МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «Санкт-Петербургский национальный исследовательский университет ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине «Информационные системы и базы данных»

Вариант № 1675

Выполнил:

Студент группы P3116

Билошицкий Михаил Владимирович

Преподаватель:

Горбунов Михаил Витальевич

Санкт-Петербург, 2023

Содержание

Текст задания

Лабораторная работа #4

Задание.

По варианту, выданному преподавателем, составить и выполнить запросы к базе данных "Учебный процесс".

Команда для подключения к базе данных ucheb:

psql -h pg -d ucheb

Отчёт по лабораторной работе должен содержать:

1. Текст задания.
2. Реализацию запросов на SQL.
3. Планы выполнения запросов.
4. Ответы на вопросы, представленные в задании.
5. Выводы по работе.

Темы для подготовки к защите лабораторной работы:

1. Индексы
2. Оптимизация запросов
3. Выбор плана выполнения запросов

Введите вариант: 1675

Внимание! У разных вариантов разный текст задания!

Составить запросы на языке SQL (пункты 1-2).

Для каждого запроса предложить индексы, добавление которых уменьшит время выполнения запроса (указать таблицы/атрибуты, для которых нужно добавить индексы, написать тип индекса; объяснить, почему добавление индекса будет полезным для данного запроса).

Для запросов 1-2 необходимо составить возможные планы выполнения запросов. Планы составляются на основании предположения, что в таблицах отсутствуют индексы. Из составленных планов необходимо выбрать оптимальный и объяснить свой выбор.
Изменятся ли планы при добавлении индекса и как?

Для запросов 1-2 необходимо добавить в отчет вывод команды EXPLAIN ANALYZE [запрос]

Подробные ответы на все вышеперечисленные вопросы должны присутствовать в отчете (планы выполнения запросов должны быть нарисованы, ответы на вопросы - представлены в текстовом виде).

1. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:
   Н_ТИПЫ_ВЕДОМОСТЕЙ, Н_ВЕДОМОСТИ.
   Вывести атрибуты: Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н_ВЕДОМОСТИ.ИД.
   Фильтры (AND):
   a) Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД = 1.
   b) Н_ВЕДОМОСТИ.ЧЛВК_ИД < 153285.
   c) Н_ВЕДОМОСТИ.ЧЛВК_ИД > 142390.
   Вид соединения: INNER JOIN.
2. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:
   Таблицы: Н_ЛЮДИ, Н_ВЕДОМОСТИ, Н_СЕССИЯ.
   Вывести атрибуты: Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ДАТА, Н_СЕССИЯ.ЧЛВК_ИД.
   Фильтры (AND):
   a) Н_ЛЮДИ.ИД = 100865.
   b) Н_ВЕДОМОСТИ.ДАТА > 2022-06-08.
   Вид соединения: INNER JOIN.

Реализация запросов на SQL

Запрос 1.
Индексы:

- Таблица Н_ТИПЫ_ВЕДОМОСТЕЙ атрибут ИД, индекс **не будет эффективен**, так как таблица имеет небольшое количество строк и добавление индекса только замедлит выполнение запроса.
- Таблица Н_ВЕДОМОСТИ атрибут ТВ_ИД, индекс **не будет эффективен**, так как атрибут ТВ_ИД является внешним ключом таблицы Н_ТИПЫ_ВЕДОМОСТЕЙ атрибута ИД, которое содержит малое количество строк, следовательно, атрибут ТВ_ИД содержит малое количество уникальных значений и использование индекса не даст значительного ускорения запроса.
- Таблица Н_ВЕДОМОСТИ атрибут ЧЛВК_ИД, индекс b-tree. **Индекс полезен**, так как в таблице содержится большое количество значений и атрибут ЧЛВК_ИД содержит большое количество уникальных значений, по которым будет производится дальнейшая фильтрация и индекс ускорит выполнение запроса. Используется индекс типа b-tree, так как данный тип индекса лучше осуществляет сортировку данных, чем индекс типа hash, что в нашем запросе полезно, так как происходит фильтрация числового атрибута с операторами больше-меньше.

```
CREATE INDEX idx_типы_ведомостей ON Н_ТИПЫ_ВЕДОМОСТЕЙ (ИД);
CREATE INDEX idx_ведомости_тв_ид ON Н_ВЕДОМОСТИ (ТВ_ИД);
CREATE INDEX idx_ведомости_члвк_ид ON Н_ВЕДОМОСТИ (ЧЛВК_ИД);

SELECT Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н_ВЕДОМОСТИ.ИД
FROM Н_ВЕДОМОСТИ
JOIN Н_ТИПЫ_ВЕДОМОСТЕЙ ON Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД = Н_ВЕДОМОСТИ.ТВ_ИД
WHERE
    Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД = 1 AND
    Н_ВЕДОМОСТИ.ЧЛВК_ИД < 153285 AND
    Н_ВЕДОМОСТИ.ЧЛВК_ИД > 142390;
```

Вывод EXPLAIN ANALYZE:

```
Nested Loop  (cost=496.17..5502.69 rows=30924 width=422) (actual time=1.916..15.219
rows=28671 loops=1)
   ->  Seq Scan on "Н_ТИПЫ_ВЕДОМОСТЕЙ"  (cost=0.00..1.04 rows=1 width=422) (actual
time=0.019..0.022 rows=1 loops=1)
         Filter: ("ИД" = 1)
         Rows Removed by Filter: 2
   ->  Bitmap Heap Scan on "Н_ВЕДОМОСТИ"  (cost=496.17..5192.41 rows=30924 width=8)
(actual time=1.893..11.767 rows=28671 loops=1)
         Recheck Cond: (("ЧЛВК_ИД" < 153285) AND ("ЧЛВК_ИД" > 142390))
         Filter: ("ТВ_ИД" = 1)
```

```
        Rows Removed by Filter: 7602
        Heap Blocks: exact=1645
        ->  Bitmap Index Scan on "ВЕД_ЧЛВК_FK_IFK"  (cost=0.00..488.44 rows=36014
width=0) (actual time=1.656..1.657 rows=36273 loops=1)
              Index Cond: (("ЧЛВК_ИД" < 153285) AND ("ЧЛВК_ИД" > 142390))
Planning Time: 0.835 ms
Execution Time: 16.650 ms
```

Запрос 2.

Индексы:

- Таблица Н_ЛЮДИ атрибут ИД, индекс hash. **Индекс полезен**, так как используется соединение таблиц Н_ЛЮДИ и Н_ВЕДОМОСТИ по данному атрибуту, а также фильтрация через оператор =, соответственно будет осуществляться поиск уникального значения атрибута по большой таблице Н_ЛЮДИ и при присоединении с Н_ВЕДОМОСТИ, с чем быстро может справиться hash индекс, который работает эффективно при поиске данных.
- Таблица Н_ВЕДОМОСТИ атрибут ЧЛВК_ИД, индекс hash. **Индекс полезен**, так как используется соединение таблиц Н_ЛЮДИ и Н_ВЕДОМОСТИ по данному атрибуту.
- Таблица Н_ВЕДОМОСТИ атрибут ДАТА, индекс b-tree. **Индекс полезен** так как в таблице содержится большое количество значений и атрибут ДАТА содержит большое количество уникальных значений, по которым будет производится дальнейшая фильтрация и индекс ускорит выполнение запроса. Используется индекс типа b-tree, так как данный тип индекса лучше осуществляет сортировку данных, чем индекс типа hash, что в нашем запросе полезно, так как происходит фильтрация с операторами больше-меньше.

```
CREATE INDEX idx_люди_ид ON Н_ЛЮДИ USING HASH (ИД);
CREATE INDEX idx_ведомости_члвк_ид ON Н_ВЕДОМОСТИ USING HASH (ЧЛВК_ИД);
CREATE INDEX idx_ведомости_дата ON Н_ВЕДОМОСТИ (ДАТА);

SELECT Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ДАТА, Н_СЕССИЯ.ЧЛВК_ИД
FROM Н_ЛЮДИ
JOIN Н_ВЕДОМОСТИ ON Н_ЛЮДИ.ИД = Н_ВЕДОМОСТИ.ЧЛВК_ИД
JOIN Н_СЕССИЯ ON Н_ЛЮДИ.ИД = Н_СЕССИЯ.ЧЛВК_ИД
WHERE
    Н_ЛЮДИ.ИД = 100865 AND
    Н_ВЕДОМОСТИ.ДАТА > '2022-06-08';
```

Вывод EXPLAIN ANALYZE:

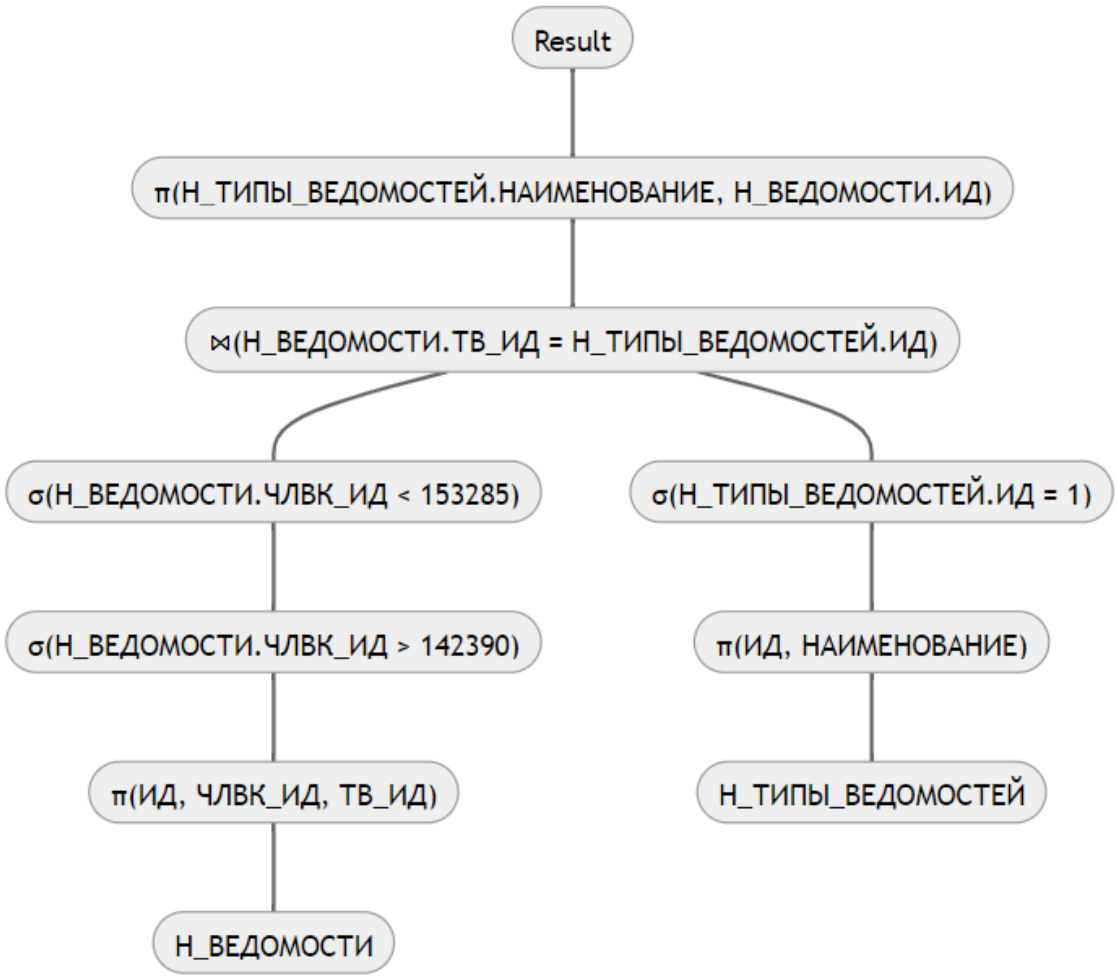
```
Nested Loop  (cost=0.86..20.21 rows=15 width=28) (actual time=0.036..0.037 rows=0
loops=1)
->  Nested Loop  (cost=0.58..15.51 rows=1 width=28) (actual time=0.035..0.036 rows=0
loops=1)
```

```
        ->  Index Scan using "ЧЛВК_PK" on "Н_ЛЮДИ"  (cost=0.28..8.30 rows=1 width=20)
(actual time=0.032..0.032 rows=1 loops=1)
            Index Cond: ("ИД" = 100865)
        ->  Index Scan using "ВЕД_ДАТА_I" on "Н_ВЕДОМОСТИ"  (cost=0.29..7.20 rows=1
width=12) (actual time=0.002..0.002 rows=0 loops=1)
            Index Cond: ("ДАТА" > '2022-06-08 00:00:00'::timestamp without time zone)
            Filter: ("ЧЛВК_ИД" = 100865)
->  Index Only Scan using "SYS_C003500_IFK" on "Н_СЕССИЯ"  (cost=0.28..4.54 rows=15
width=4) (never executed)
        Index Cond: ("ЧЛВК_ИД" = 100865)
        Heap Fetches: 0
Planning Time: 0.704 ms
```

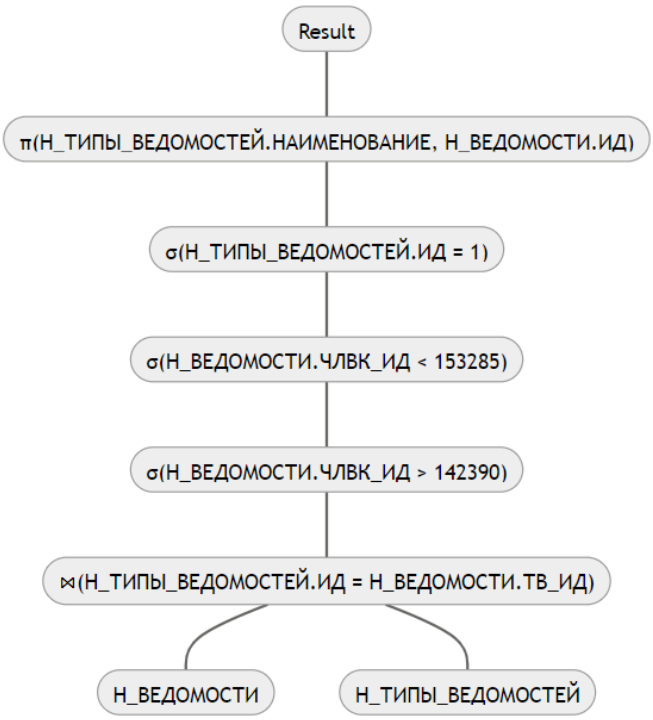
Планы выполнения запросов

Запрос 1.

План 1.



План 2.

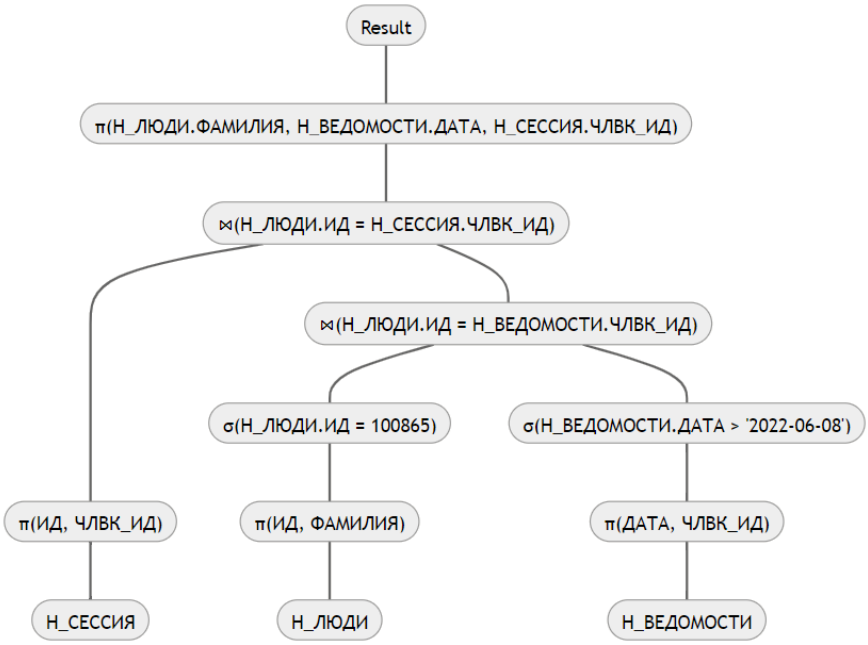


Оптимальным планом является план 1, так как в нем выборка и проекция производится как можно раньше, следовательно, в соединении таблиц участвует меньшее количество строк, чем в менее оптимальном плане 2.

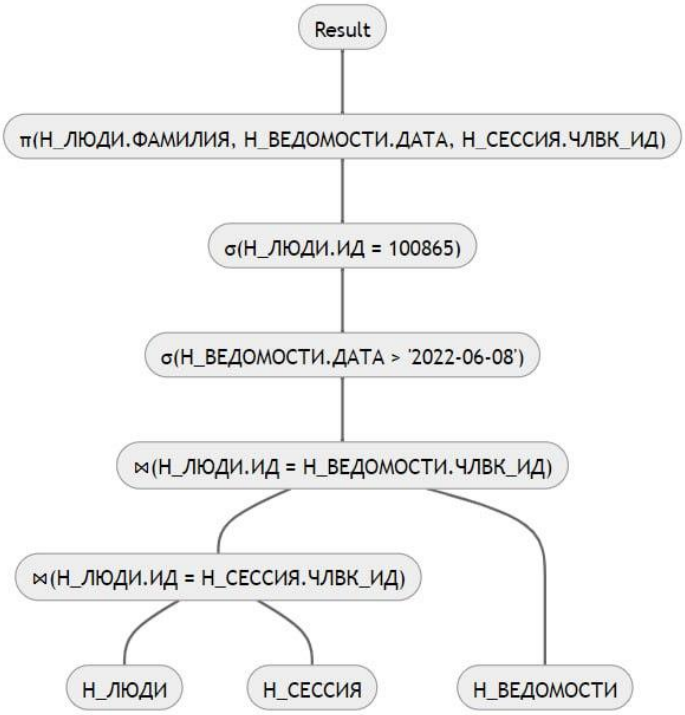
При добавлении индексов план выполнения запроса не изменится, так как индексы только ускорят фильтрацию и соединение таблиц. В результате время выполнения запроса сократится. Но, стоит отметить, что СУБД выполняет проекцию перед выполнением фильтрации.

Запрос 2.

План 1.



План 2.



Оптимальным планом является план 1, так как в нем выборка и проекция производится как можно раньше, следовательно, в соединении таблиц участвует меньшее количество строк, чем в менее оптимальном плане 2.

При добавлении индексов план выполнения запроса не изменится, так как индексы только ускорят фильтрацию и соединение таблиц. В результате время выполнения запроса сократится. Но, стоит отметить, что СУБД выполняет проекцию перед выполнением фильтрации.

Вывод

В рамках данной лабораторной работы были изучены индексы в PostgreSQL. Получены и отработаны на практике знания по построению планов запросов. Проведен анализ команды EXPLAIN ANALYZE.