

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: Пиотуховский А.А.

Факультет: ИКТ

Группа: K3241

Преподаватель: Говорова М.М.

ИТМО

Санкт-Петербург 2023

Оглавление

Цель работы	3
Практическое задание	3
Вариант 10. БД «Автовокзал»	3
Схема базы данных	4
Выполнение	6
Вывод.....	13

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.

Вариант 10. БД «Автовокзал»

Описание предметной области:

С автовокзала ежедневно отправляется несколько междугородных/международных автобусных рейсов. Номер рейса определяется маршрутом и временем отправления. По всем промежуточным остановкам на маршруте известны название, тип населенного пункта, время прибытия, отправления, время стоянки.

Автобусы курсируют по расписанию, но могут назначаться дополнительные рейсы на заданный период или определенные даты.

Билеты могут продаваться предварительно, но не ранее чем за 10 суток. В билете указывается номер места в автобусе. На каждый рейс может продаваться не более 10 билетов без места, цена на которые снижается на 10%. Пунктами отправления и назначения, согласно билету, могут быть промежуточные остановки.

Билеты могут продаваться в кассе автовокзала или онлайн.

Необходимо учитывать, что местом посадки и высадки пассажира могут быть промежуточные остановки согласно купленному билету.

На каждый рейс формируется экипаж из двух водителей.

БД должна содержать следующий минимальный набор сведений: Номер рейса. Номер водителя. Номер автобуса. Паспортные данные водителя. Пункт отправления. Пункт назначения. Промежуточные остановки. Дата отправления. Время отправления. Время в пути. Тип автобуса. Количество мест в автобусе. Страна. Производитель. Год выпуска.

Номер билета. Номер места в автобусе (при наличии). Цена билета. ФИО пассажира.
Паспортные данные пассажира.

Схема базы данных

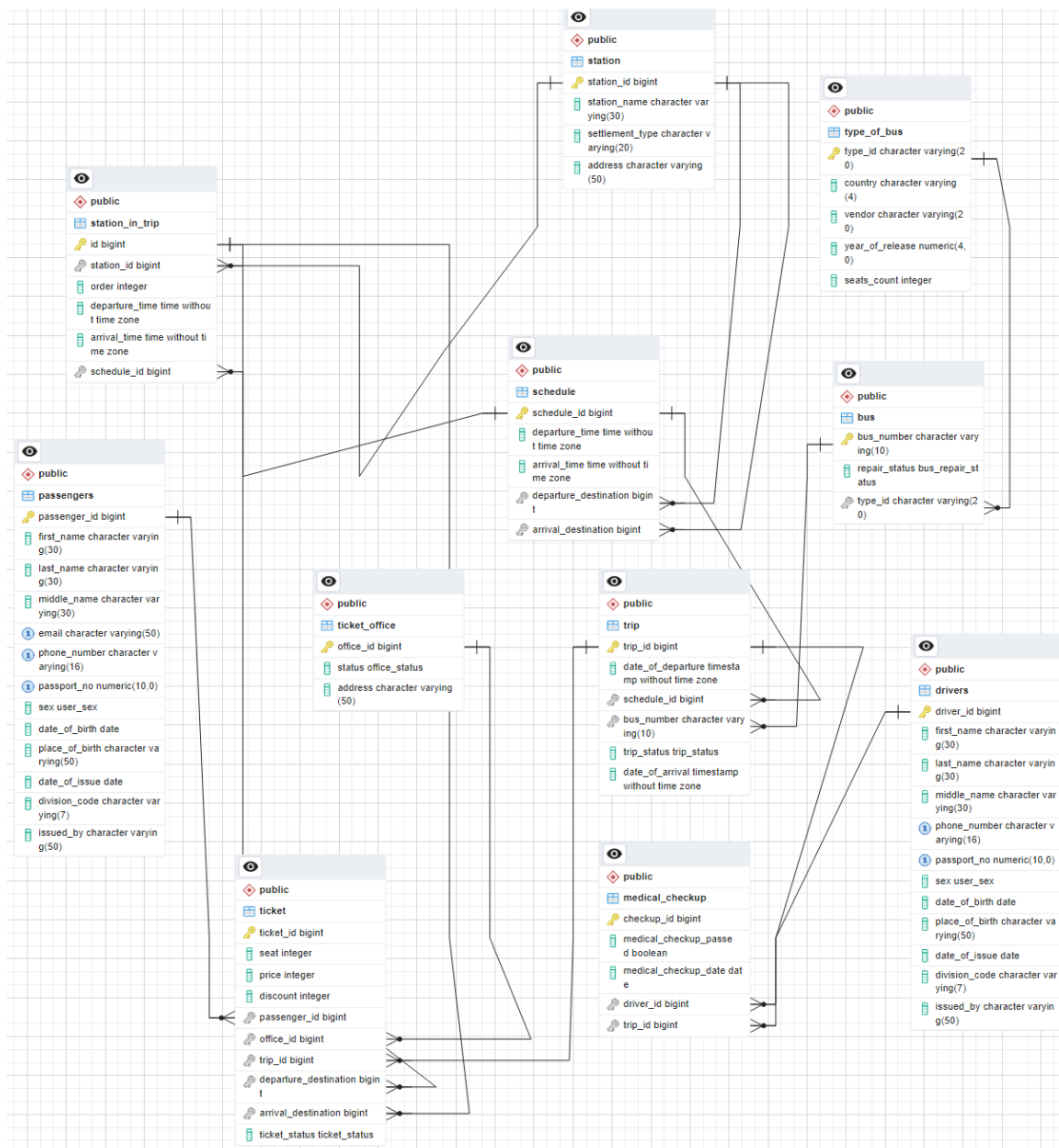


Рисунок 1 – Схема логической модели базы данных в pgAdmin4.

ERD в pgAdmin 4 строит верные, но запутанные связи. Без пересоздания всех связей собрать таблицы в аккуратном порядке не получилось. На рисунке 2 представлена эта же логическая схема, но выполненная в IDEA от JetBrains (Powered by yFiles).

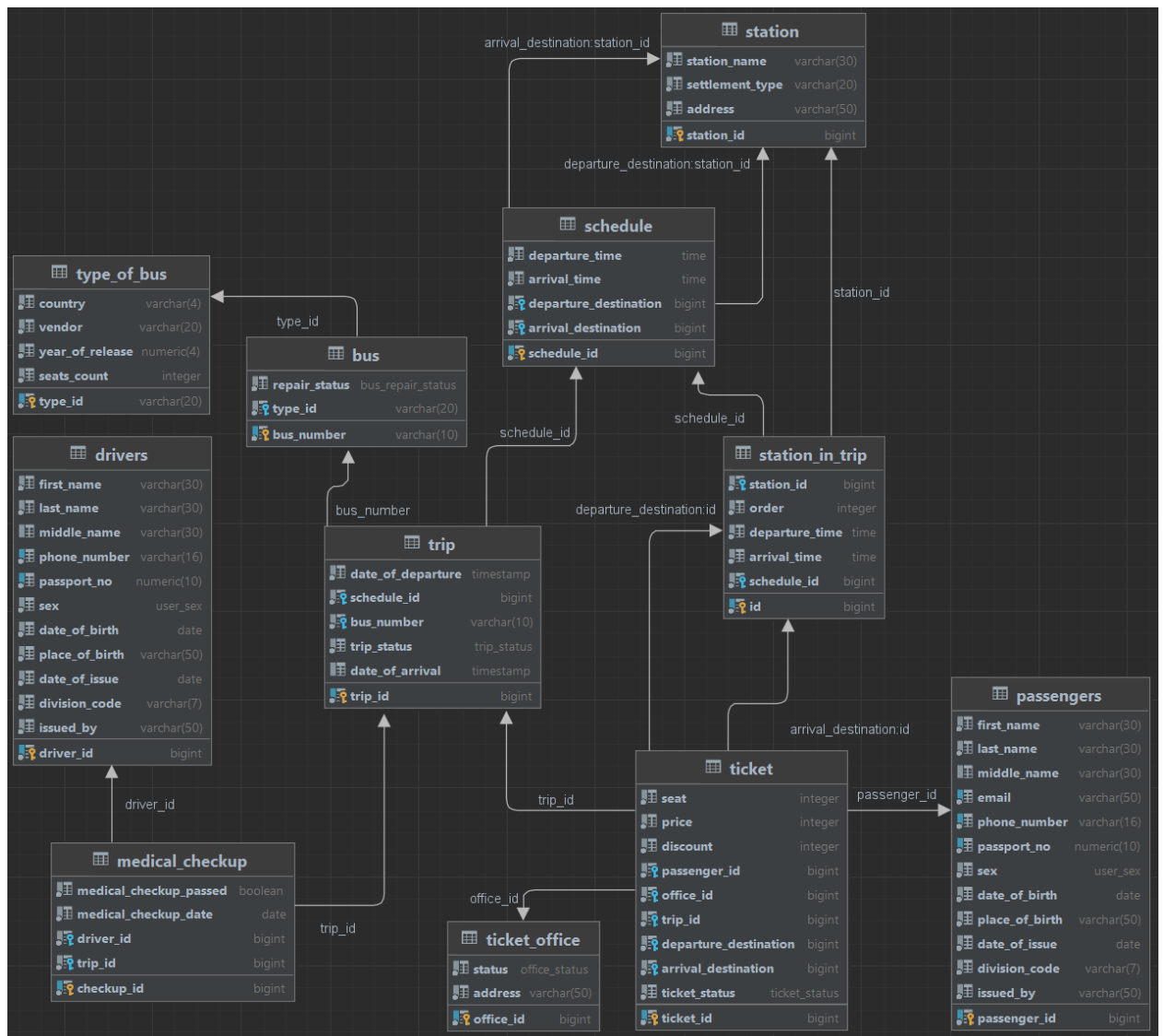


Рисунок 2 – Схема логической модели базы данных в IDEA от JetBrains.

Выполнение

Запрос 1. Вывести фамилии водителей и номера автобусов, отправившиеся в рейсы до 12 часов текущего дня.

```
SELECT drivers.last_name, trip.bus_number
from trip
      INNER JOIN medical_checkup ON trip.trip_id = medical_checkup.trip_id
      LEFT JOIN drivers ON medical_checkup.driver_id = drivers.driver_id
where trip.trip_id in (SELECT trip_id
                      FROM trip
                      WHERE date_of_departure between current_date +
'00:00:00'::time and current_date + '12:00:00'::time
                      AND trip_status != 'canceled'::trip_status)
AND medical_checkup_passed;
```

	last_name character varying	bus_number character varying
1	Тороп	e947at147
2	Садко	e947at147

Рисунок 3 – Результат работы запроса №1.

Запрос 2. Рассчитать выручку от продажи билетов за прошедший день.

```
SELECT sum(price - discount)
FROM ticket
WHERE date(date_of_transaction) = current_date - INTERVAL '1 day'
AND ticket_status = 'paid'::ticket_status;
```

	sum bigint
1	39

Рисунок 5 – Результат работы запроса №2.

Запрос 3. Вывести список водителей, которые не выполнили ни одного рейса за прошедший день.

```
SELECT drivers.last_name
from drivers
      INNER JOIN public.medical_checkup mc on drivers.driver_id =
mc.driver_id
      LEFT JOIN public.trip t on mc.trip_id = t.trip_id
WHERE t.trip_id not in (SELECT trip_id
                        FROM trip
                        WHERE date(date_of_departure) = current_date -
INTERVAL '1 day'
                        AND trip_status != 'canceled'::trip_status
                        AND mc.medical_checkup_passed);
```


	last_name character varying 
1	Крутоголов
2	Садко

Рисунок 7 – Результат работы запроса №3.

Запрос 5. Сколько рейсов выполнил каждый водитель за последний месяц.

```
SELECT driver_id,
       last_name,
       (select count(*)
        from trip
        left join public.medical_checkup mc on trip.trip_id =
mc.trip_id
        where drivers.driver_id = mc.driver_id
        and mc.medical_checkup_passed
        and trip_status != 'canceled'::trip_status
        and date(date_of_departure) >= now() - interval '1 month') as count
from drivers
group by driver_id;
```




	driver_id [PK] bigint 	last_name character varying 	count bigint 
1	1	Крутоголов	0
2	3	Садко	1
3	4	Тороп	1

Рисунок 9 – Результат работы запроса №5.

Запрос 6. Вывести самый популярный(ые) тип(ы) автобуса(ов), который(е) используется(ются) в рейсах.

```
SELECT type.type_id, count(*) as count
from type_of_bus as type
    left join bus b on type.type_id = b.type_id
    right join trip t on b.bus_number = t.bus_number
group by type.type_id
having count(*) = (select max(cnt)
    from (select count(*) as cnt
        from type_of_bus as type
            left join bus b on type.type_id = b.type_id
            right join trip t on b.bus_number =
t.bus_number
        group by type.type_id) as subquery)
order by count desc;
```

	type_id [PK] character varying	count bigint
1	Лазурный	2

Рисунок 11 – Результат работы запроса №6.

Запрос 7. Вывести данные водителя, который провел максимальное время в пути за прошедшую неделю.

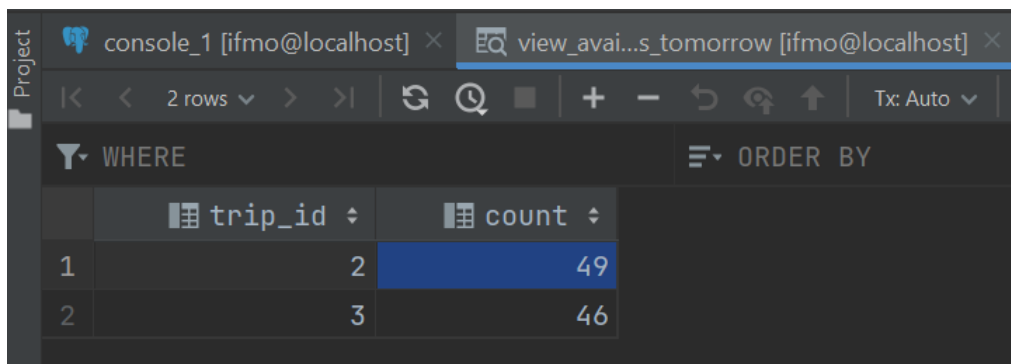
```
SELECT drivers.*
from drivers
    right join medical_checkup mc on drivers.driver_id = mc.driver_id
    inner join public.trip t on mc.trip_id = t.trip_id
where trip_status != 'canceled'::trip_status
and mc.medical_checkup_passed
and date(t.date_of_departure) >= current_date - interval '1 week'
and (t.date_of_departure - t.date_of_arrival) =
(SELECT max(cnt) from (select (date_of_departure - date_of_arrival) as
cnt
    from trip
        inner join public.medical_checkup mc on
trip.trip_id = mc.trip_id
        left join public.drivers dr on
mc.driver_id = dr.driver_id
    where trip.trip_status !=
'canceled'::trip_status
    and mc.medical_checkup_passed
    and date(trip.date_of_departure) >=
current_date - interval '1 week') as subquery)
group by drivers.driver_id;
```

	driver_id [PK] bigint	first_name character varying	last_name character varying	middle_name character varying	phone_number character varying	passport_no numeric (10)	sex user_sex	date_of_birth date	place_of_birth character varying
1	3	Михаил	Садко	Романович	+7908654322	7657435262	M	1978-01-02	Санкт-Петербург

Рисунок 13 – Результат работы запроса №7.

Представление 1. Количество свободных мест на все рейсы на завтра;

```
create view view_available_tickets_tomorrow as
SELECT trip_id,
       (select seats_count
        from type_of_bus
         right join bus on type_of_bus.type_id = bus.type_id
         right join trip t on bus.bus_number = t.bus_number
        where trip_status != 'canceled'::trip_status
          and t.trip_id = trip.trip_id) -
       sum((select count(*) from ticket as ti where ti.trip_id =
trip.trip_id))
from trip
where date(date_of_departure) = current_date + interval '1 day'
group by trip_id;
```

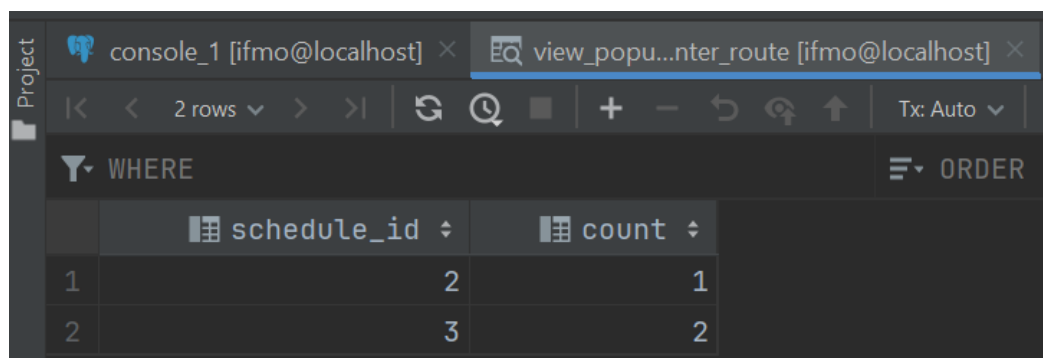


trip_id	count
1	49
2	46

Рисунок 14 – Представление №1.

Представление 2. Самый популярный маршрут этой зимой.

```
create view view_popular_winter as
SELECT trip.schedule_id, count(*)
from ticket
  right join trip on ticket.trip_id = trip.trip_id
where ticket_status = 'paid'::ticket_status
  and trip.trip_status != 'canceled'::trip_status
  and trip.date_of_departure BETWEEN
    DATE_TRUNC('year', CURRENT_DATE - INTERVAL '1 year') + INTERVAL '11
months' AND
    DATE_TRUNC('year', CURRENT_DATE) + INTERVAL '2 months'
group by trip.schedule_id;
```

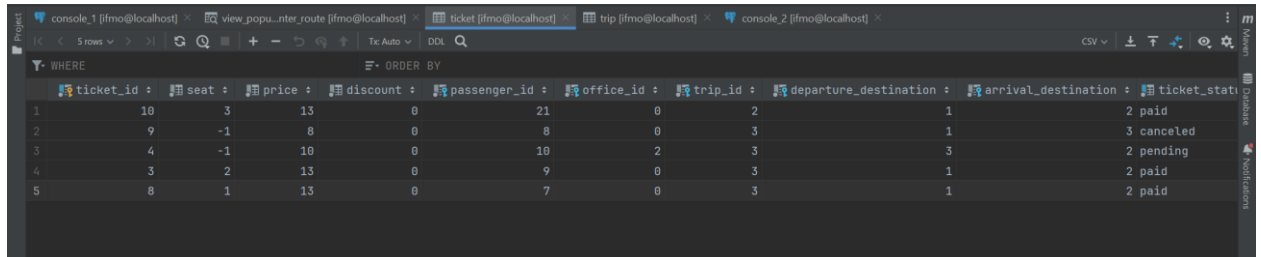


schedule_id	count
2	1
3	2

Рисунок 15 – Представление №2.

Модификация данных. Вставка.

```
INSERT INTO ticket (seat, price, discount, passenger_id, office_id, trip_id, departure_destination,
                    arrival_destination, ticket_status, date_of_transaction)
VALUES (3, 13, 0, (select passengers.passenger_id from passengers where email
= 'leia9@scout.com'),
        (SELECT office_id from ticket_office WHERE address = 'ONLINE'), 2, 1,
2, DEFAULT, DEFAULT);
```

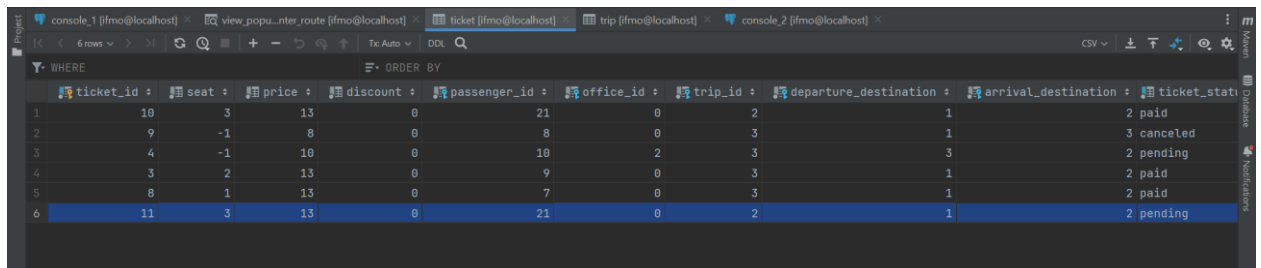


	ticket_id	seat	price	discount	passenger_id	office_id	trip_id	departure_destination	arrival_destination	ticket_status
1	10	3	13	0	21	0	2	1	2	paid
2	9	-1	8	0	8	0	3	1	3	canceled
3	4	-1	10	0	10	2	3	3	2	pending
4	3	2	13	0	9	0	3	1	2	paid
5	8	1	13	0	7	0	3	1	2	paid

Рисунок 16 – Состояние до выполнения запроса (вставка).

```
lfmo.public> INSERT INTO ticket (seat, price, discount, passenger_id, office_id, trip_id, departure_destination,
                                arrival_destination, ticket_status, date_of_transaction)
                                VALUES (3, 13, 0, (select passengers.passenger_id from passengers where email = 'leia9@scout.com'), (S
[2023-12-07 19:52:37] 1 row affected in 22 ms
```

Рисунок 17 – Запрос (вставка).

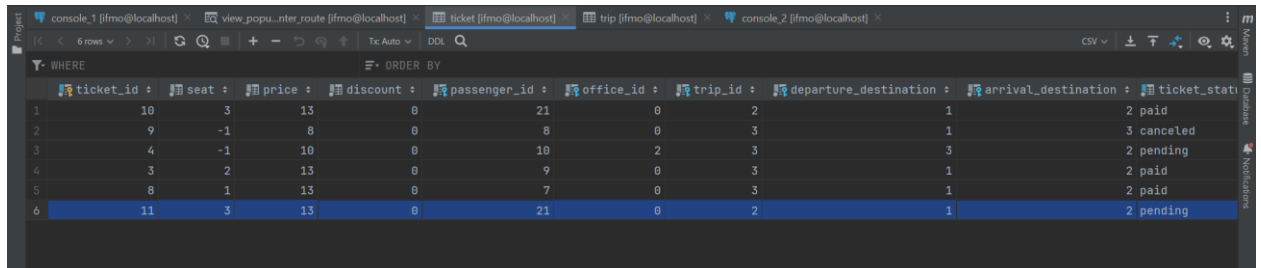


	ticket_id	seat	price	discount	passenger_id	office_id	trip_id	departure_destination	arrival_destination	ticket_status
1	10	3	13	0	21	0	2	1	2	paid
2	9	-1	8	0	8	0	3	1	3	canceled
3	4	-1	10	0	10	2	3	3	2	pending
4	3	2	13	0	9	0	3	1	2	paid
5	8	1	13	0	7	0	3	1	2	paid
6	11	3	13	0	21	0	2	1	2	pending

Рисунок 18 – Состояние после выполнения запроса (вставка).

Модификация данных. Изменение.

```
UPDATE ticket
SET ticket_status = 'paid'::ticket_status
WHERE passenger_id = (select passengers.passenger_id from passengers where
email = 'leia9@scout.com')
and ticket_status = 'pending'::ticket_status;
```

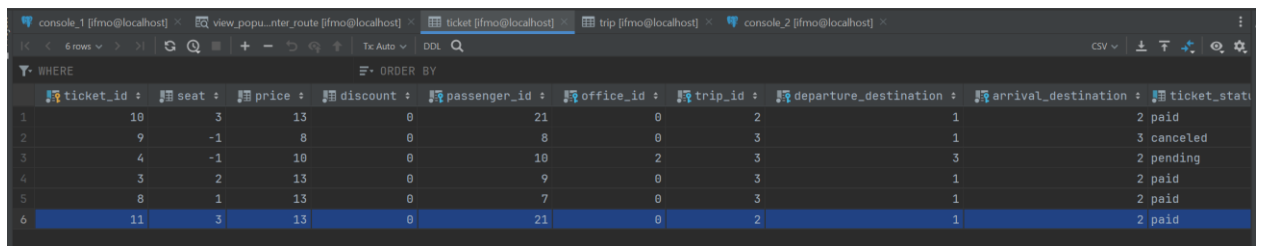


	ticket_id	seat	price	discount	passenger_id	office_id	trip_id	departure_destination	arrival_destination	ticket_status
1	10	3	13	0	21	0	2	1	2	paid
2	9	-1	8	0	8	0	3	1	3	canceled
3	4	-1	10	0	10	2	3	3	2	pending
4	3	2	13	0	9	0	3	1	2	paid
5	8	1	13	0	7	0	3	1	2	paid
6	11	3	13	0	21	0	2	1	2	pending

Рисунок 19 – Состояние до выполнения запроса (изменение).

```
(fmo_public)> UPDATE ticket
SET ticket_status = 'paid'::ticket_status
WHERE passenger_id = (select passengers.passenger_id from passengers where email = 'leia9@scout.com')
[2023-12-07 19:53:39] 1 row affected in 4 ms
```

Рисунок 20 – Запрос (изменение).

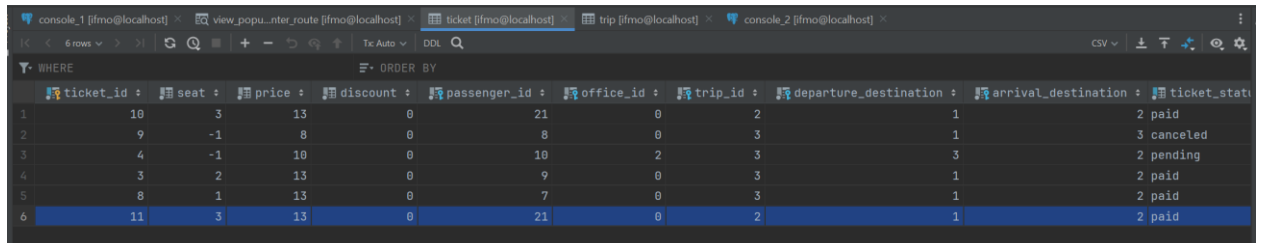


	ticket_id	seat	price	discount	passenger_id	office_id	trip_id	departure_destination	arrival_destination	ticket_status
1	10	3	13	0	21	0	2	1	2	paid
2	9	-1	8	0	8	0	3	1	3	canceled
3	4	-1	10	0	10	2	3	3	2	pending
4	3	2	13	0	9	0	3	1	2	paid
5	8	1	13	0	7	0	3	1	2	paid
6	11	3	13	0	21	0	2	1	2	paid

Рисунок 21 – Состояние до выполнения запроса (изменение).

Модификация данных. Удаление.

```
DELETE
FROM ticket
WHERE passenger_id = (select passengers.passenger_id from passengers where
email = 'leia9@scout.com')
and ticket_status = 'paid'::ticket_status;
```

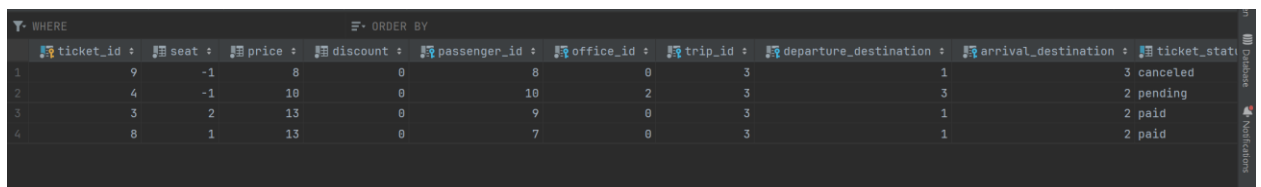


	ticket_id	seat	price	discount	passenger_id	office_id	trip_id	departure_destination	arrival_destination	ticket_status
1	10	3	13	0	21	0	2	1	2	paid
2	9	-1	8	0	8	0	3	1	3	canceled
3	4	-1	10	0	10	2	3	3	2	pending
4	3	2	13	0	9	0	3	1	2	paid
5	8	1	13	0	7	0	3	1	2	paid
6	11	3	13	0	21	0	2	1	2	paid

Рисунок 22 – Состояние до выполнения запроса (удаление).

```
ifmo.public> DELETE
FROM ticket
WHERE passenger_id = (select passengers.passenger_id from passengers where email = 'leia9@scout.com')
[2023-12-07 19:54:21] 2 rows affected in 5 ms
```

Рисунок 23 – Запрос (удаление).



	ticket_id	seat	price	discount	passenger_id	office_id	trip_id	departure_destination	arrival_destination	ticket_status
1	9	-1	8	0	8	0	3	1	3	canceled
2	4	-1	10	0	10	2	3	3	2	pending
3	3	2	13	0	9	0	3	1	2	paid
4	8	1	13	0	7	0	3	1	2	paid

Рисунок 24 – Состояние после выполнения запроса (удаление).

Создание простого индекса.

```
create index ticket_ticket_status_index
on ticket (ticket_status);
```

Создание составного индекса.

```
create index trip_trip_id_bus_number_index
on trip (trip_id, bus_number);
```

Вывод

В ходе лабораторной работы я научился создавать представления и запросы на выборку данных к базе данных PostgreSQL. В процессе лабораторной работы был изучен новый синтаксис PostgreSQL. Были написаны требуемые SQL запросы. Также было сравнено время выполнения запроса с индексами и без: с индексами быстрее.