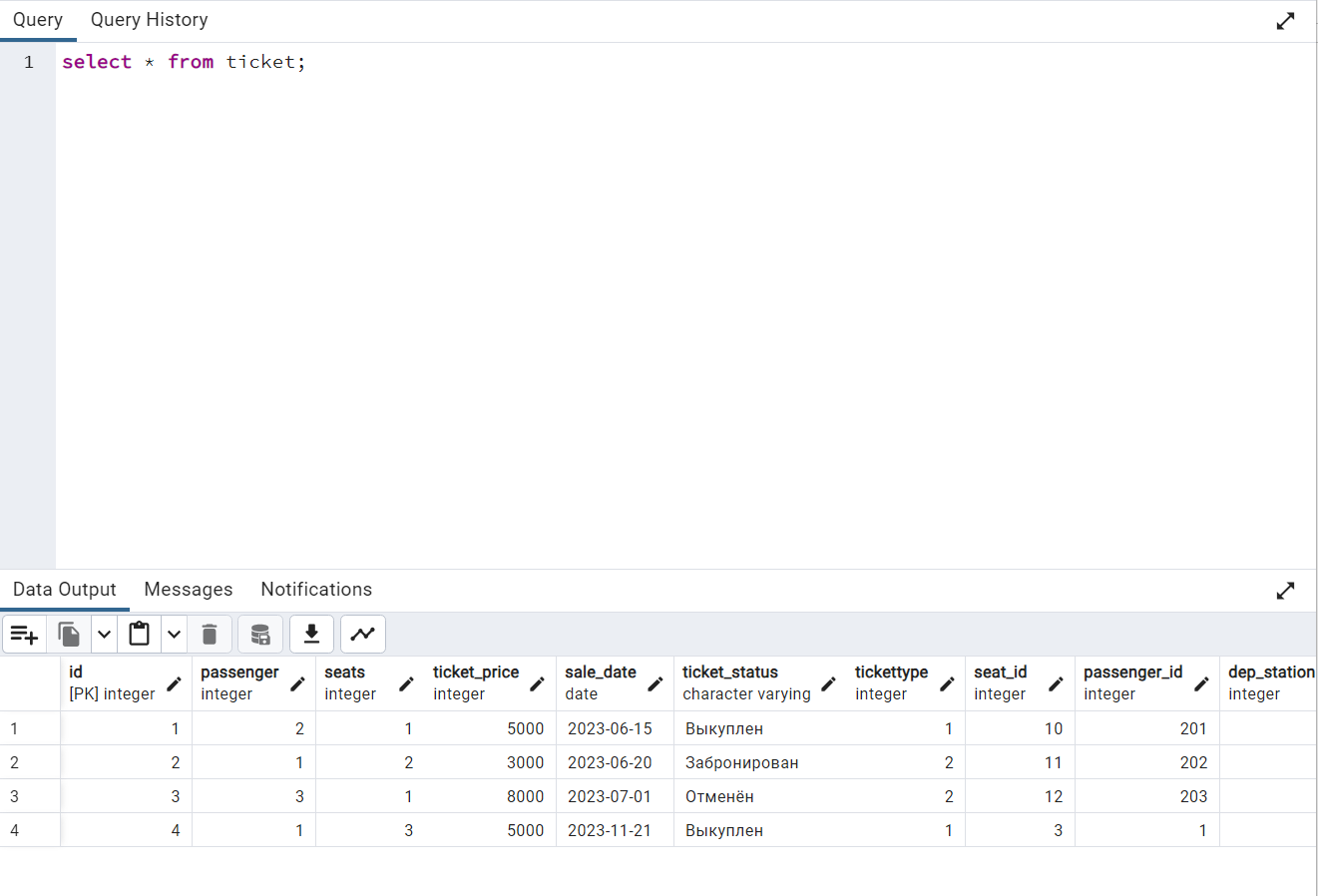
**WHERE** wt.wagon\_name = 'Купе'

**AND** tk.id **IS** **NULL**

**LIMIT** 1

);





**UPDATE**

**UPDATE** ticket

**SET** ticket\_status = 'Отменен'

**WHERE** id **IN** (

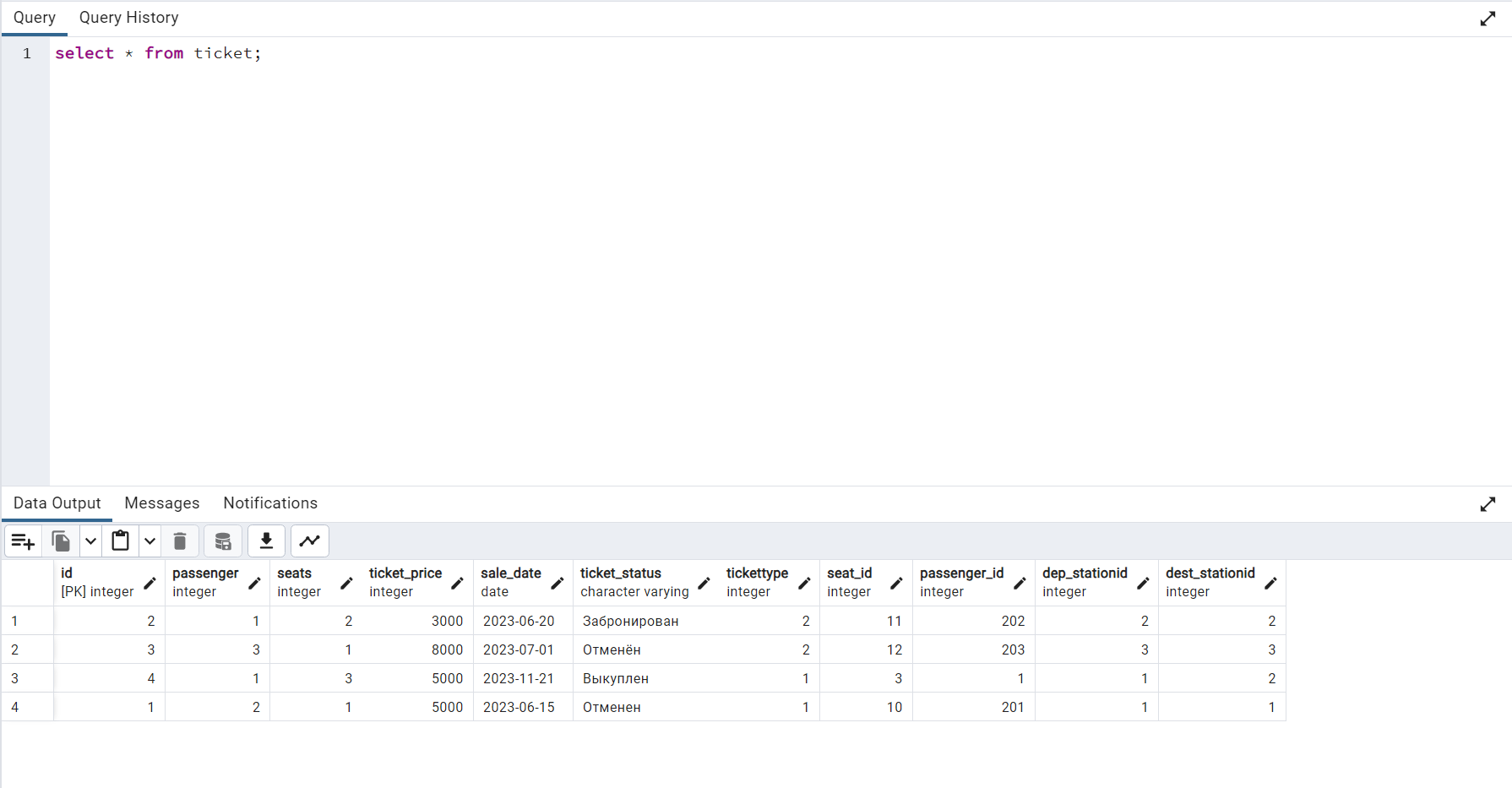
**SELECT** tk.id **FROM** ticket tk

**INNER** **JOIN** stnumber sn **ON** sn.id = tk.dest\_stationid

**JOIN** trip tr **ON** sn.trip\_id = tr.id

**WHERE** tr.status = 'Отменен'

);



DELETE

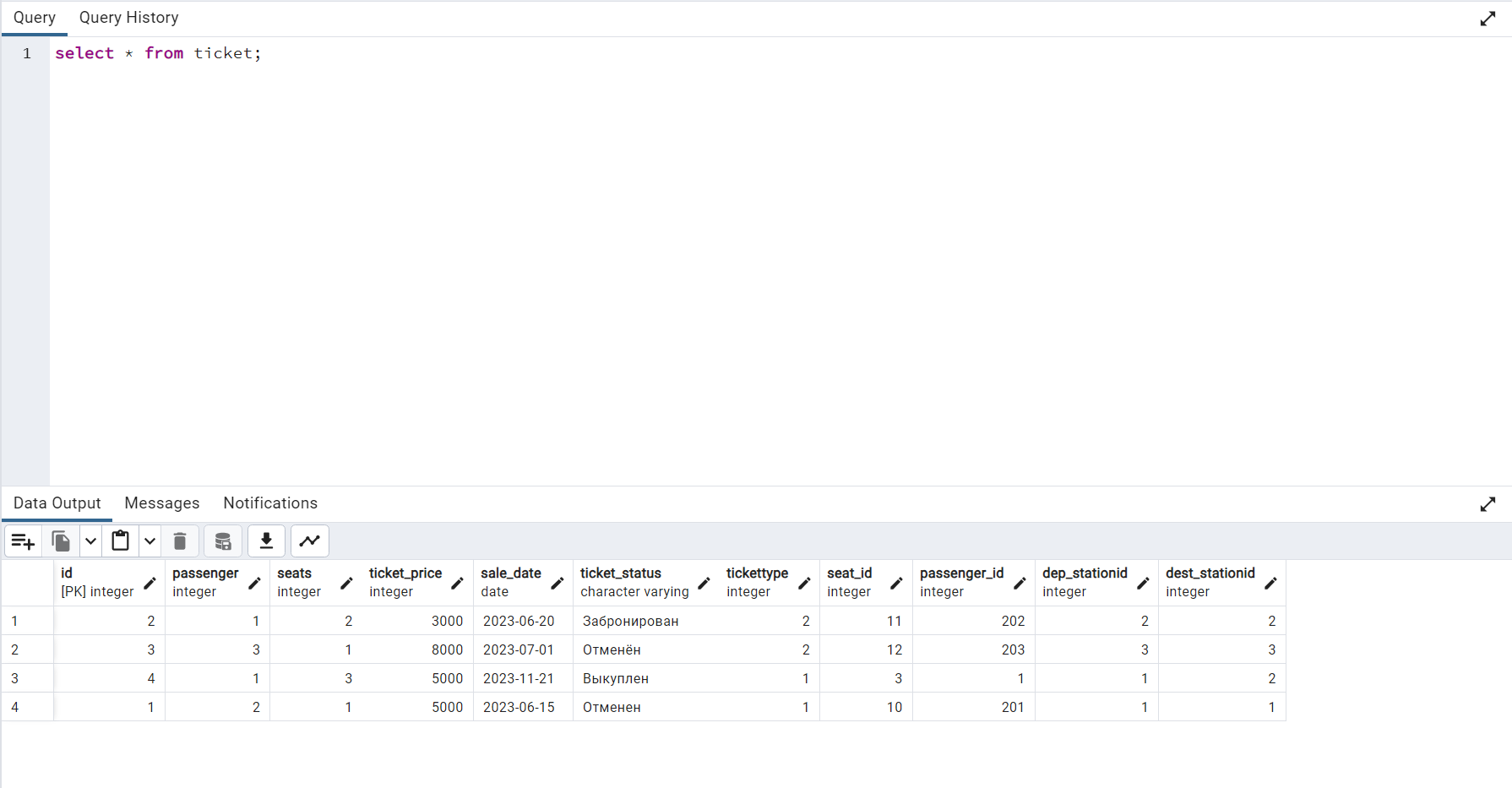
**DELETE** **FROM** ticket

**WHERE** dep\_stationid = (

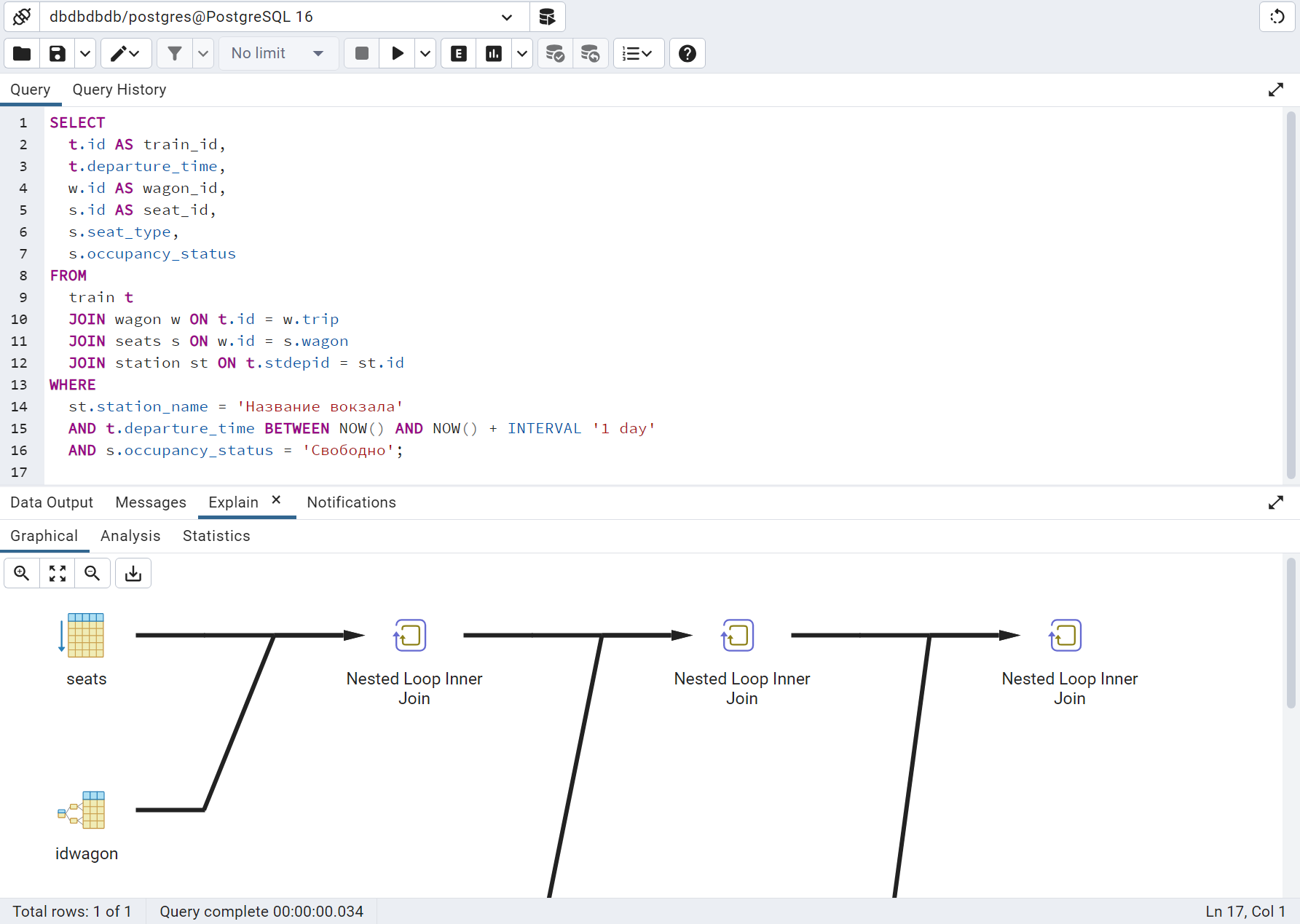
**SELECT** id **FROM** station

**WHERE** station\_name = 'Станция закрыта на ремонт'

);



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.

Запрос №1. Свободные места в купейные вагоны всех рейсов до Москвы на текущие сутки.

**SELECT**

tr.id **AS** train\_id,

w.wagon\_number,

s.id **AS** seat\_id,

s.seat\_type

**FROM**

train tr

**INNER** **JOIN** trip tp **ON** tr.id = tp.train

**INNER** **JOIN** wagon w **ON** tr.id = w.trip

**INNER** **JOIN** wtype wt **ON** w.wtype = wt.id

**INNER** **JOIN** seats s **ON** w.id = s.wagon

**LEFT** **JOIN** ticket tk **ON** s.id = tk.seat\_id **AND** tk.sale\_date = **CURRENT\_DATE**

**INNER** **JOIN** station st **ON** tp.dest\_stationid = st.id

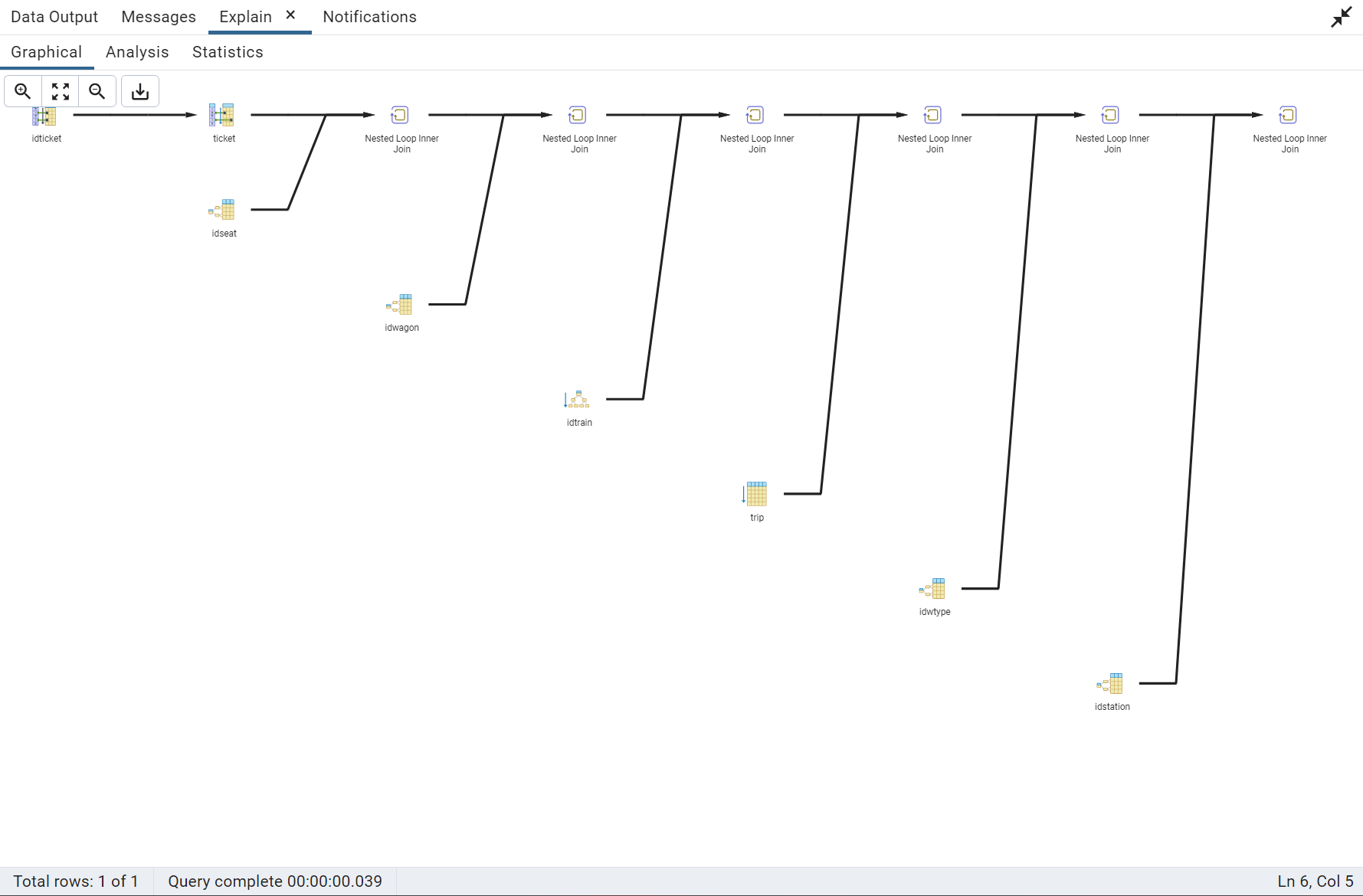
**WHERE**

st.station\_name = 'Москва'

**AND** wt.wagon\_name = 'Купе'

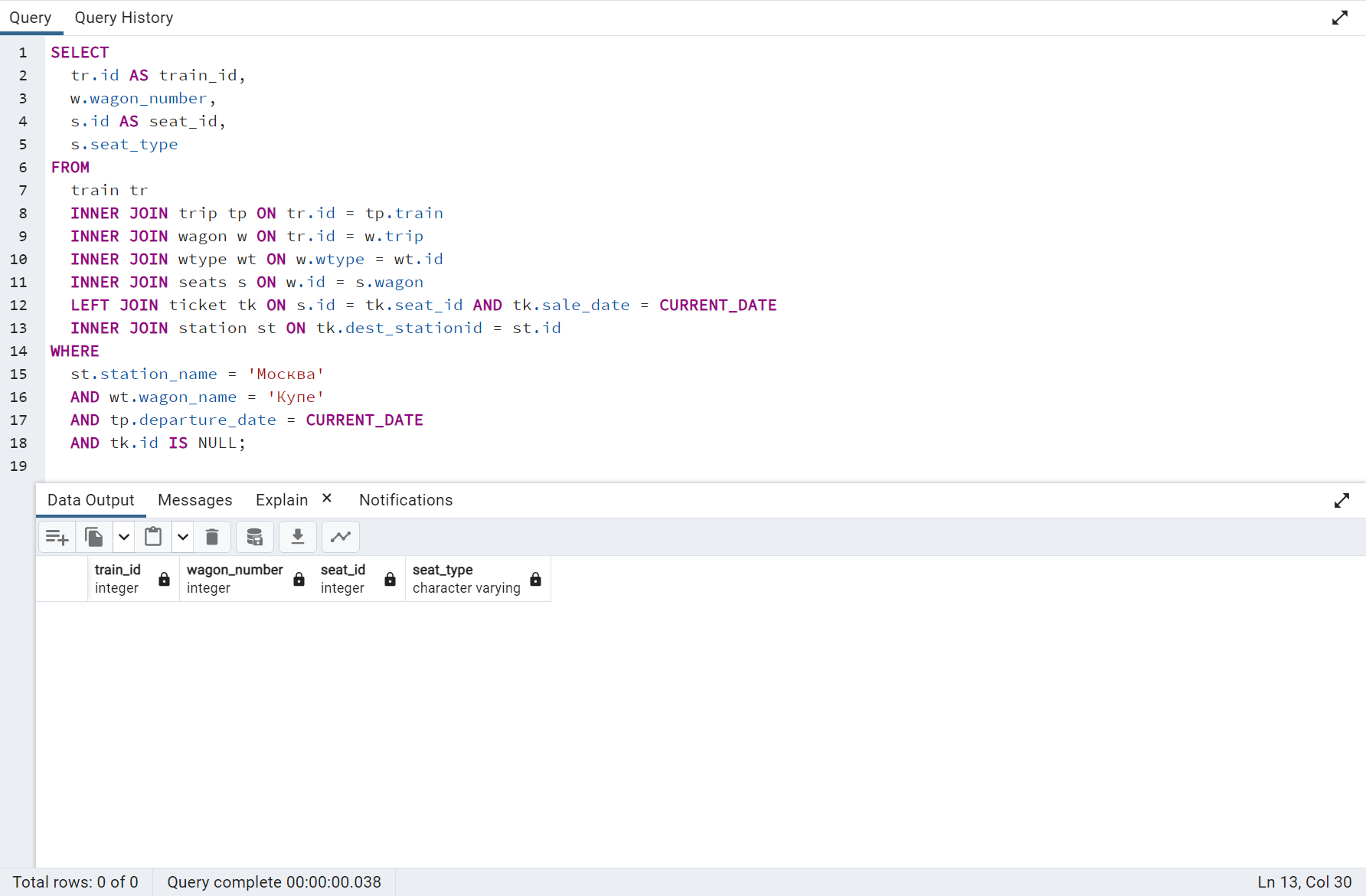
**AND** tp.departure\_date = **CURRENT\_DATE**

**AND** tk.id **IS** **NULL**;



Индекс:

**CREATE** **INDEX** idx\_station\_name **ON** station(station\_name);



Запрос №2. Свободные места на все поезда, отправляющиеся с вокзала в течение следующих суток.

**SELECT**

t.id **AS** train\_id,

t.departure\_time,

w.id **AS** wagon\_id,

s.id **AS** seat\_id,

s.seat\_type,

s.occupancy\_status

**FROM**

train t

**JOIN** wagon w **ON** t.id = w.trip

**JOIN** seats s **ON** w.id = s.wagon

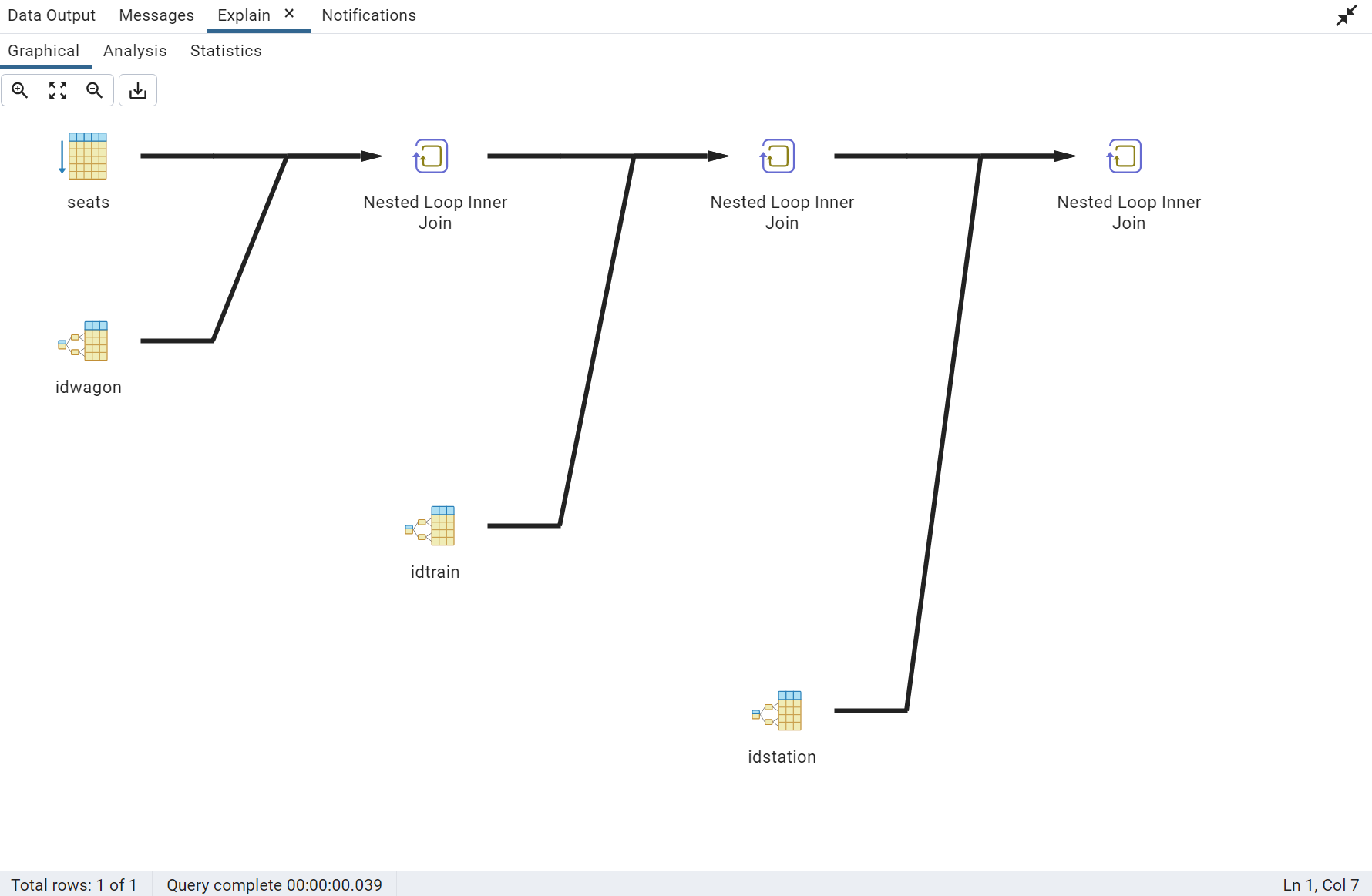
**JOIN** station st **ON** t.stdepid = st.id

**WHERE**

st.station\_name = 'Москва'

**AND** t.departure\_time **BETWEEN** **NOW**() **AND** **NOW**() + **INTERVAL** '1 day'

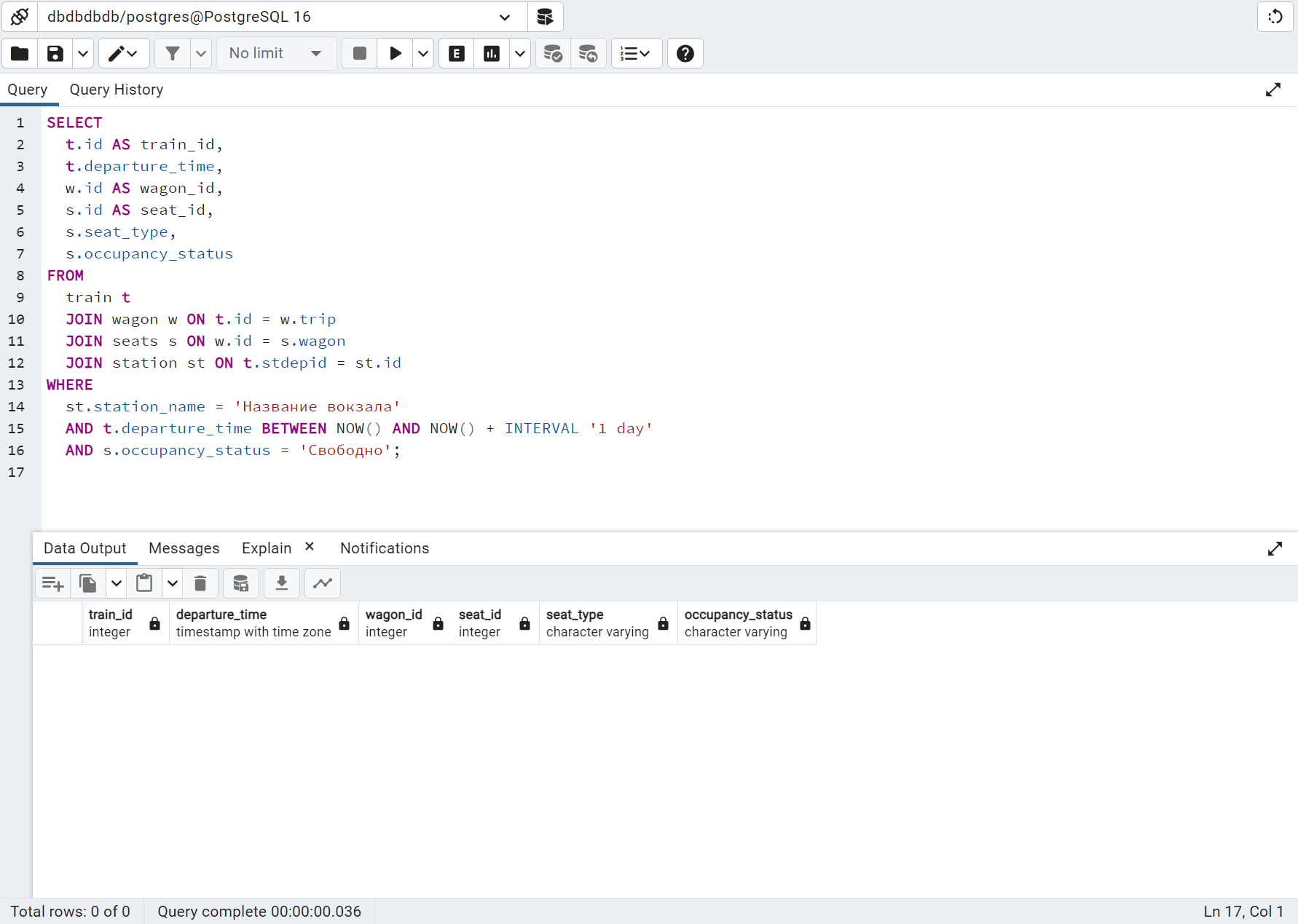
**AND** s.occupancy\_status = 'Свободно';



Индекс:

**CREATE** **INDEX** idx\_train\_station\_seats **ON** train(departure\_time) INCLUDE (id);

**CREATE** **INDEX** idx\_seats\_occupancy **ON** seats(occupancy\_status) INCLUDE (wagon);

****

# Вывод

Во время лабораторной работы я научился выполнять разнообразные SQL-запросы к базе данных, а также создавать представления и индексы. Кроме того, я сравнил скорость выполнения SELECT-запросов до и после создания индексов. Безусловно, использование индексов привело к сокращению времени выполнения запросов (однако стоит отметить, что не к существенному, так как данных достаточно мало).