

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования «Национальный исследовательский университет ИТМО»  
Факультет инфокоммуникационных технологий

Лабораторная работа №2  
«Запросы на выборку и модификацию данных, представления и  
индексы в PostgreSQL»

По дисциплине  
«Проектирование и реализация баз данных»

Выполнил:  
Кириллова В.Е  
Группа:  
К3240  
Преподаватель:  
Говорова М.М.



Санкт-Петербург  
2022 г

## 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 (Вариант 8. БД «Аэропорт»)

#### Задание 1. Создайте запросы:

- Определить расчетное время полета по всем маршрутам.
- Определить расход топлива по всем маршрутам.
- Вывести данные о том, сколько свободных мест оставалось в самолетах, совершавших полет по заданному из рейсов за вчерашний день.
- Рассчитать убытки компании за счет непроданных билетов за вчерашний день.
- Определить, какой тип самолетов чаще всего летал в заданный аэропорт назначения.
- Вывести список самолетов, “возраст” которых превышает средний “возраст” самолетов этого типа.
- Определить тип самолетов, летающих во все аэропорты назначения.

#### Задание 2. Создайте представления:

- для пассажиров авиакомпании о рейсах в Москву на ближайшую неделю;
- количество самолетов каждого типа, летавшими за последний месяц.

#### Задание 3.

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

#### Задание 4.

Изучить графическое представление запросов и просмотреть историю запросов

#### Задание 5.

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.

# СХЕМА БАЗЫ ДАННЫХ

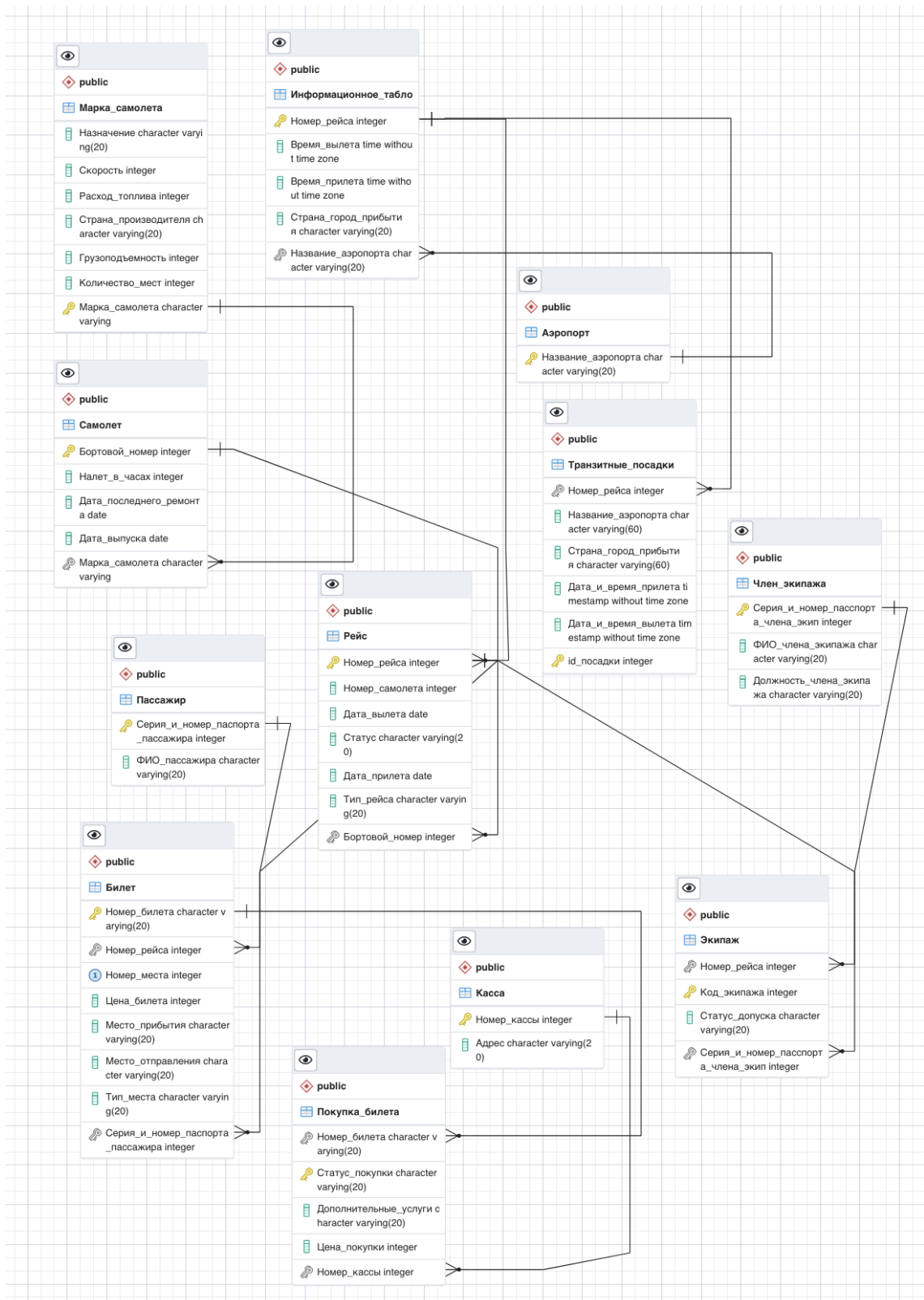


Рисунок 1 - Схема логической модели данных

## ВЫПОЛНЕНИЕ

### Запросы к базе данных

1. Определить расчетное время полета по всем маршрутам.

```
select extract(epoch from sum(Время_полета - Пересадка))/3600 as
Время_в_воздухе
from
(
    select Рейс.Номер_рейса, Рейс.Бортовой_номер, (Дата_прилета +
Время_прилета) - (Дата_вылета + Время_вылета) as Время_полета,
sum(Дата_и_время_вылета - Дата_и_время_прилета) as Пересадка from
Информационное_табло
    inner join Рейс on Рейс.Номер_рейса =
Информационное_табло.Номер_рейса
    inner join Транзитные_посадки on Транзитные_посадки.Номер_рейса =
Информационное_табло.Номер_рейса
    group by Рейс.Номер_рейса, Время_полета
)
as Таблица
```

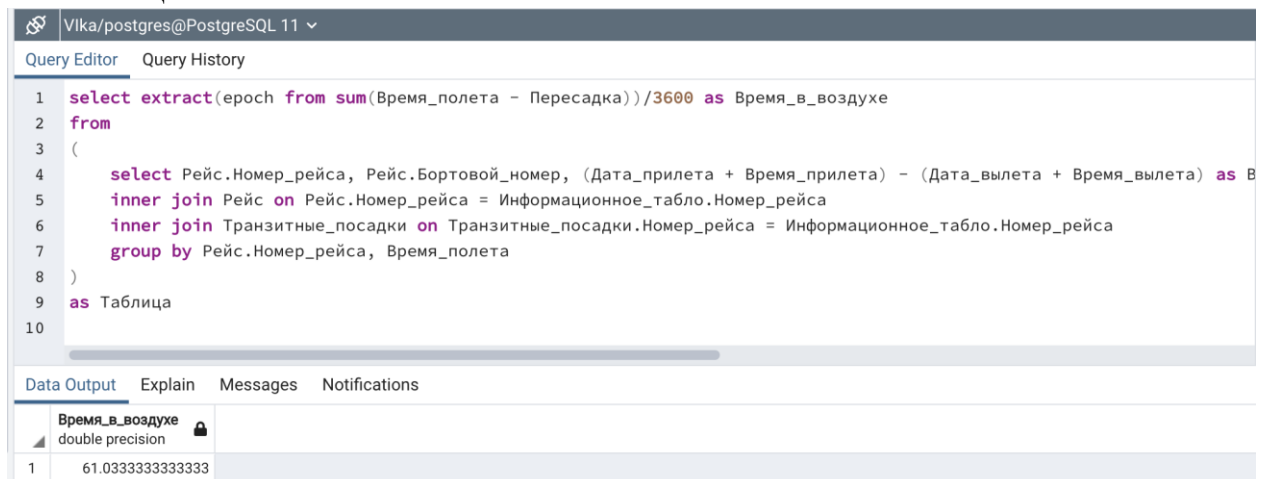
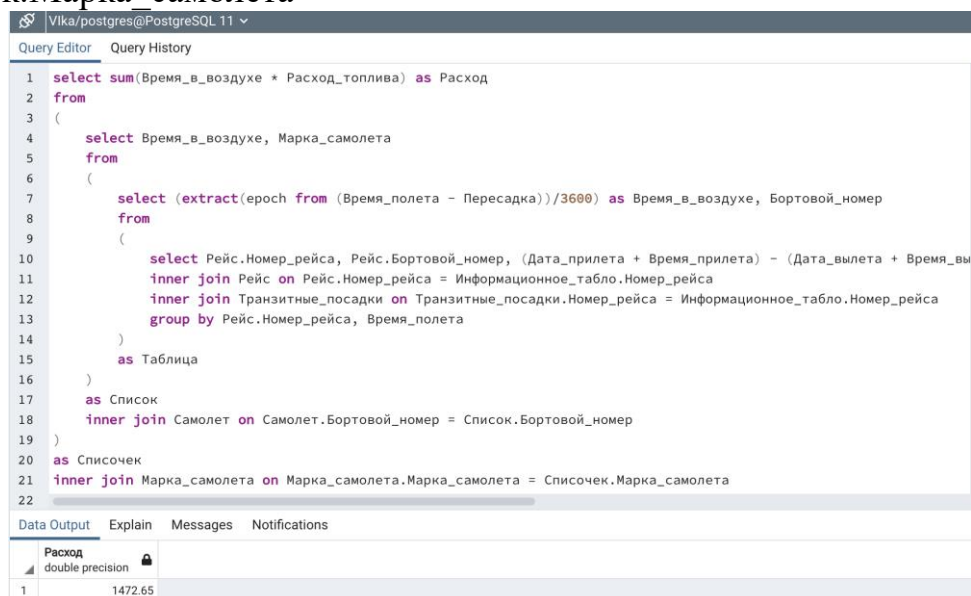


Рисунок 2

## 2. Определить расход топлива по всем маршрутам.

```
select sum(Время_в_воздухе * Расход_топлива) as Расход
from
(
    select Время_в_воздухе, Марка_самолета
    from
    (
        select (extract(epoch from (Время_полета - Пересадка)))/3600) as
Время_в_воздухе, Бортовой_номер
    from
    (
        select Рейс.Номер_рейса, Рейс.Бортовой_номер,
(Дата_прилета + Время_прилета) - (Дата_вылета + Время_вылета) as
Время_полета, sum(Дата_и_время_вылета - Дата_и_время_прилета) as
Пересадка from Информационное_табло
        inner join Рейс on Рейс.Номер_рейса =
Информационное_табло.Номер_рейса
        inner join Транзитные_посадки on
Транзитные_посадки.Номер_рейса = Информационное_табло.Номер_рейса
        group by Рейс.Номер_рейса, Время_полета
    )
    )
    as Таблица
)
as Список
inner join Самолет on Самолет.Бортовой_номер =
Список.Бортовой_номер
)
as Списочек
inner join Марка_самолета on Марка_самолета.Марка_самолета =
Списочек.Марка_самолета
```



The screenshot shows a PostgreSQL Query Editor window with a query that calculates the total fuel consumption across all routes. The query is a multi-level nested SELECT statement. It starts with a subquery that calculates the time in the air for each flight segment by subtracting the stopover time from the total flight time. This is then joined with the flight schedule table, the transit stop table, and the aircraft table to calculate the total fuel consumption for each route. The final result is a single row showing the total fuel consumption for all routes.

Расход
1472.65

Рисунок 3

3. Вывести данные о том, сколько свободных мест оставалось в самолетах, совершавших полет по заданному из рейсов за вчерашний день.

```
select Количество_мест - (select count(Номер_рейса) from Билет where
Номер_рейса = 591) from Марка_самолета where Марка_самолета = (select
Марка_самолета from Самолет where Бортовой_номер = (select
Бортовой_номер from Рейс where Номер_рейса = 591))
```

Vika/postgres@PostgreSQL 11

Query Editor

Query History

1

select

Количество\_мест - (select count(Номер\_рейса) from Билет where Номер\_рейса = 591)

2

from

Марка\_самолета where Марка\_самолета = (select Марка\_самолета

3

from

Самолет where Бортовой\_номер = (select Бортовой\_номер from Рейс where Номер\_рейса = 591))

Data Output

Explain

Messages

Notifications

?column?

bigint

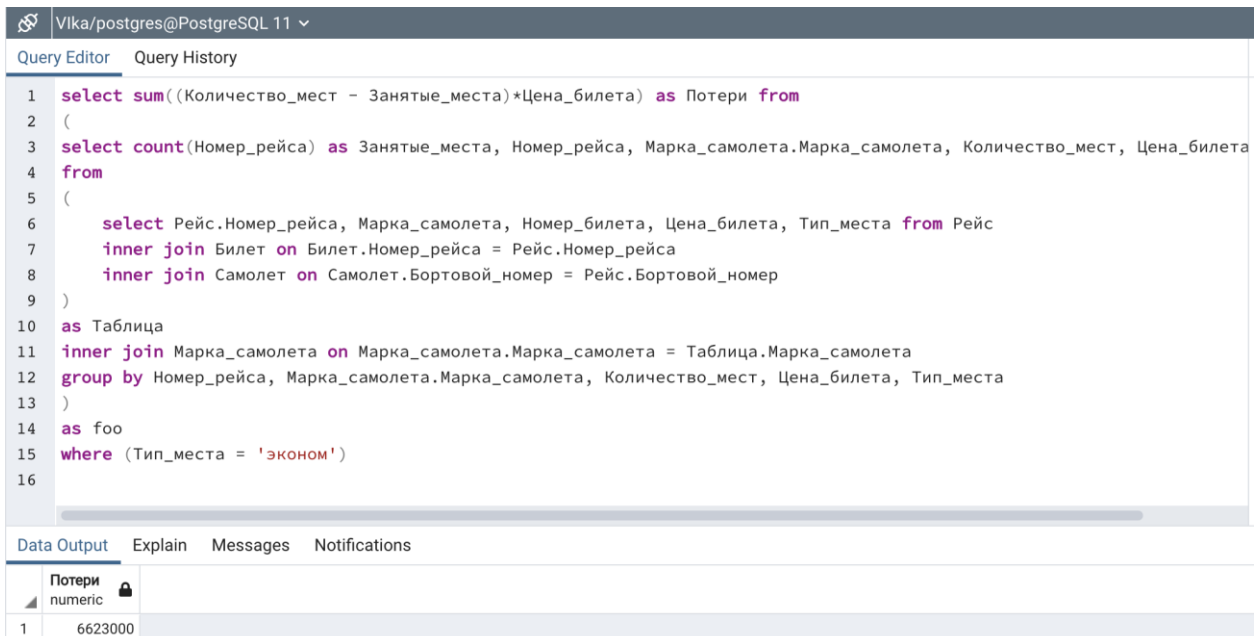
1

187

Рисунок 4

4. Рассчитать убытки компании за счет не проданных билетов за вчерашний день.

```
select sum((Количество_мест - Занятые_места)*Цена_билета) as Потери from
(
select count(Номер_рейса) as Занятые_места, Номер_рейса,
Марка_самолета.Марка_самолета, Количество_мест, Цена_билета,
Тип_места
from
(
select Рейс.Номер_рейса, Марка_самолета, Номер_билета,
Цена_билета, Тип_места from Рейс
inner join Билет on Билет.Номер_рейса = Рейс.Номер_рейса
inner join Самолет on Самолет.Бортовой_номер = Рейс.Бортовой_номер
where(Дата_отправления = 'yestarday'::date)
)
as Таблица
inner join Марка_самолета on Марка_самолета.Марка_самолета =
Таблица.Марка_самолета
group by Номер_рейса, Марка_самолета.Марка_самолета, Количество_мест,
Цена_билета, Тип_места
)
as foo
where (Тип_места = 'эконом')
```



The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 select sum((Количество_мест - Занятые_места)*Цена_билета) as Потери from
2 (
3 select count(Номер_рейса) as Занятые_места, Номер_рейса, Марка_самолета.Марка_самолета, Количество_мест, Цена_билета
4 from
5 (
6 select Рейс.Номер_рейса, Марка_самолета, Номер_билета, Цена_билета, Тип_места from Рейс
7 inner join Билет on Билет.Номер_рейса = Рейс.Номер_рейса
8 inner join Самолет on Самолет.Бортовой_номер = Рейс.Бортовой_номер
9 )
10 as Таблица
11 inner join Марка_самолета on Марка_самолета.Марка_самолета = Таблица.Марка_самолета
12 group by Номер_рейса, Марка_самолета.Марка_самолета, Количество_мест, Цена_билета, Тип_места
13 )
14 as foo
15 where (Тип_места = 'эконом')
16
```

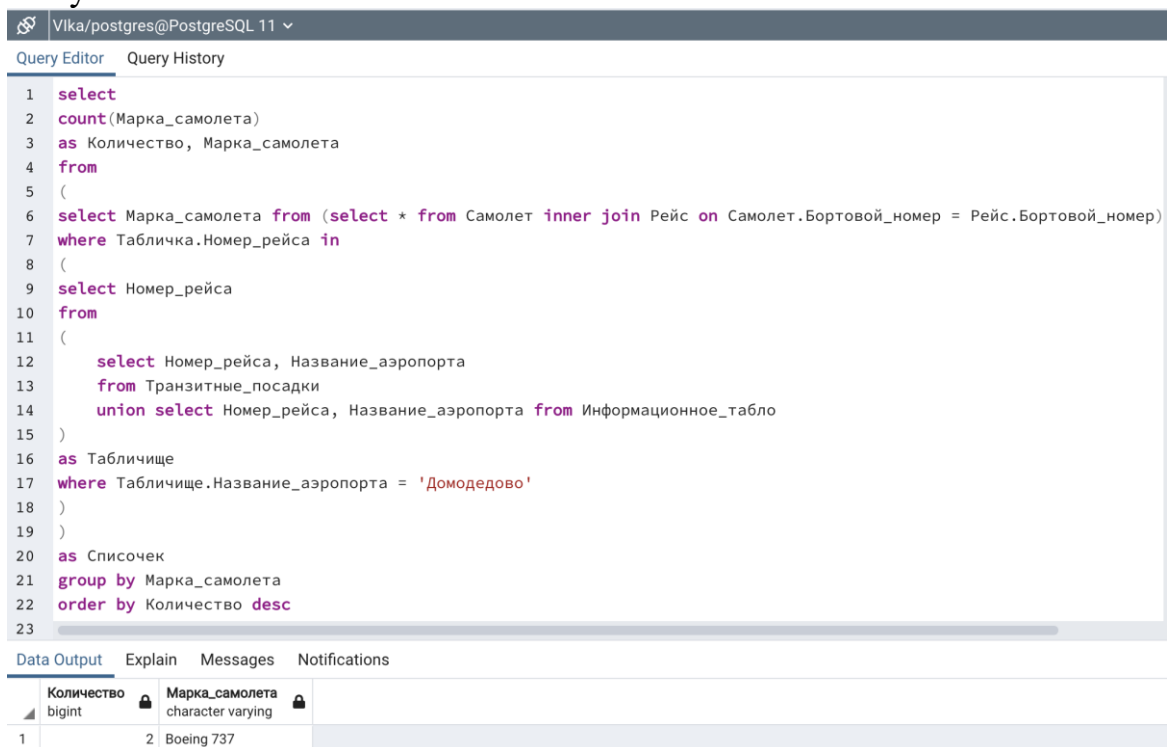
The Data Output tab shows the result of the query:

Потери
6623000

Рисунок 5

5. Определить, какой тип самолетов чаще всего летал в заданный аэропорт назначения.

```
select
count(Марка_самолета) as Количество, Марка_самолета
from
(
select Марка_самолета from (select * from Самолет inner join Рейс on
Самолет.Бортовой_номер = Рейс.Бортовой_номер) as Табличка
where Табличка.Номер_рейса in
(
select Номер_рейса
from
(
select Номер_рейса, Название_аэропорта
from Транзитные_посадки
union select Номер_рейса, Название_аэропорта from
Информационное_табло
)
)
as Табличке
where Табличке.Название_аэропорта = 'Домодедово'
)
)
as Списочек
group by Марка_самолета
order by Количество desc
```



The screenshot shows a PostgreSQL query editor with the following query:

```
1 select
2 count(Марка_самолета)
3 as Количество, Марка_самолета
4 from
5 (
6 select Марка_самолета from (select * from Самолет inner join Рейс on
7 Самолет.Бортовой_номер = Рейс.Бортовой_номер) as Табличка
8 where Табличка.Номер_рейса in
9 (
10 select Номер_рейса
11 from
12 (
13 select Номер_рейса, Название_аэропорта
14 from Транзитные_посадки
15 union select Номер_рейса, Название_аэропорта from
16 Информационное_табло
17 )
18 )
19 as Табличке
20 where Табличке.Название_аэропорта = 'Домодедово'
21 )
22 )
23 as Списочек
24 group by Марка_самолета
25 order by Количество desc
```

The Data Output tab shows the following results:

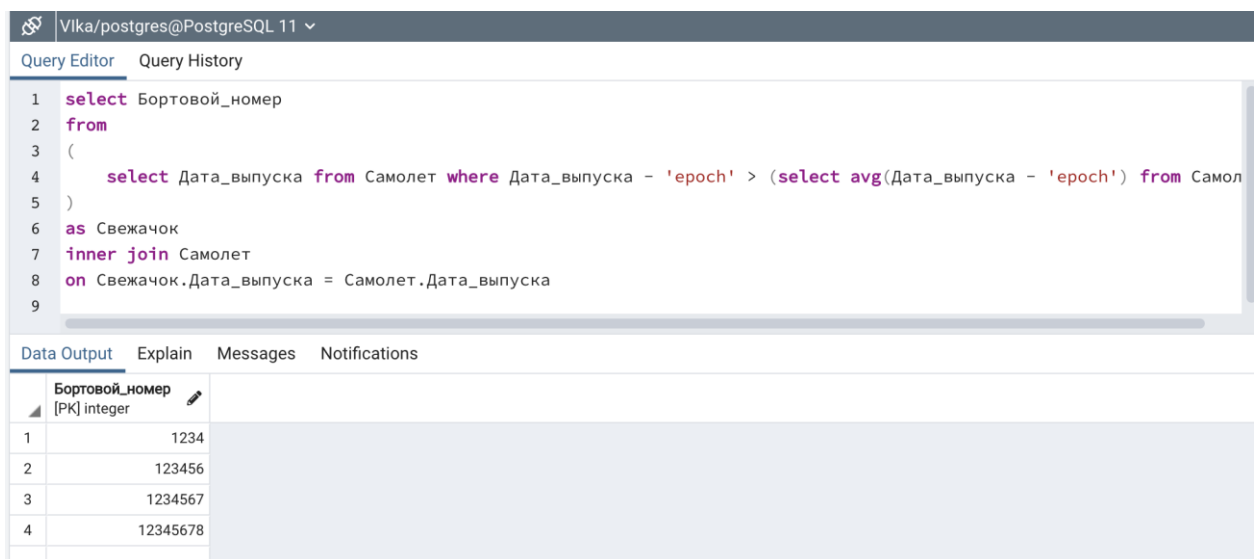
Количество bigint	Марка_самолета character varying
1	Boeing 737

Рисунок 6



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

```
select Бортовой_номер
from
(
    select Дата_выпуска from Самолет where Дата_выпуска - 'epoch' >
(select avg(Дата_выпуска - 'epoch') from Самолет)
)
as Свежачок
inner join Самолет
on Свежачок.Дата_выпуска = Самолет.Дата_выпуска
```



The screenshot shows a PostgreSQL query editor interface. The top bar indicates the user is 'Vika/postgres@PostgreSQL 11'. Below the bar are tabs for 'Query Editor' and 'Query History'. The 'Query Editor' tab is active, displaying a SQL query. Below the query editor are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the results of the query. The table has one column, 'Бортовой\_номер', which is an integer. The results show four rows with values 1234, 123456, 1234567, and 12345678.

Бортовой_номер [PK] integer
1
2
3
4

Рисунок 7

7. Определить тип самолетов, летающих во все аэропорты назначения.

```
select Марка_самолета
from
(
    select count(Название_аэропорта) as Количество_аэропортов,
Марка_самолета
    from Рейс
    inner join Информационное_табло on
Информационное_табло.Номер_рейса = Рейс.Номер_рейса
    inner join Самолет on Самолет.Бортовой_номер = Рейс.Бортовой_номер
    group by Марка_самолета
)
as foo
where (Количество_аэропортов = (select count(Название_аэропорта) from
Информационное_табло))
```

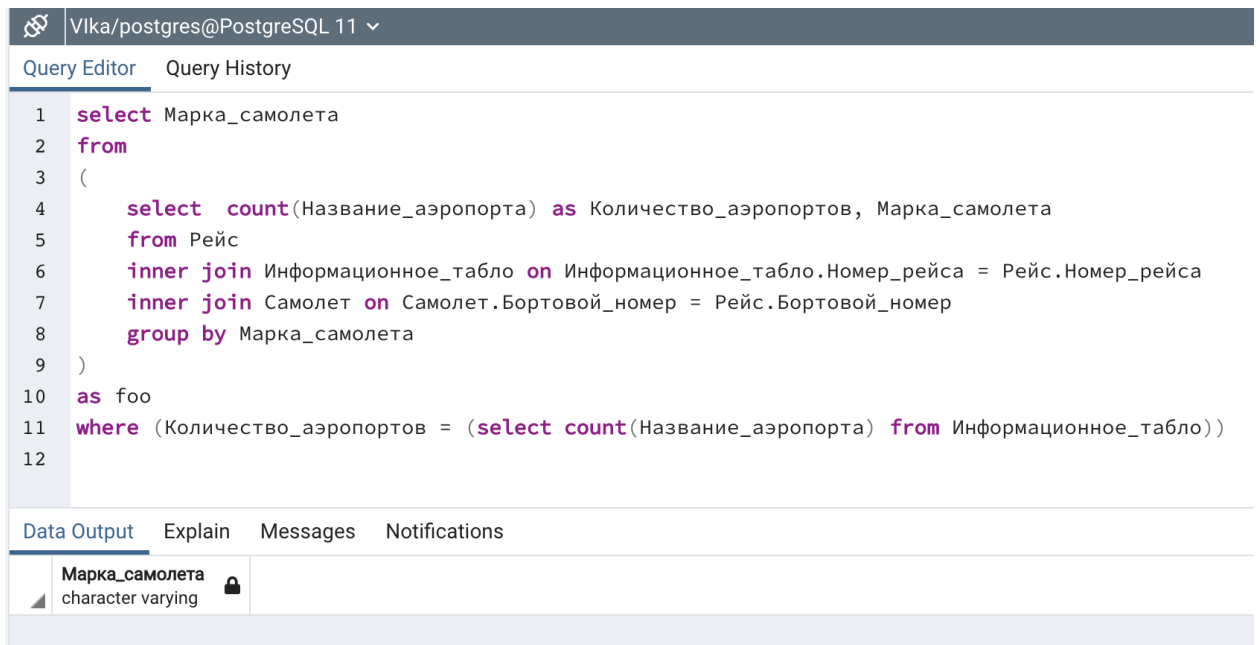


Рисунок 8

## ПРЕДСТАВЛЕНИЯ

1. Для пассажиров авиакомпании о рейсах в Москву на ближайшую неделю

```
create view Рейсы_в_москву as
(
select Рейс.Номер_рейса, Страна_город_прибытия, Дата_вылета,
Время_вылета from Рейс
inner join Информационное_табло on
Информационное_табло.Номер_рейса = Рейс.Номер_рейса
where Дата_вылета > ('now'::date - '1 month'::interval) and
Страна_город_прибытия = 'Россия, Москва'
)
```

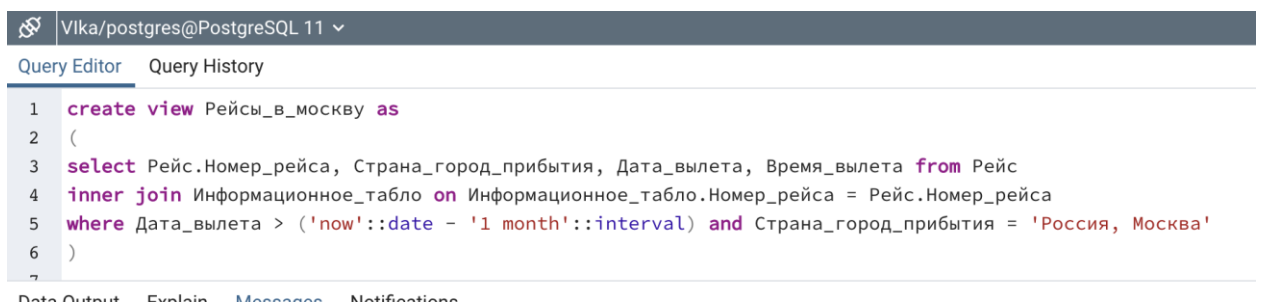


Рисунок 9

2. Количество самолетов каждого типа, летавшими за последний месяц.

```
create view Летящие_самолеты as
(
select count(Бортовой_номер), Марка_самолета.Марка_самолета
from
(
select Рейс.Бортовой_номер, Марка_самолета from Рейс
inner join Самолет on Самолет.Бортовой_номер =
Рейс.Бортовой_номер
where Дата_вылета > ('2022-05-01')
)
as foo
inner join Марка_самолета on Марка_самолета.Марка_самолета =
foo.Марка_самолета
group by Марка_самолета.Марка_самолета
)
```

Viika/postgres@PostgreSQL 11

Query EditorQuery History

```
1 create view Летящие_самолеты as
2 (
3 select count(Бортовой_номер), Марка_самолета.Марка_самолета
4 from
5 (
6 select Рейс.Бортовой_номер, Марка_самолета from Рейс
7     inner join Самолет on Самолет.Бортовой_номер = Рейс.Бортовой_номер
8     where Дата_вылета > ('2022-05-01')
9 )
10 as foo
11 inner join Марка_самолета on Марка_самолета.Марка_самолета = foo.Марка_самолета
12 group by Марка_самолета.Марка_самолета
13 )
```

Data OutputExplainMessagesNotifications

CREATE VIEW

Query returned successfully in 67 msec.

Рисунок 10

## Запросы на модификацию данных

DELETE INSERT UPDATE

### INSERT

Добавляем ФИО и паспортные данные члена экипажа в таблицу Пассажир

```
insert into Пассажир
values
(
    (select Серия_и_номер_паспорта_члена_экип from Член_экипажа where
Серия_и_номер_паспорта_члена_экип = '1111111111'),
    (select ФИО_члена_экипажа from Член_экипажа where
Серия_и_номер_паспорта_члена_экип = '1111111111')
)
```

### DELETE

Удаляем ФИО и паспортные данные члена экипажа из таблицы Пассажир

```
delete from Пассажир
where
(
    Серия_и_номер_паспорта_пассажира =
    (select Серия_и_номер_паспорта_члена_экип from Член_экипажа where
ФИО_члена_экипажа = 'Петров П.П.')
)
```

### UPDATE

Увеличиваем цену 12-го билета в 2 раза

```
update Билет
set Цена_билета =
(
    select Цена_билета*2
    from Билет
    where Номер_билета = '12'
)
where Номер_билета = '12'
```

## ИНДЕКСЫ

1)

До индекса – выполнение 88 мсек

Query Editor   Query History

```
1 select ФИО_пассажира, Адрес
2 from
3 (
4     select ФИО_пассажира, Номер_кассы from Билет
5     inner join Пассажир on Пассажир.Серия_и_номер_паспорта_пассажира = Билет.Серия_и_номер_паспорта_пассажира
6     inner join Покупка_билета on Билет.Номер_билета = Покупка_билета.Номер_билета
7 )
8 as Табличка
9 inner join Касса on Касса.Номер_кассы = Табличка.Номер_кассы
10 where (Адрес like 'у%')
11
12
```

Data Output   Explain   Messages   Notifications

Successfully run. Total query runtime: 88 msec.  
1 rows affected.

Индекс: create index Адрес\_ид on Касса(Адрес)

После – выполнение 35 мсек

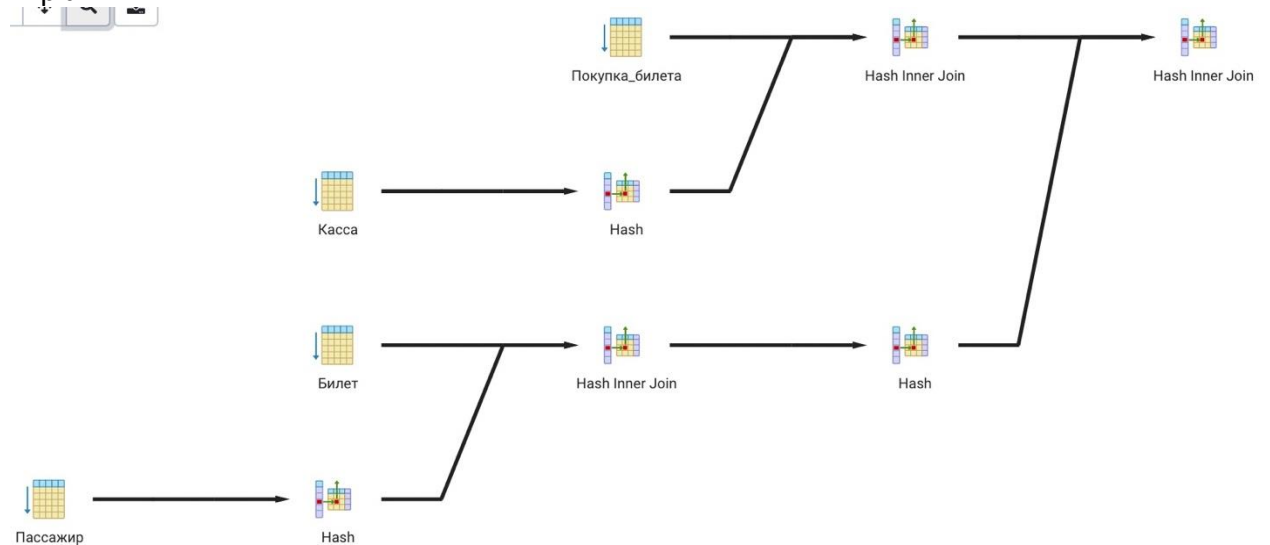
Query Editor   Query History

```
1 select ФИО_пассажира, Адрес
2 from
3 (
4     select ФИО_пассажира, Номер_кассы from Билет
5     inner join Пассажир on Пассажир.Серия_и_номер_паспорта_пассажира = Билет.Серия_и_номер_паспорта_пассажира
6     inner join Покупка_билета on Билет.Номер_билета = Покупка_билета.Номер_билета
7 )
8 as Табличка
9 inner join Касса on Касса.Номер_кассы = Табличка.Номер_кассы
10 where (Адрес like 'у%')
11
12
```

Data Output   Explain   Messages   Notifications

Successfully run. Total query runtime: 35 msec.  
1 rows affected.

Explain:



2)

До индекса – выполнение 66 мсек

```
1 select Номер_места, Место_прибытия
2 from
3 (
4     select Номер_рейса from Самолет
5     inner join Рейс on Рейс.Бортовой_номер = Самолет.Бортовой_номер
6     inner join Марка_самолета on Марка_самолета.Марка_самолета = Самолет.Марка_самолета
7     where (Марка_самолета.Марка_самолета = (select Марка_самолета from Марка_самолета order by Скорость desc limit
8 )
9 as Табл
10 inner join Билет on Билет.Номер_рейса = Табл.Номер_рейса
11 where (Номер_места > 24)
12
13
```

Data Output Explain Messages Notifications

Successfully run. Total query runtime: 66 msec.  
6 rows affected.

Составной индекс: create index Билет\_ид on Билет(Номер\_места, Место\_прибытия)

После – выполнение 37 мсек

```
1 select Номер_места, Место_прибытия
2 from
3 (
4     select Номер_рейса from Самолет
5     inner join Рейс on Рейс.Бортовой_номер = Самолет.Бортовой_номер
6     inner join Марка_самолета on Марка_самолета.Марка_самолета = Самолет.Марка_самолета
7     where (Марка_самолета.Марка_самолета = (select Марка_самолета from Марка_самолета order by Скорость desc limit
8 )
9 as Табл
10 inner join Билет on Билет.Номер_рейса = Табл.Номер_рейса
11 where (Номер_места > 24)
12
13
```

Data Output Explain Messages Notifications

Successfully run. Total query runtime: 37 msec.  
6 rows affected.

Explain:

