Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Новосибирский государственный технический университет»

Кафедра вычислительной техники

**Отчет**

По лабораторной работе №5

По дисциплине: «Базы данных»

Индексы и язык запросов SQL

Вариант 10

Выполнили: Павлюк А.С. Преподаватель: Харюткина С. А.

Казанцев К. О.

Группа: АВТ-008

Факультет: АВТ

Новосибирск, 2022

Оглавление

[Цель работы 3](#_Toc113792942)

[Задание 3](#_Toc113792943)

[Структура базы данных 5](#_Toc113792944)

[Ход работы 5](#_Toc113792945)

[Вывод 13](#_Toc113792946)

[Приложение 13](#_Toc113792947)

Цель работы

Изучение индексов в PostgreSQL и их влияния на скорость поиска информации в базе данных. Изучение команды EXPLAIN. Получение навыков по работе с командой SELECT в среде PostgreSQL.

Задание

Ознакомиться с теоретическими сведениями о возможностях создания индексов и ограничений в среде PostgreSQL. С помощью функции add\_n(int) добавить в какую-либо из таблиц, созданных в лабораторной работе No1, 1000 записей (таблица не должно содержать полей типа date, но должно быть хотя бы одно поле типа char).

Выполнить команду EXPLAIN ANALYSE select \* from таблица where поле(условие)значение. Здесь таблица – имя таблицы, в которую добавлялись записи, поле – имя поля, по которому осуществляется выборка, условие – поочередно каждое из условий =, <,>, between, значение – значение поля, являющееся критерием выборки. Проанализировать полученную после выполнения операции информацию.

Создать индекс типа BTREE для поля, по которому ранее осуществлялась выборка. Выполнить команду EXPLAIN ANALYSE с теми же условиями, что и без индекса, проанализировать полученную после выполнения операции информацию, сравнить результаты выполнения операции по каждому из условий выборки. Удалить индекс.

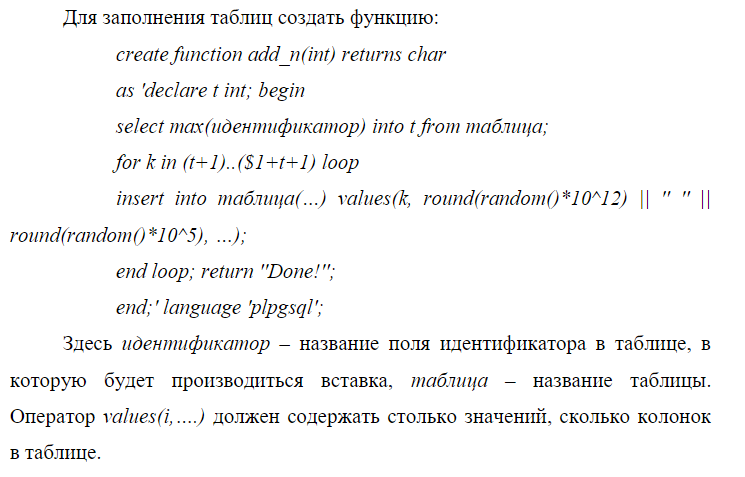
Создать индекс типа HASH для того же поля. Выполнить команду EXPLAIN ANALYSE с теми же условиями, что и ранее, проанализировать полученную после выполнения операции информацию, сравнить результаты выполнения операции по каждому из условий выборки. Удалить индекс.

Выполнить команду EXPLAIN ANALYSE select \* from таблица where upper(поле)(условие)значение, условия те же, что и ранее, значение вводится большими буквами, поле должно быть типа char. По полю типа char той же таблицы создать функциональный индекс с функцией upper(поле). Выполнить команду EXPLAIN ANALYSE с теми же условиями, что и ранее, проанализировать полученную после выполнения операции информацию, сравнить результаты выполнения операции по каждому из условий выборки с информацией, полученной при выборке по этому полю без индекса.

Наложить на выбранные поля идентификаторов в таблицах ограничение уникальности. Наложить ограничения согласно своему варианту. Проверить работоспособность ограничений путем добавления в таблицы данных, удовлетворяющих и не удовлетворяющих условиям ограничений. Ознакомиться с теоретическими сведениями о способах выборки данных из базы командой SELECT.

Выполнить запросы согласно своему варианту. Если в базе нет данных, удовлетворяющих условиям запроса, изменить условия, либо добавить подходящие данные. Просмотреть и проанализировать полученную в результате выполнения операций информацию.

Сохранить базу в файл для использования в следующих работах.



*Вариант 10:* вывести информацию обо всех руководителей туров, выполняющие туры типа «авиа» в город Москву. Найти все туры заданной длительности и стоимостью от 5000 руб. до 25000 руб. Найти все автобусные туры, которые начинаются в ближайший месяц. Найти все железнодорожные туры, осуществленные руководителями туров со стажем работы от 5 до 10 лет. Найти все туры типа «авиа» и «автобусные» в страны Польша и Германия для групп с количеством более 20 человек.

Структура базы данных

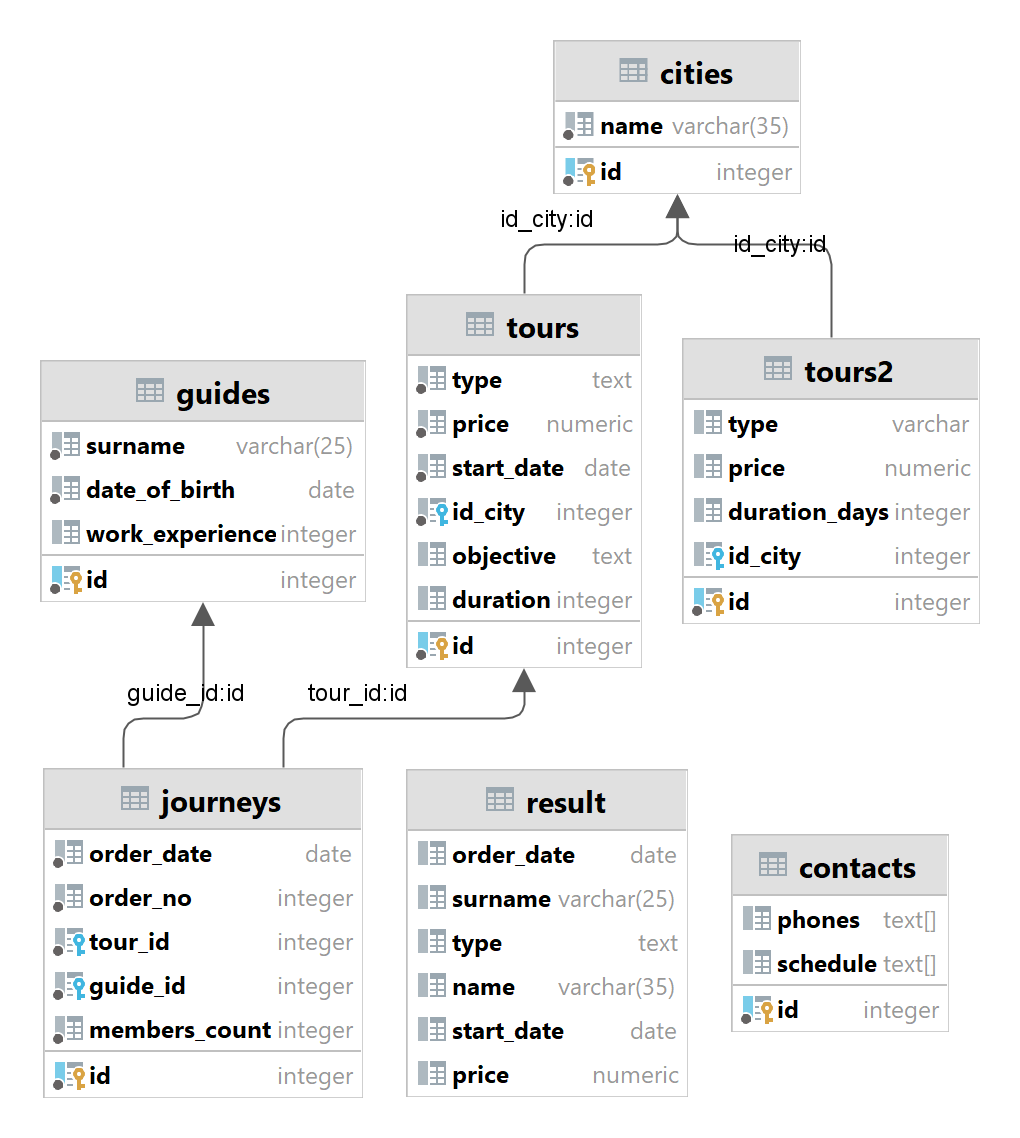
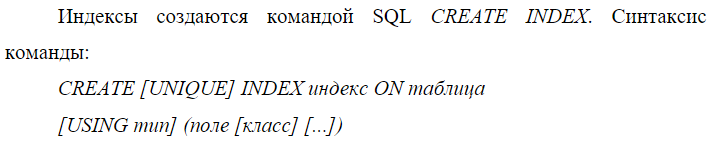


Рисунок 1. Структура базы данных

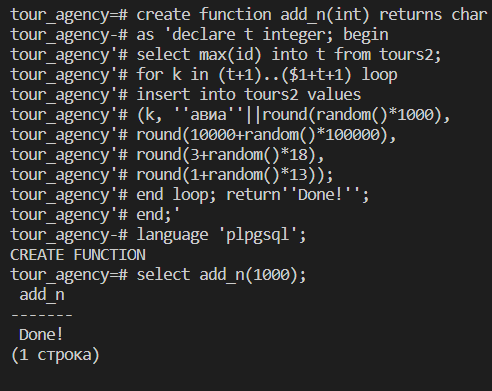
Ход работы

Индексом называется объект базы данных, позволяющий значительно повысить скорость обращения к базе за счет ускоренной обработки команд, содержащих сравнительные критерии. Хранимая в индексах информация о размещении данных по одному или нескольким полям таблицы повышает эффективность поиска записей при условной выборке.

Индексы ускоряют доступ к базе, но они также сопряжены с дополнительными затратами ресурсов. При изменении данных поле индекс приходится обновлять, поэтому поддержание редко используемых индексов отрицательно сказывается на быстродействии системы (затраты времени на поддержание индексов превышают экономию от их использования). Как правило, индексы определяются только для полей, часто указываемых в условиях поиска.



Создадим функцию add\_n(int)



Создадим копию таблицы tours, заменив дату начала тура на продолжительность в днях. Далее будем проводить анализ поиска по полю price.

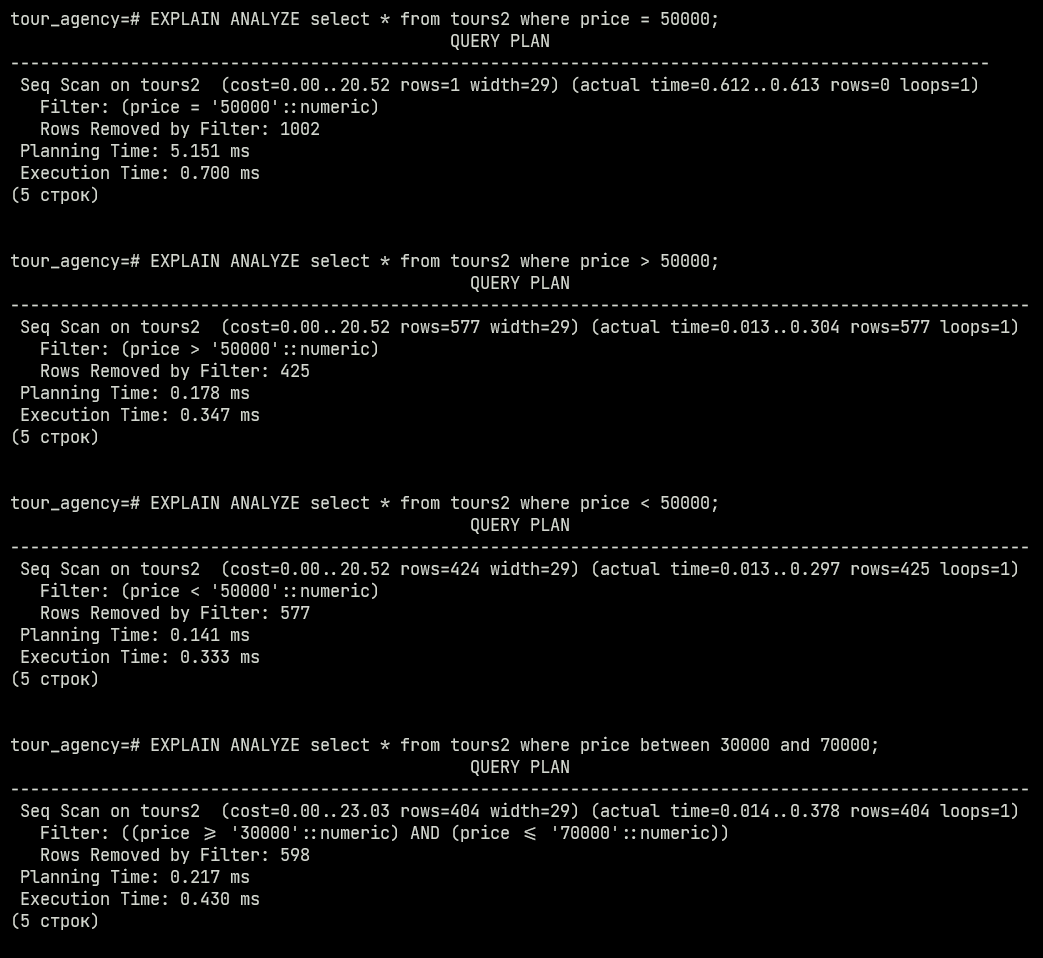


Рисунок 2. поиск без индекса

Далее добавим индекс типа BTREE и запустим поиск.

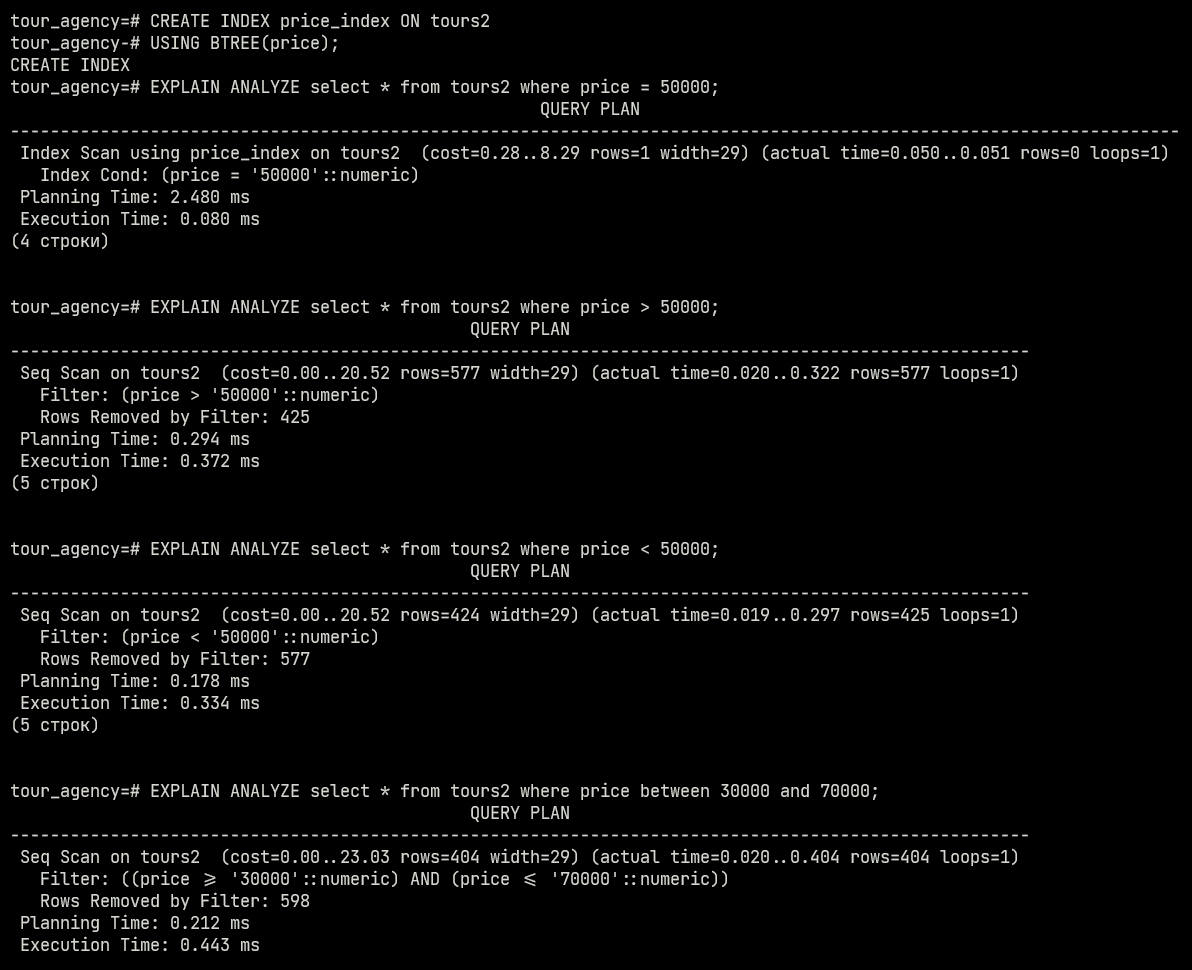


Рисунок 3. поиск с индексом btree

Удалим индекс btree и создадим hash-индекс.

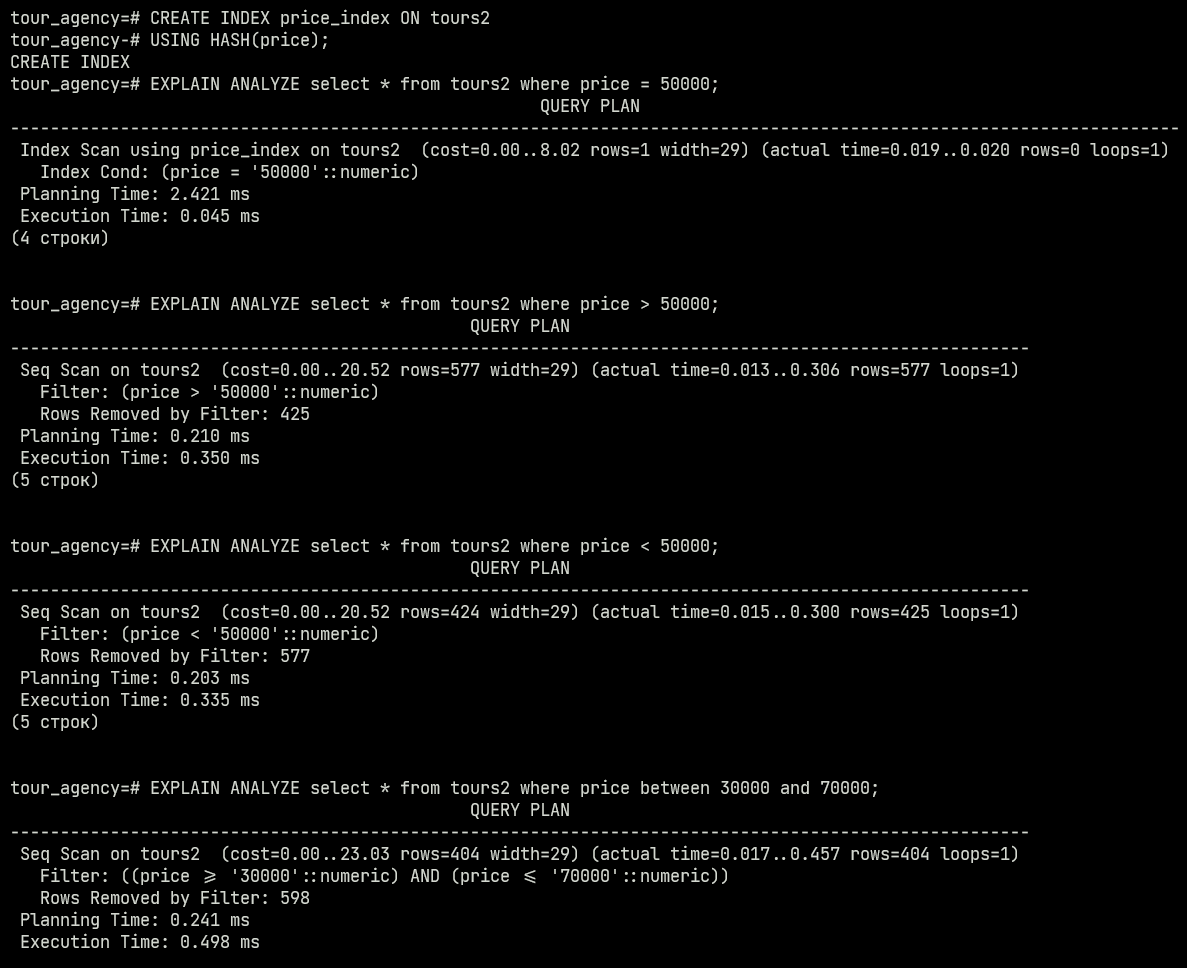


Рисунок 4. Поиск с индексом hash

Для анализа составим сравнительную таблицу.

Таблица 1. Время выполнения запросов (int)

|  |  |  |  |
| --- | --- | --- | --- |
| Параметр поиска | Без индекса | BTREE | HASH |
| Price = 5000 | 0.700 | 0.080 | 0.045 |
| Price > 5000 | 0.347 | 0.372 | 0.350 |
| Price < 5000 | 0.333 | 0.334 | 0.335 |
| 30000 <= price <= 70000 | 0.430 | 0.443 | 0.498 |

Из таблицы видно, что точечный поиск проходит значительно быстрее с индексами, причем hash-индекс оказался лучше btree-индекса.

Теперь сравним поиск по полю типа varchar.

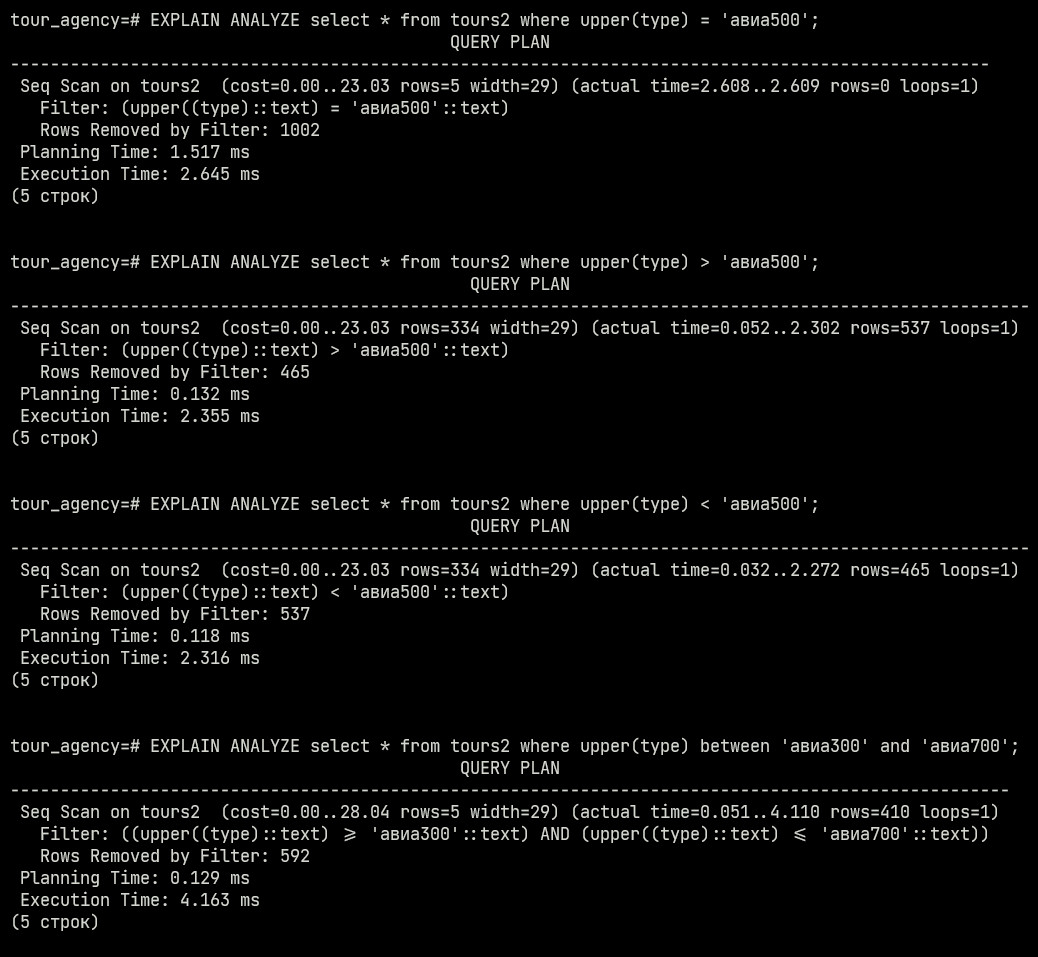


Рисунок 5. Без индексов varchar

Добавим функциональный индекс.

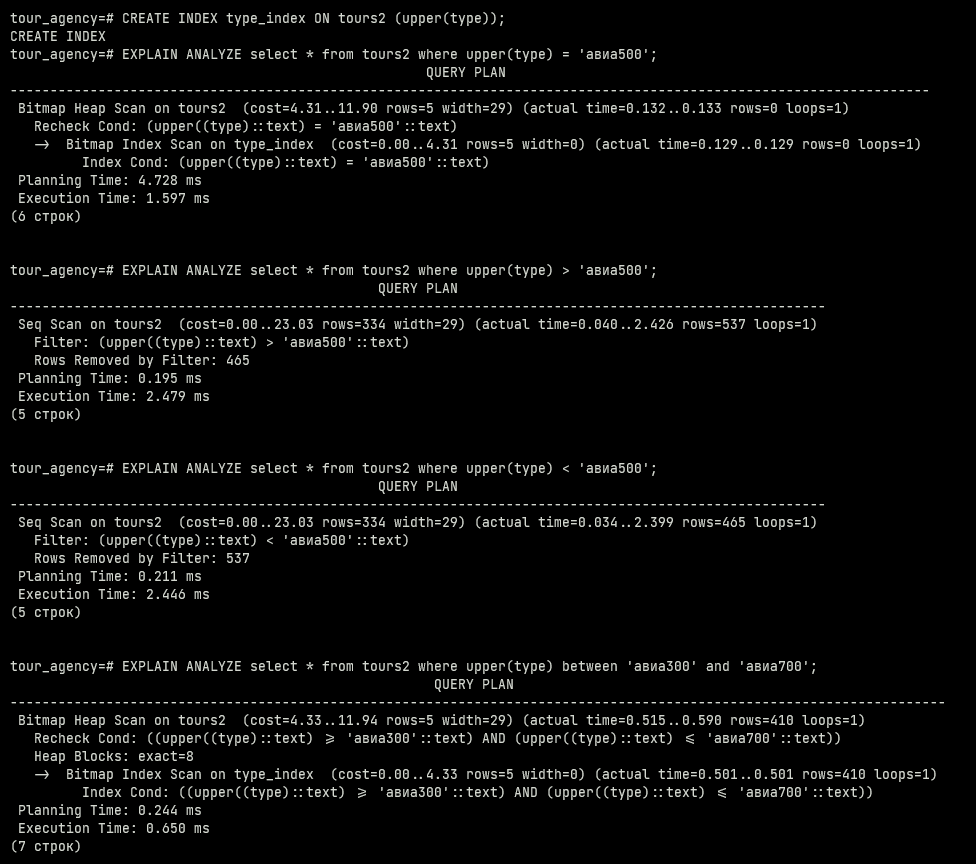


Рисунок 6. С функциональным индексом

Для анализа составим сравнительную таблицу.

Таблица 2. Время выполнения запросов (varchar)

|  |  |  |
| --- | --- | --- |
| Параметр поиска | Без индекса | Функциональный |
| Type = ‘авиа500’ | 2.645 | 1.597 |
| Type > ‘авиа500’ | 2.355 | 2.479 |
| Type < ‘авиа500’ | 2.316 | 2.446 |
| ‘авиа300’ <= type <= ‘авиа700’ | 4.163 | 0.650 |

Видно преимущество индексов при выполнении операции с BETWEEN.

Далее добавим ограничение уникальности на идентификатор.

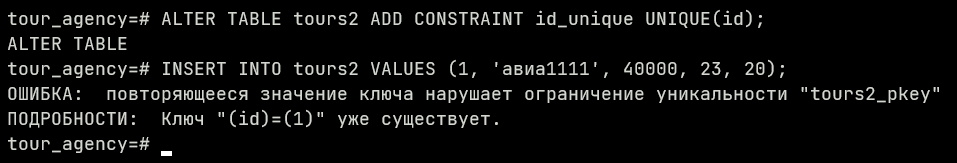


Рисунок 7. Создание ограничения уникальности

Теперь выполним запросы по варианту.

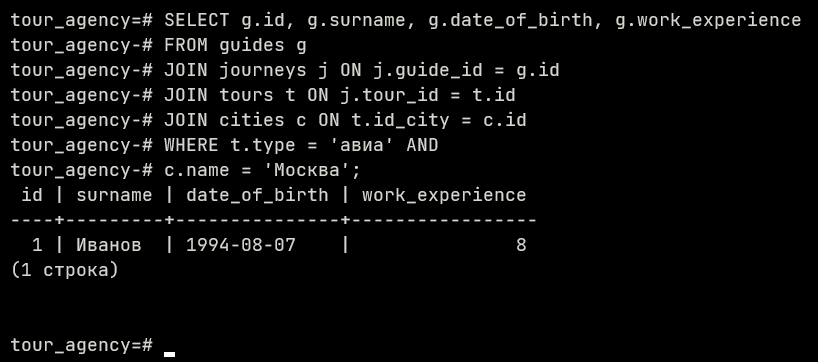


Рисунок 8. Информация обо всех руководителей туров, выполняющие авиатуры в Москву

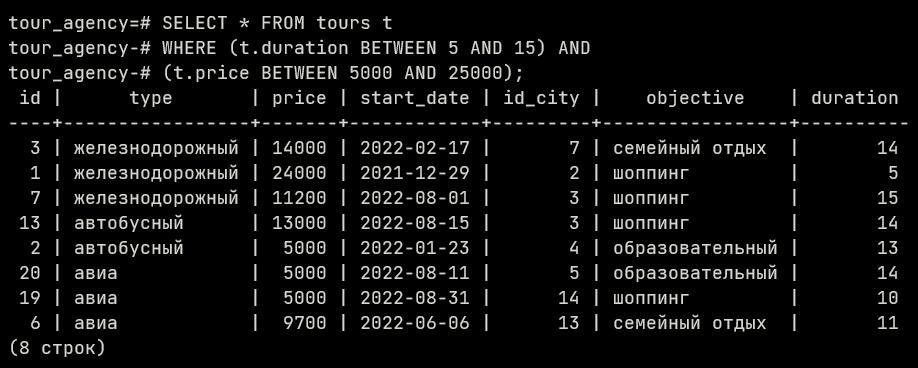


Рисунок 9. Туры длительностью от 5 до 15 дней и стоимостью от 5000 руб. до 25000 руб.

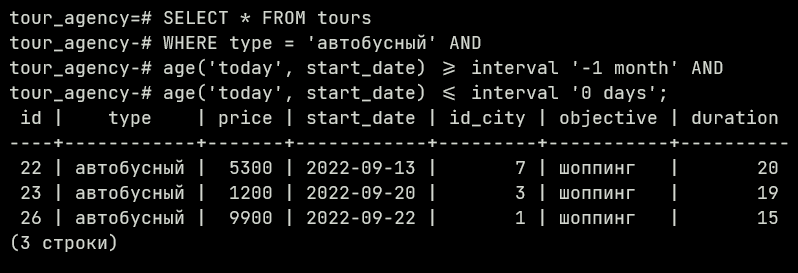


Рисунок 10. Все автобусные туры, которые начинаются в ближайший месяц

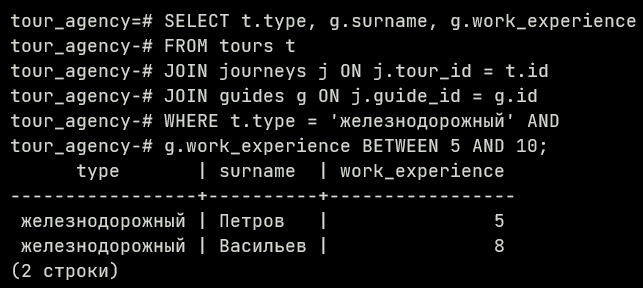


Рисунок 11. Все железнодорожные туры, осуществленные руководителями туров со стажем работы от 5 до 10 лет

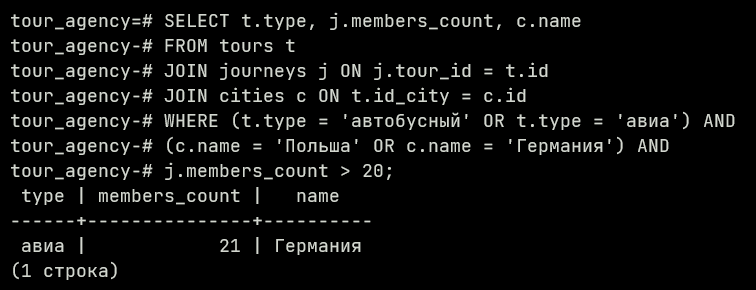


Рисунок 12. Все туры типа «авиа» и «автобусные» в страны Польша и Германия для групп с количеством более 20 человек.

Осталось сохранить базу данных в файл. Для этого воспользуемся встроенной командой pg\_dump.

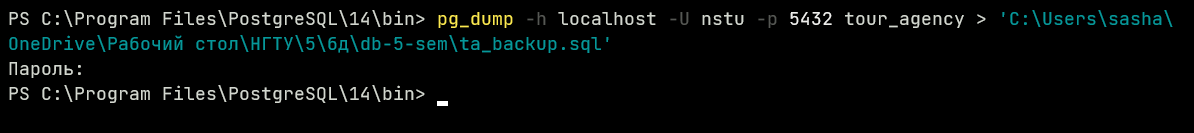
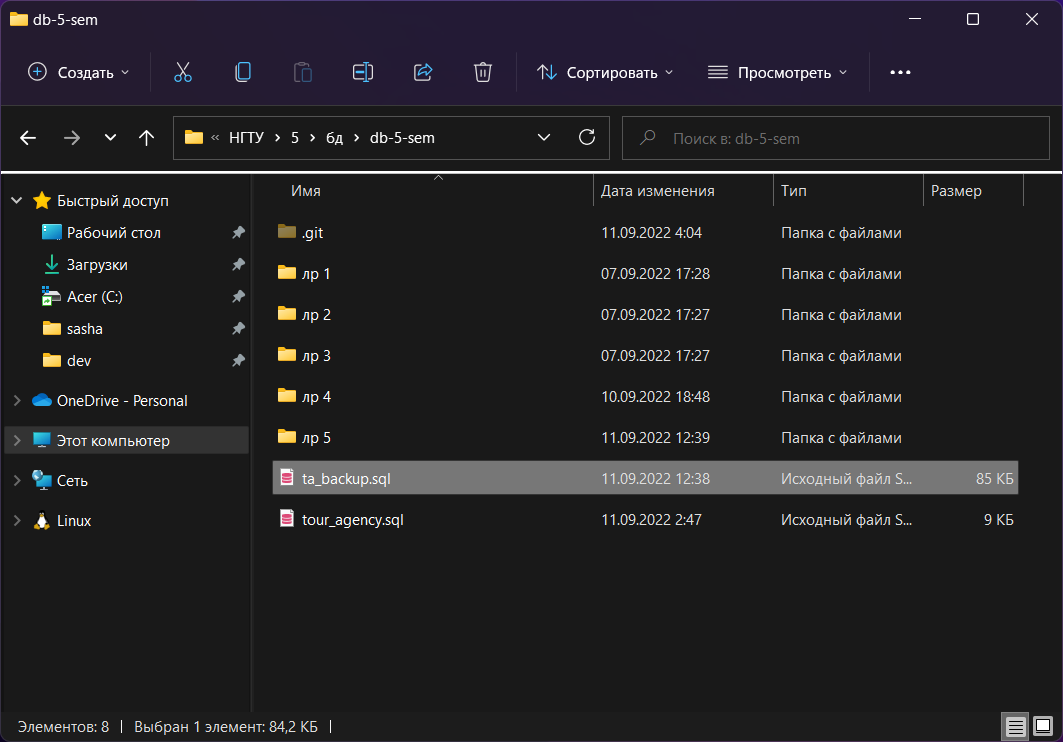


Рисунок 13. Сохранение бд в файл

В результате получим:



Вывод

В результате выполнения лабораторной работы были изучены индексы в PostgreSQL и их влияния на скорость поиска информации в базе данных. Также были изучены команды EXPLAIN и получены навыки по работе с командой SELECT в среде PostgreSQL.

Приложение

CREATE TABLE tours2 (

id integer,

type varchar,

price numeric,

duration\_days integer,

id\_city integer,

PRIMARY KEY(id),

FOREIGN KEY(id\_city) REFERENCES cities(id)

);

create function add\_n(int) returns char

as 'declare t int; begin

select max(id) into t from tours2;

for k in (t+1)..($1+t+1) loop

insert into tours2 values

(k, ''авиа''||round(random()\*1000),

round(10000+random()\*100000),

round(3+random()\*18),

round(1+random()\*13));

end loop; return''Done!'';

end;'

language 'plpgsql';

-- вставить значение перед вызовом функции add\_n(int)

EXPLAIN ANALYZE select \* from tours2 where price = 50000;

CREATE INDEX price\_index ON tours2 USING BTREE(price);

CREATE INDEX price\_index ON tours2 USING HASH(price);

EXPLAIN ANALYZE select \* from tours2 where upper(type) = 'авиа500';

CREATE INDEX type\_index ON tours2 (upper(type));

-- 1

SELECT g.id, g.surname, g.date\_of\_birth, g.work\_experience

FROM guides g

JOIN journeys j ON j.guide\_id = g.id

JOIN tours t ON j.tour\_id = t.id

JOIN cities c ON t.id\_city = c.id

WHERE t.type = 'авиа' AND

c.name = 'Москва';

-- 2

tour\_agency=# SELECT \* FROM tours t

tour\_agency-# WHERE (t.duration BETWEEN 5 AND 15) AND

tour\_agency-# (t.price BETWEEN 5000 AND 25000);

-- 3

SELECT \* FROM tours

WHERE type = 'автобусный' AND

age('today', start\_date) >= interval '-1 month' AND

age('today', start\_date) <= interval '0 days';

-- 4

SELECT t.type, g.surname, g.work\_experience

FROM tours t

JOIN journeys j ON j.tour\_id = t.id

JOIN guides g ON j.guide\_id = g.id

WHERE t.type = 'железнодорожный' AND

g.work\_experience BETWEEN 5 AND 10;

-- 5

SELECT t.type, j.members\_count, c.name

FROM tours t

JOIN journeys j ON j.tour\_id = t.id

JOIN cities c ON t.id\_city = c.id

WHERE (t.type = 'автобусный' OR t.type = 'авиа') AND

(c.name = 'Польша' OR c.name = 'Германия') AND

j.members\_count > 20;