*Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»
Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №4 по дисциплине «Базы данных» Вариант 8555

Выполнил:
Студент группы P3112
Медведев Ярослав Александрович
Преподаватель:
Максимов Андрей Николаевич

г. Санкт-Петербург
2024

Ход работы

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:
   Таблицы: Н_ОЦЕНКИ, Н_ВЕДОМОСТИ.
   Вывести атрибуты: Н_ОЦЕНКИ.ПРИМЕЧАНИЕ, Н_ВЕДОМОСТИ.ДАТА.
   Фильтры (AND):
   a) Н_ОЦЕНКИ.ПРИМЕЧАНИЕ < хорошо.
   b) Н_ВЕДОМОСТИ.ДАТА < 1998-01-05.
   c) Н_ВЕДОМОСТИ.ДАТА = 2022-06-08.
   Вид соединения: LEFT 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.ЧЛВК_ИД, Н_СЕССИЯ.ДАТА.
   Фильтры (AND):
   a) Н_ЛЮДИ.ФАМИЛИЯ < Петров.
   b) Н_ВЕДОМОСТИ.ДАТА > 1998-01-05.
   Вид соединения: RIGHT JOIN.

SQL-запросы

1 запрос:

```
SELECT Н_ОЦЕНКИ.ПРИМЕЧАНИЕ, Н_ВЕДОМОСТИ.ДАТА
FROM Н_ОЦЕНКИ
LEFT JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ОЦЕНКА = Н_ОЦЕНКИ.КОД
WHERE Н_ОЦЕНКИ.ПРИМЕЧАНИЕ < 'ХОРОШО' AND Н_ВЕДОМОСТИ.ДАТА <
'1998-01-05' AND Н_ВЕДОМОСТИ.ДАТА = '2022-06-08';
```

2 запрос:

```
SELECT Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.ДАТА
FROM Н_ЛЮДИ
RIGHT JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
RIGHT JOIN Н_СЕССИЯ ON Н_СЕССИЯ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
WHERE Н_ЛЮДИ.ФАМИЛИЯ < 'Петров' AND Н_ВЕДОМОСТИ.ДАТА >
'1998-01-05';
```

Возможные индексы

1 запрос:

1. Добавление индексов на поля ОЦЕНКА и КОД таблиц Н_*ВЕДОМОСТИ и Н*_ОЦЕНКИ соответственно ускорит операцию LEFT JOIN (Hash индексы, тк при операции “=” они работают за O(1)).
   ```
   CREATE INDEX OCENKA_INDEX ON Н_ВЕДОМОСТИ USING hash(ОЦЕНКА);
   ```
2. Добавление индексов на поля ПРИМЕЧАНИЯ и ДАТА таблиц Н_*ОЦЕНКИ и Н*_ВЕДОМОСТИ соответственно для ускорения выборки (B-tree индексы, тк они справляются с операциями “<”, “>” за O(log n)).

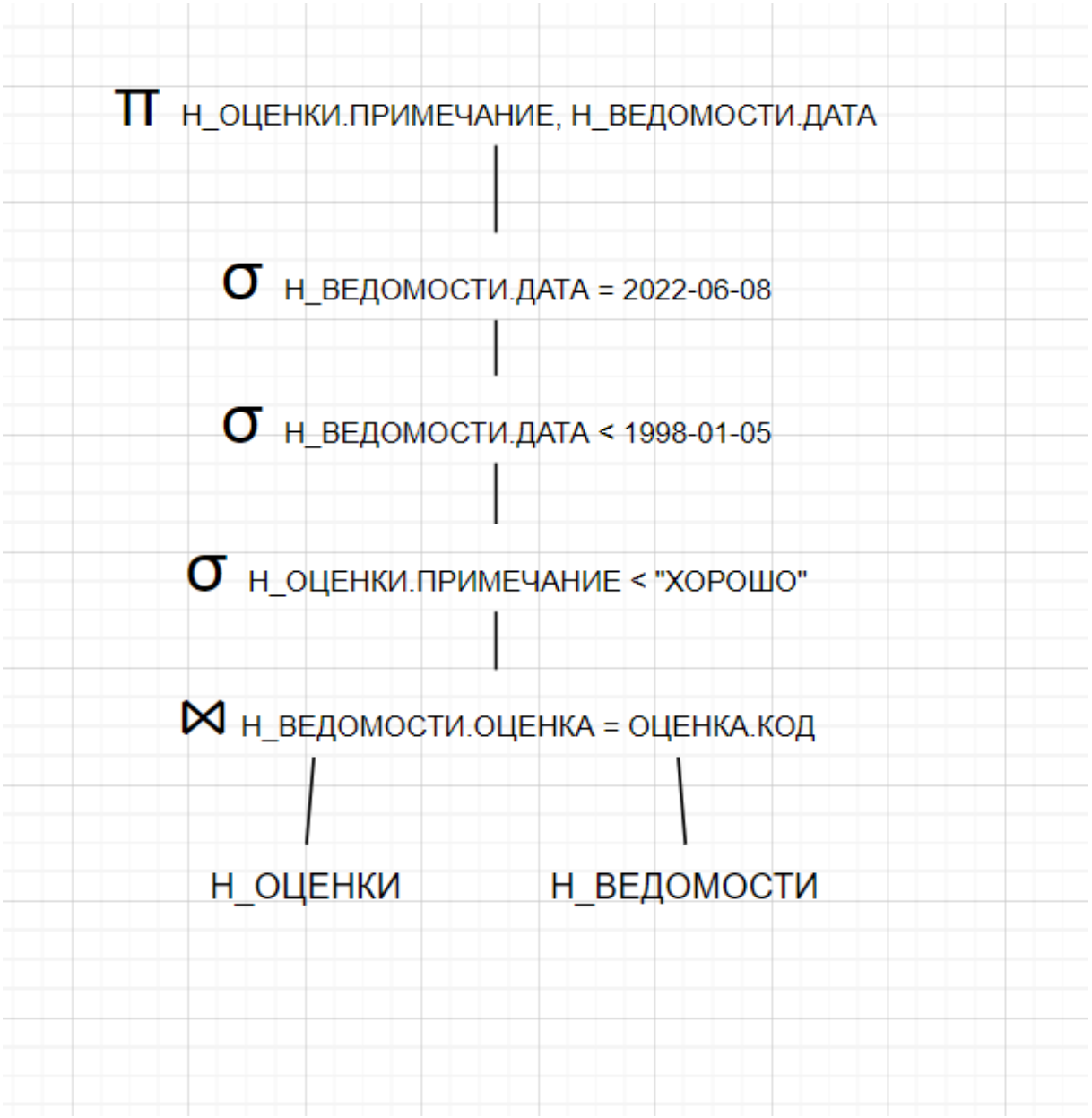
```
CREATE INDEX PRIMECH_INDEX ON H_ОЦЕНКИ USING
btree(ПРИМЕЧАНИЯ);
CREATE INDEX DATE_INDEX ON H_ВЕДОМОСТИ USING btree(ДАТА);
```

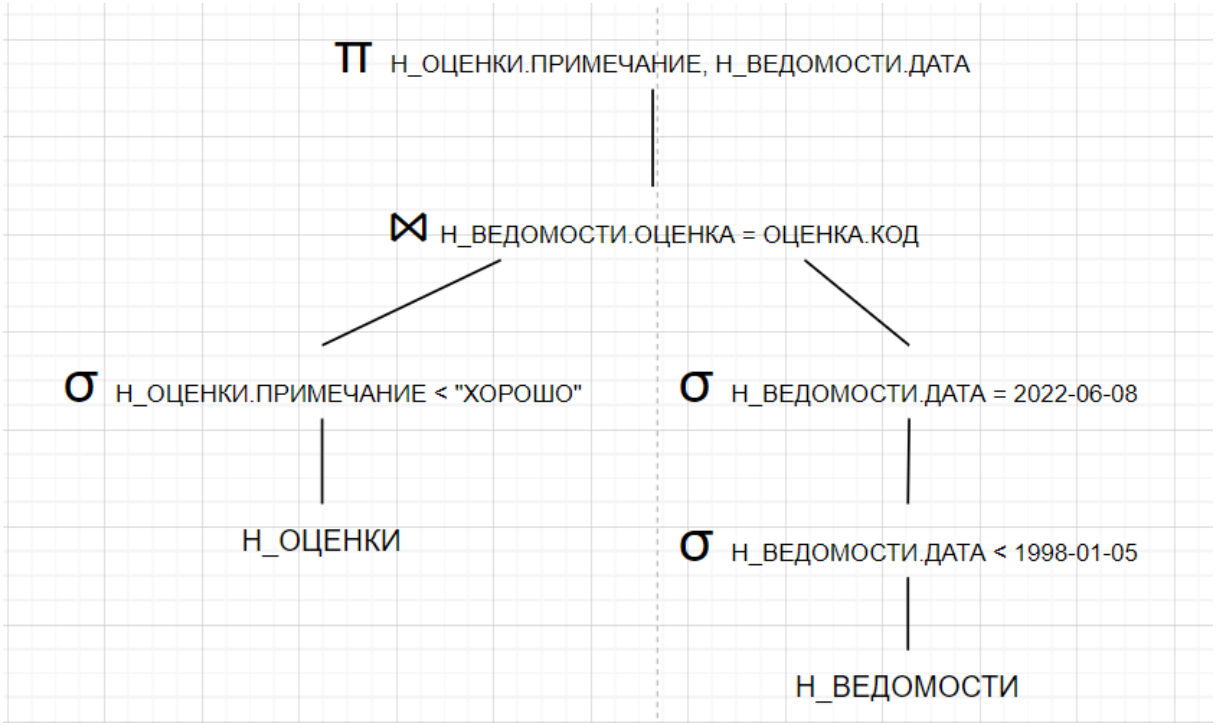
2 запрос:

1. Добавление индексов на поля ЧЛВК*_ИД, ИД и ЧЛВК_ИД таблиц Н_ВЕДОМОСТИ, Н_ЛЮДИ и Н_СЕССИЯ* соответственно ускорит операцию RIGHT JOIN (Hash индексы, тк при операции “=” они работают за O(1)).
```
CREATE INDEX PEOPLE_ID_INDEX ON H_ВЕДОМОСТИ USING
hash(ЧЛВК_ИД);
CREATE INDEX CHLVK_ID_INDEX ON HСЕССИЯ USING hash(ЧЛВК_ИД);
```
2. Добавление индексов на поля ФАМИЛИЯ и ДАТА таблиц Н*_ЛЮДИ и Н*_ВЕДОМОСТИ соответственно для ускорения выборки (B-tree индексы, тк они справляются с операциями “<”, “>” за O(log n)).
```
CREATE INDEX SURNAME_INDEX ON H_ЛЮДИ USING btree(ФАМИЛИЯ);
CREATE INDEX DATE_INDEX ON H_ВЕДОМОСТИ USING btree(ДАТА);
```

Планы запросов

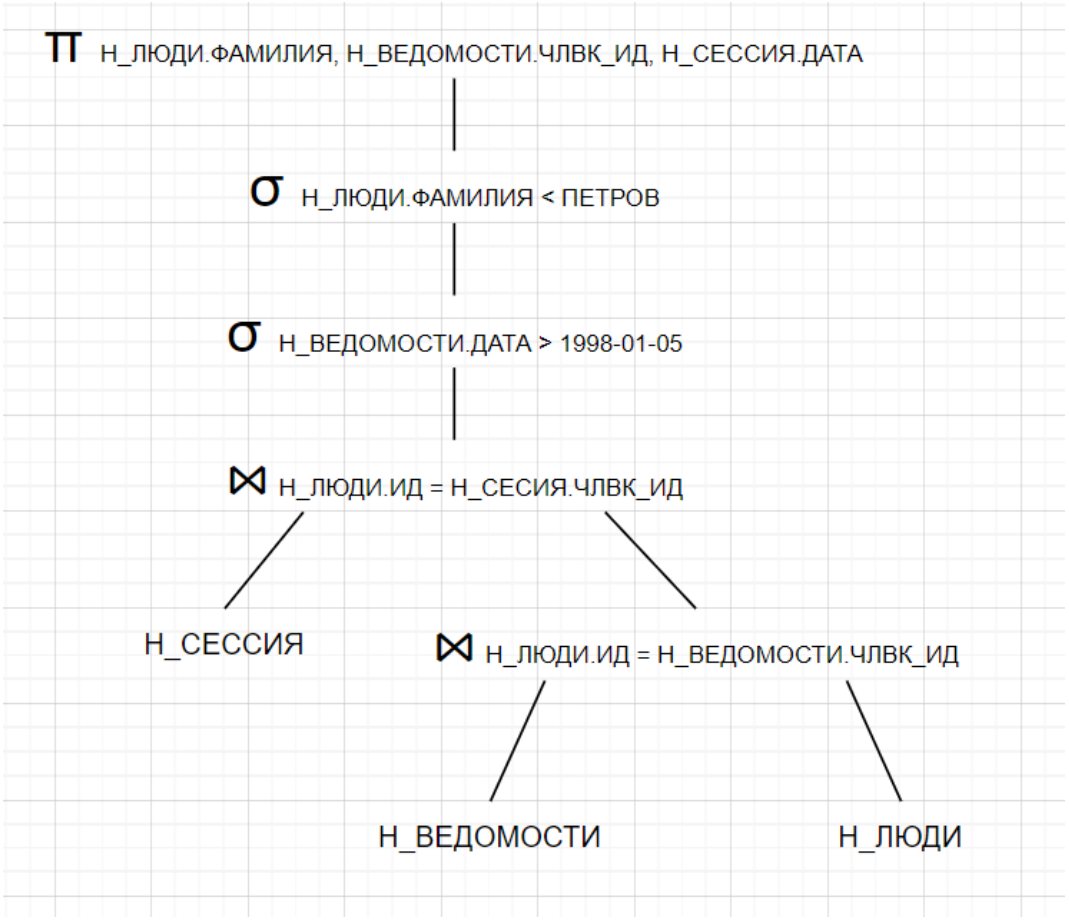
1 запрос:

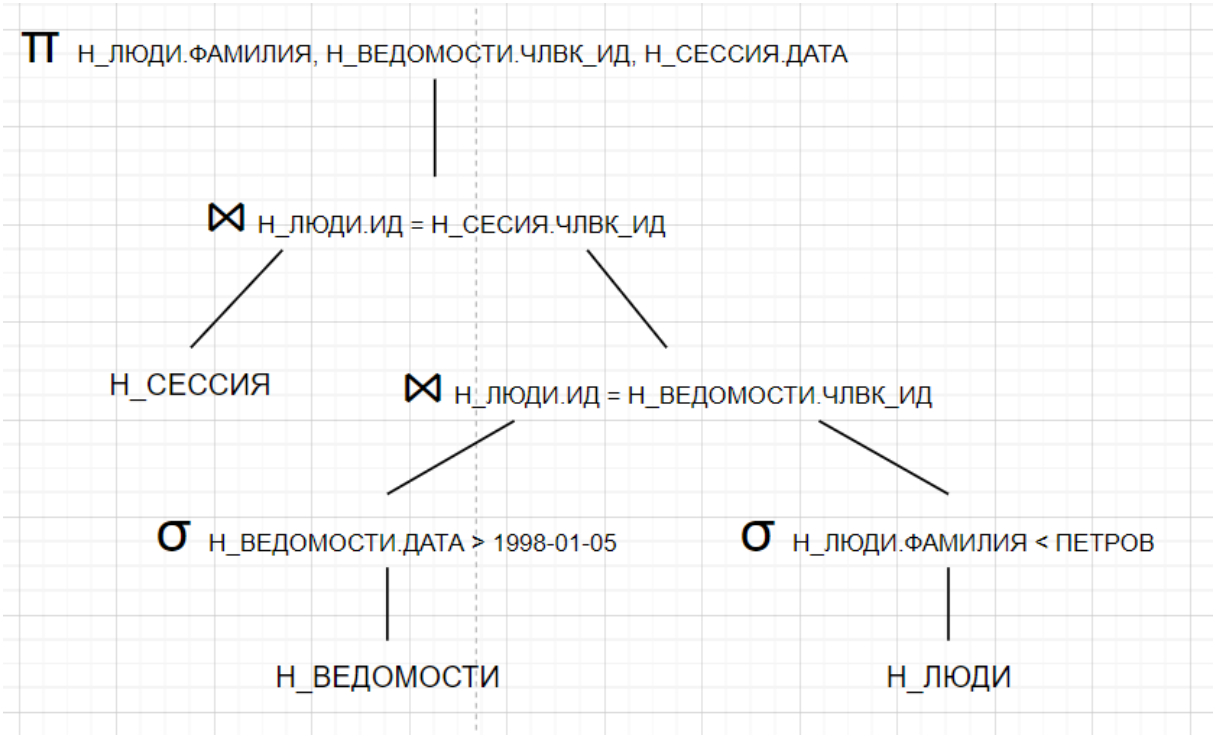




Второй план является наиболее оптимальным, тк производится соединение только нужной нам выборки. В данном случае операция соединения является более затратной, чем операция выборки. При добавлении индексов ускорятся операции выборки и соединения.

2 запрос:





Второй план является наиболее оптимальным, тк производится соединение только нужной нам выборки. В данном случае операция соединения является более затратной, чем операция выборки. При добавлении индексов ускорятся операции выборки и соединения.

Анализ запросов

1 запрос:

```
QUERY PLAN
Nested Loop  (cost=0.29..8.43 rows=1 width=30)
  Join Filter: (("Н_ОЦЕНКИ"."КОД")::text = ("Н_ВЕДОМОСТИ"."ОЦЕНКА")::text)
  ->  Index Scan using "ВЕД_ДАТА_I" on "Н_ВЕДОМОСТИ"  (cost=0.29..7.20 rows=1 width=14)
        Index Cond: (("ДАТА" < '1998-01-05 00:00:00'::timestamp without time zone) AND ("ДАТА" = '2022-06-08 00:00:00'::timestamp without time zone))
  ->  Seq Scan on "Н_ОЦЕНКИ"  (cost=0.00..1.11 rows=9 width=27)
        Filter: (("ПРИМЕЧАНИЕ")::text < 'ХОРОШО'::text)
```

2 запрос:

```
QUERY PLAN
Nested Loop  (cost=0.60..3555.64 rows=131379 width=28)
  Join Filter: ("Н_ЛЮДИ"."ИД" = "Н_ВЕДОМОСТИ"."ЧЛВК_ИД")
  ->  Nested Loop  (cost=0.29..357.52 rows=2080 width=32)
        ->  Seq Scan on "Н_СЕССИЯ"  (cost=0.00..108.52 rows=3752 width=12)
        ->  Memoize  (cost=0.29..0.87 rows=1 width=20)
              Cache Key: "Н_СЕССИЯ"."ЧЛВК_ИД"
              Cache Mode: logical
              ->  Index Scan using "ЧЛВК_PK" on "Н_ЛЮДИ"  (cost=0.28..0.86 rows=1 width=20)
                    Index Cond: ("ИД" = "Н_СЕССИЯ"."ЧЛВК_ИД")
                    Filter: (("ФАМИЛИЯ")::text < 'Петров'::text)
  ->  Memoize  (cost=0.30..6.16 rows=66 width=4)
        Cache Key: "Н_СЕССИЯ"."ЧЛВК_ИД"
        Cache Mode: logical
        ->  Index Scan using "ВЕД_ЧЛВК_FK_IFK" on "Н_ВЕДОМОСТИ"  (cost=0.29..6.15 rows=66 width=4)
              Index Cond: ("ЧЛВК_ИД" = "Н_СЕССИЯ"."ЧЛВК_ИД")
              Filter: ("ДАТА" > '1998-01-05 00:00:00'::timestamp without time zone)
```

Вывод

При выполнении данной лабораторной работы я узнал об оптимизации запросов в PostgreSQL и индексах. Рассмотрел как строить планы запросов, а также осознал важность оптимизировать запросы в PostgreSQL.