|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа, обработки и интерпретации больших данных**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 2 |

**Название:** Оптимизация запросов. Основы EXPLAIN в PostgreSQL. Индексация

**Дисциплина:** Технология параллельных систем баз данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ6-12М |  |  | С.В. Астахов | |
|  | (Группа) |  | (Подпись, дата) | | (И.О. Фамилия) |
|  |  |  |  | |  |
| Преподаватель |  |  |  | |  |
|  |  |  | (Подпись, дата) | | (И.О. Фамилия) |

Москва, 2023

**Введение**

**1. Цель работы:** формирование следующей компетенции: студент должен получить навыки работы с командой EXPLAIN. Он также должен познакомиться с эффективными методами индексации в PostgreSQL.

**Ход выполнения**

**2. Основы EXPLAIN.**

Создадим таблицу, заполним ее миллионом записей, выполним команду EXPLAIN для запроса выборки всех записей, затем добавим записи в таблицу и выполним EXPLAIN повторно. Как видно из рисунка 1, при вставке новых записей в таблицу, ее статистика не обновляется автоматически, что приводит к повторяющимся результатам при вызове EXPLAIN.

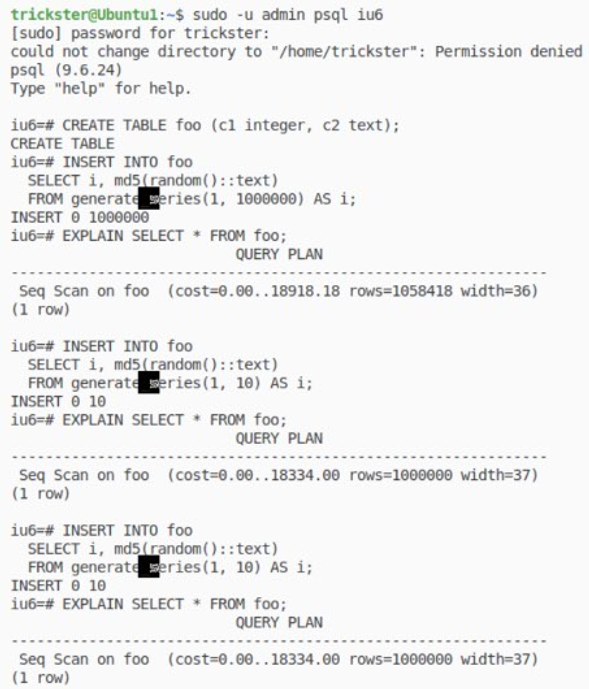


Рисунок 1 — пример EXPLAIN без обновления статистики

Обновим статистику и убедимся, что результаты вызова EXPLAIN изменились, так как была собрана актуальная статистика (рисунок 2).

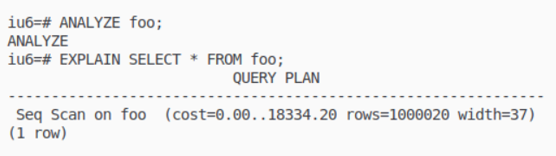


Рисунок 2 — обновление статистики по таблице

Используем EXPLAIN ANALYZE, чтобы увидеть время, потраченное на реальное исполнение запроса и количество считанных строк.

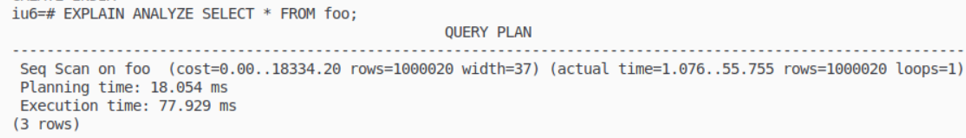


Рисунок 3 — пример выполнения EXPLAIN ANALYZE

Добавим дополнительное условие в запрос, увидим, что стоимость выполнения запроса изменилась, а количество выбранных строк уменьшилось (рисунок 4).

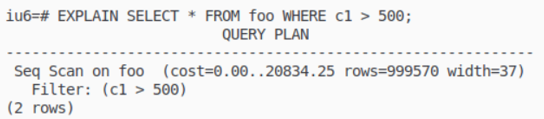


Рисунок 4 — исследование запроса с дополнительным условием

Создадим индекс для c1 и попробуем повторить запрос. Индекс использоваться не будет, так как выбирается большая доля записей из таблицы и использование индекса неэффективно (рисунки 5, 6).



Рисунок 5 — создание индекса

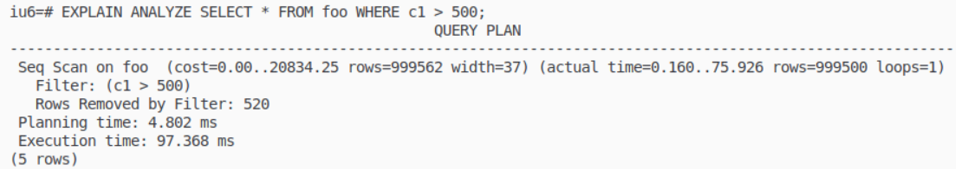


Рисунок 6 — запрос после создания индекса

Изменим запрос так, чтобы выбиралось малое число записей. Теперь индекс будет использоваться (рисунок 7).

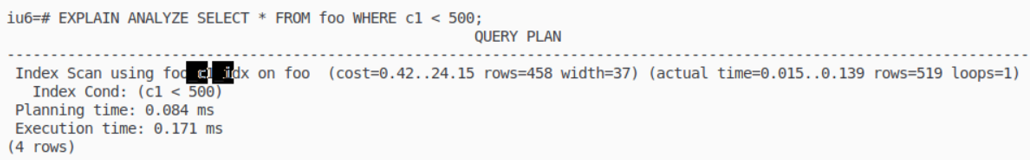


Рисунок 7 — запрос с использованием индекса

Усложним запрос, добавив условие по текстовому полю. Сначала будет проходить фильтрация, а затем — сканирование по индексу (рисунок 8).

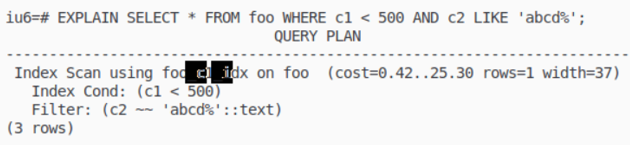


Рисунок 8 — запрос с фильтрацией по текстовому полю

Уберем условие по индексированному полю, теперь будет происходить полное сканирование таблицы, так как по полю c2 индекс не создан (рисунок 9).

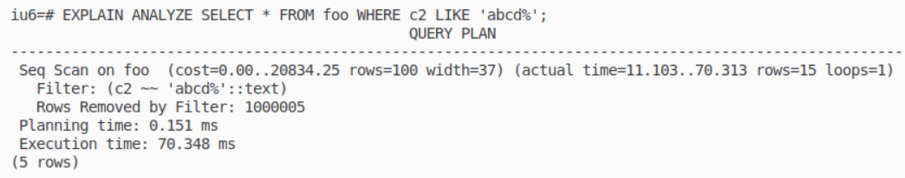


Рисунок 9 — запрос без условия по полю c1

Создадим индекс по полю c2, теперь он будет использоваться в запросе (рисунок 10).

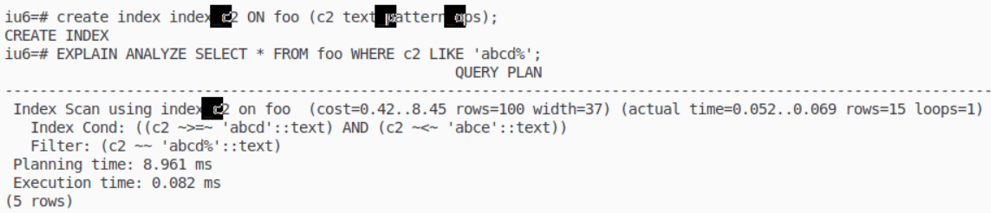


Рисунок 10 — запрос по индексированному текстовому полю

Удалим индекс по полю c1 и выполним запрос с упорядочиванием, будет использован метод сортировки external merge.

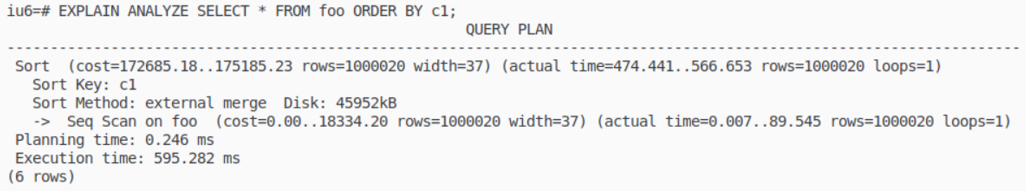


Рисунок 11 — запрос с упорядочиванием

Увеличим объем используемой оперативной памяти — теперь будет использована сортировка quicksort (рисунок 12).

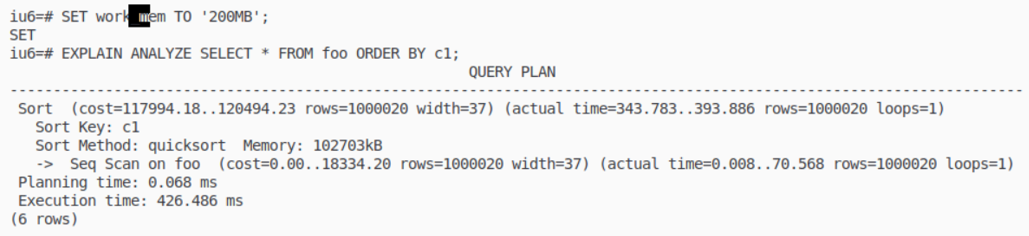


Рисунок 12 — запрос с упорядочиванием после изменения настроек

Создадим индекс для поля c1 и повторим запрос. Так как в индексе значения упорядочены, сортировка не понадобится (рисунок 13).

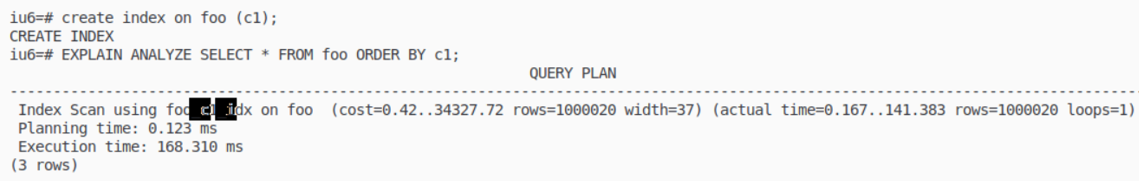


Рисунок 13 — запрос с упорядочиванием по индексу

Создадим новую таблицу и соберем для нее статистику (рисунок 14).

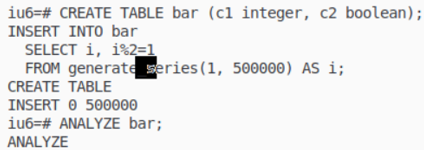


Рисунок 14 — создание новой таблицы

Выполним соединение новой таблицы со старой, затем создадим для нее индекс и выполним соединение повторно. Тип соединения изменится из-за наличия индексов (рисунок 15).

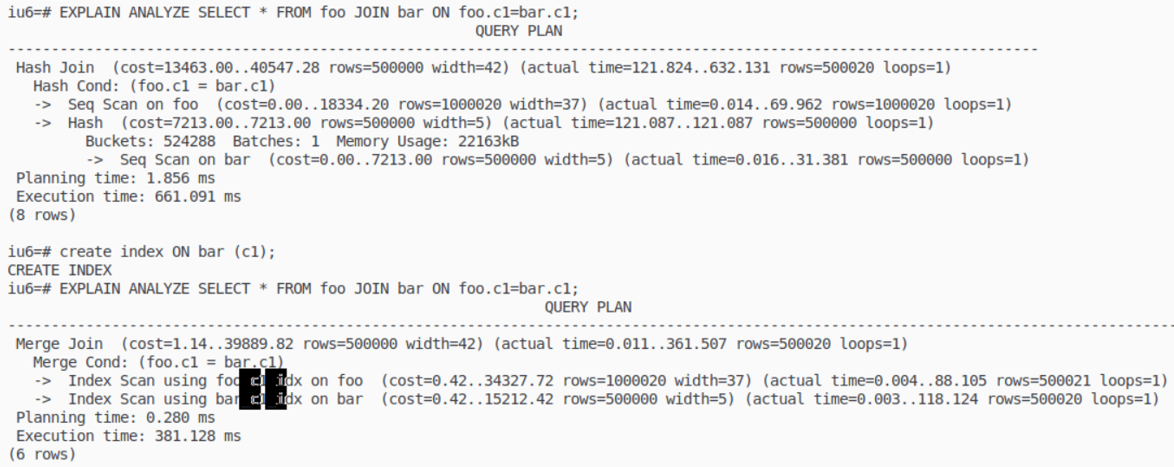


Рисунок 15 — соединение таблиц

Очистим кэш PostgreSQL и 3 раза обратимся к таблице foo. При каждом запросе ее часть будет кэшироваться для ускорения последующих обращений (рисунок 16).

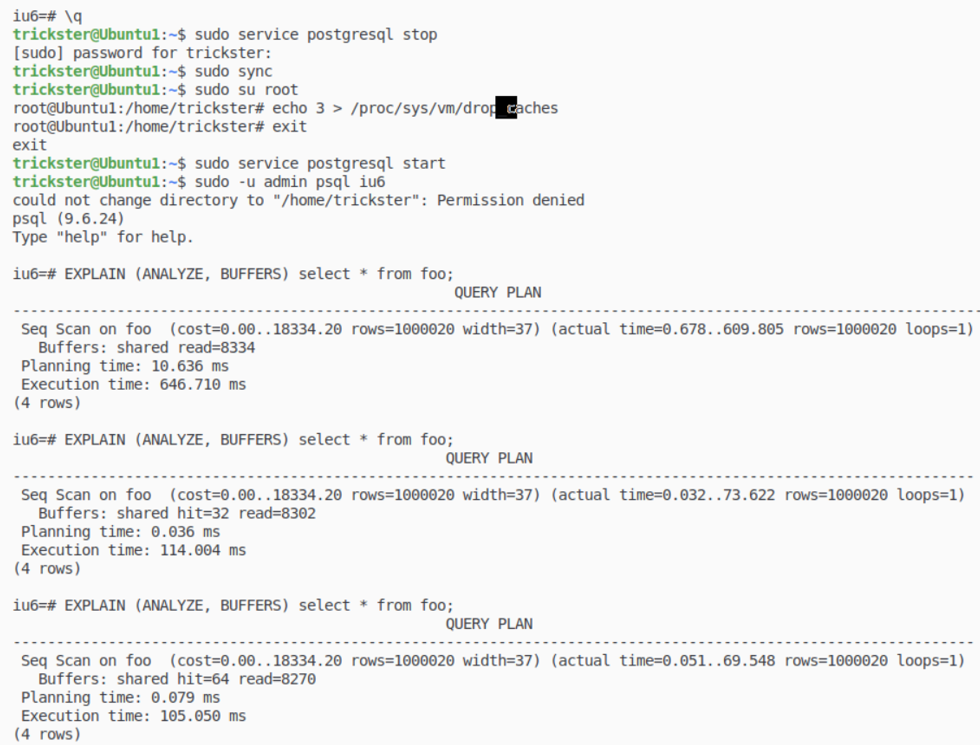


Рисунок 16 — работа кэша

**4. Эффективные методы индексации в PostgreSQL**

Переподключимся к БД от имени пользователя postgres, создадим новую таблицу и индексы в ней (рисунок 17).

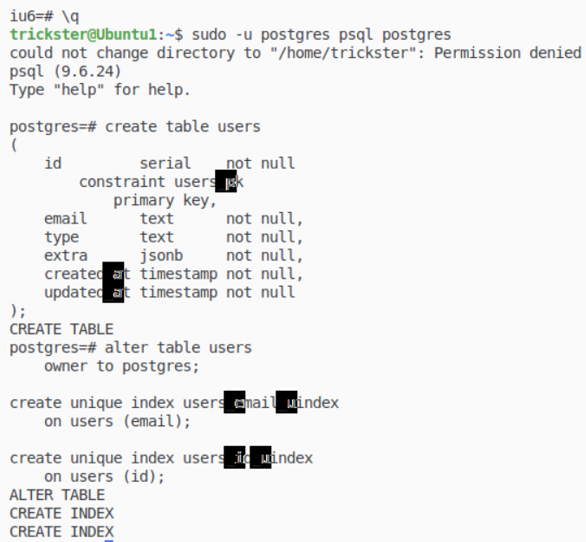


Рисунок 17 — создание новой БД

Заполним новую БД (рисунок 18).

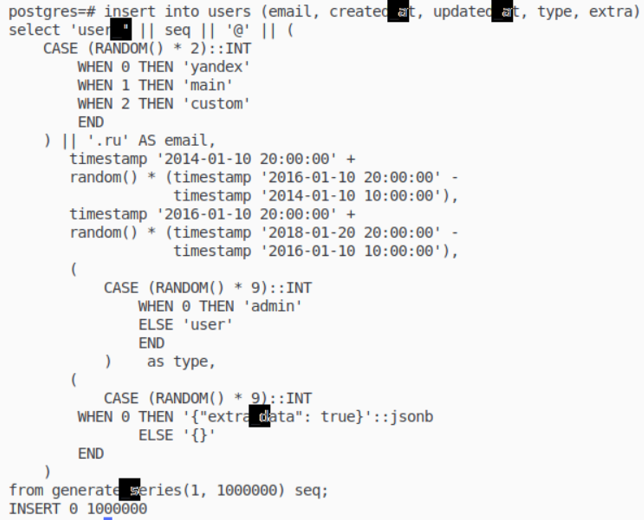


Рисунок 18 — заполнение БД

Сравним выполнение указанных в задании запросов без использования индексов и при их наличии. Очевидно, при использовании индексов по полям из условий (where) не проводилось полного сканирования, а использовались индексы, что значительно ускоряло выполнение запросов (рисунки 19-21).

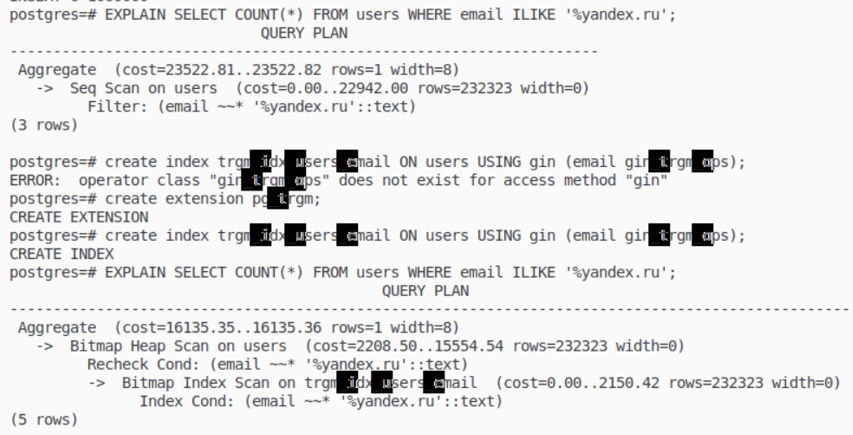


Рисунок 19 — использование индекса по полю email

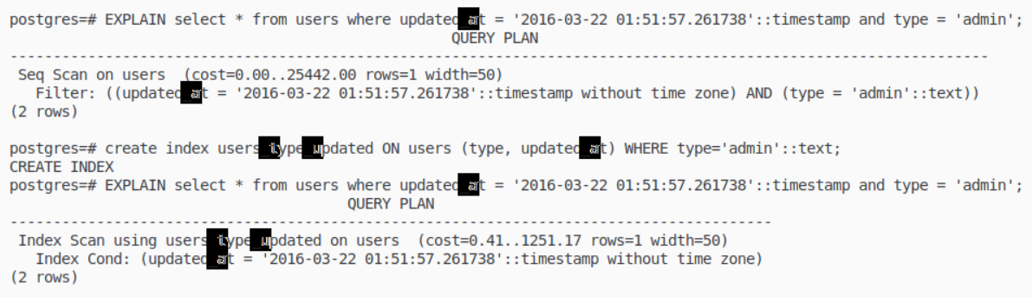


Рисунок 20 — использование частичного индекса

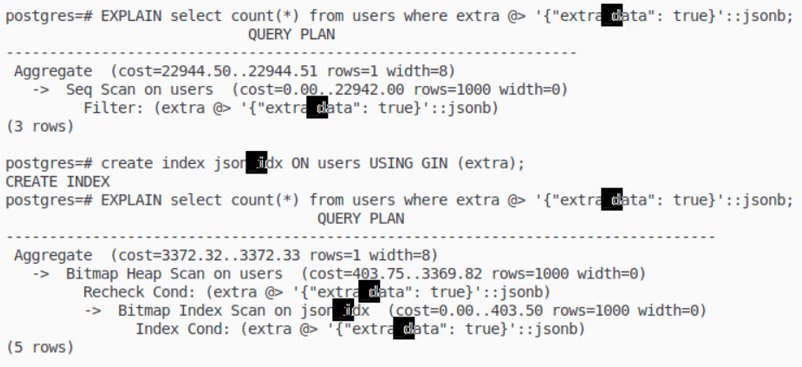


Рисунок 21 — использование индекса по json-полю

**Вывод:** в ходе лабораторной работы были получены навыки работы с командой EXPLAIN. Было произведено знакомство с эффективными методами индексации в PostgreSQL. Все задание были успешно выполнены, а их результаты соответствуют требованиям.