Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4

Вариант №1809

Выполнил

Макогон Ярослав Вадимович

Номер группы: P3118

Проверила

Бострикова Д. К.

г. Санкт-Петербург 2025г.

Содержание

Задание 3
Решение 4
Выводы 8

Задание

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

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.ИД, Н_ВЕДОМОСТИ.ИД.
   Фильтры (AND):
   a) Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ < Экзаменационный лист.
   b) Н_ВЕДОМОСТИ.ЧЛВК_ИД = 117219.
   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.ОТЧЕСТВО, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.ДАТА.
   Фильтры (AND):
   a) Н_ЛЮДИ.ОТЧЕСТВО < Сергеевич.
   b) Н_ВЕДОМОСТИ.ИД > 1250972.
   c) Н_СЕССИЯ.ДАТА > 2012-01-25.
   Вид соединения: LEFT JOIN.

Решение

```
-- 1
SELECT
    Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД,
    Н_ВЕДОМОСТИ.ИД
FROM
    Н_ВЕДОМОСТИ
LEFT JOIN
    Н_ТИПЫ_ВЕДОМОСТЕЙ
ON
    Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД = Н_ВЕДОМОСТИ.ТВ_ИД
WHERE
    Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ < 'Экзаменационный лист'
    AND Н_ВЕДОМОСТИ.ЧЛВК_ИД = 117219;
```

EXPLAIN ANALYZE

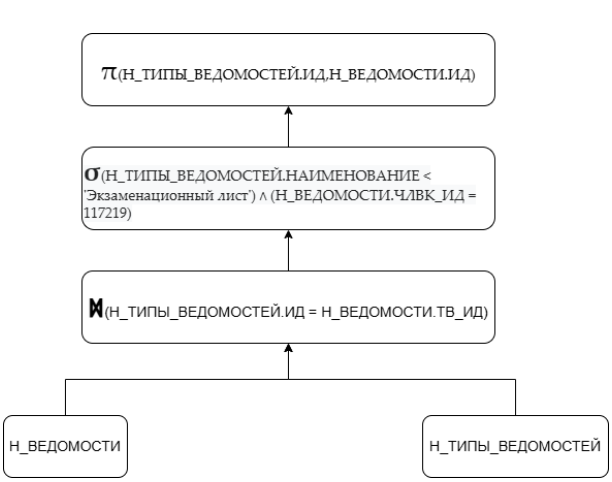
```
                                QUERY PLAN
-------------------------------------------------------------------------
 Nested Loop  (cost=0.29..200.66 rows=22 width=8) (actual time=0.032..0.082
rows=31 loops=1)
   Join Filter: ("Н_ВЕДОМОСТИ"."ТВ_ИД" = "Н_ТИПЫ_ВЕДОМОСТЕЙ"."ИД")
   Rows Removed by Join Filter: 31
   ->  Seq Scan on "Н_ТИПЫ_ВЕДОМОСТЕЙ"  (cost=0.00..1.04 rows=1 width=4)
(actual time=0.017..0.020 rows=2 loops=1)
         Filter: (("НАИМЕНОВАНИЕ")::text < 'Экзаменационный лист'::text)
         Rows Removed by Filter: 1
   ->  Index Scan using "ВЕД_ЧЛВК_FK_IFK" on "Н_ВЕДОМОСТИ"
(cost=0.29..198.81 rows=65 width=8) (actual time=0.007..0.025 rows=31
loops=2)
         Index Cond: ("ЧЛВК_ИД" = 117219)
 Planning Time: 0.182 ms
 Execution Time: 0.107 ms
```

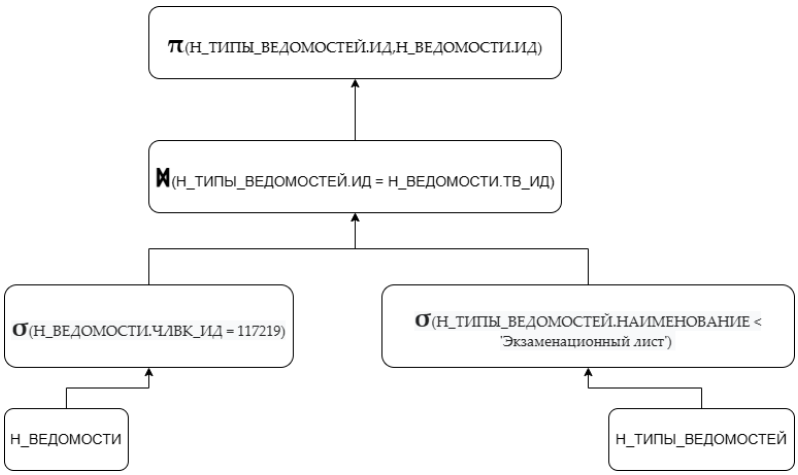
Возможные индексы

1. B-Tree INDEX для *Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ* для ускорения фильтрации *Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ < 'Экзаменационный лист'*
   В случае с операциями сравнения и строками выгодно использовать сбалансированное дерево для поиска за O(logN).

2. Hash INDEX для Н_ВЕДОМОСТИ.ЧЛВК_ИД для ускорения фильтрации Н_ВЕДОМОСТИ.ЧЛВК_ИД = 117219.
   Hash INDEX выгодно использовать конкретно для этого примера, потому что используется лишь '=' (нет операций сравнения, кроме равенства). Выполняется за O(1) – лучше, чем B-Tree в данном случае.

3. Hash INDEX для *Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД* для ускорения соединения таблиц.
   Hash INDEX выгодно использовать конкретно для этого примера, потому что используется лишь '=' при LEFT JOIN (нет операций сравнения, кроме равенства). Выполняется за O(1) – лучше, чем B-Tree в данном случае.





Оптимальный подход – второй, потому что в нем ресурсозатратная операция соединения выполняется на меньшем числе строк из-за сделанных изначально операций выборки.

2 запрос

```
-- 2
SELECT
    Н_ЛЮДИ.ОТЧЕСТВО,
    Н_ВЕДОМОСТИ.ЧЛВК_ИД,
    Н_СЕССИЯ.ДАТА
FROM
    Н_ЛЮДИ
LEFT JOIN
    Н_ВЕДОМОСТИ
ON
    Н_ЛЮДИ.ИД = Н_ВЕДОМОСТИ.ЧЛВК_ИД
LEFT JOIN
    Н_СЕССИЯ
ON
    Н_ЛЮДИ.ИД = Н_СЕССИЯ.ЧЛВК_ИД
WHERE
    Н_ЛЮДИ.ОТЧЕСТВО < 'Сергеевич'
    AND Н_ВЕДОМОСТИ.ИД > 1250972
    AND Н_СЕССИЯ.ДАТА > '2012-01-25';
```

EXPLAIN ANALYZE

```
                                   QUERY PLAN
-------------------------------------------------------------------------------
 Nested Loop  (cost=0.58..131.95 rows=4 width=32) (actual time=0.422..0.422
rows=0 loops=1)
   ->  Nested Loop  (cost=0.28..126.21 rows=1 width=36) (actual
time=0.421..0.422 rows=0 loops=1)
         ->  Seq Scan on "Н_СЕССИЯ"  (cost=0.00..117.90 rows=1 width=12)
(actual time=0.421..0.421 rows=0 loops=1)
               Filter: ("ДАТА" > '2012-01-25 00:00:00'::timestamp without
time zone)
               Rows Removed by Filter: 3752
         ->  Index Scan using "ЧЛВК_РК" on "Н_ЛЮДИ"  (cost=0.28..8.30 rows=1
width=24) (never executed)
               Index Cond: ("ИД" = "Н_СЕССИЯ"."ЧЛВК_ИД")
               Filter: (("ОТЧЕСТВО")::text < 'Сергеевич'::text)
   ->  Index Scan using "ВЕД_ЧЛВК_FK_IFK" on "Н_ВЕДОМОСТИ"  (cost=0.29..5.66
rows=8 width=4) (never executed)
         Index Cond: ("ЧЛВК_ИД" = "Н_ЛЮДИ"."ИД")
         Filter: ("ИД" > 1250972)
 Planning Time: 0.727 ms
 Execution Time: 0.459 ms
```

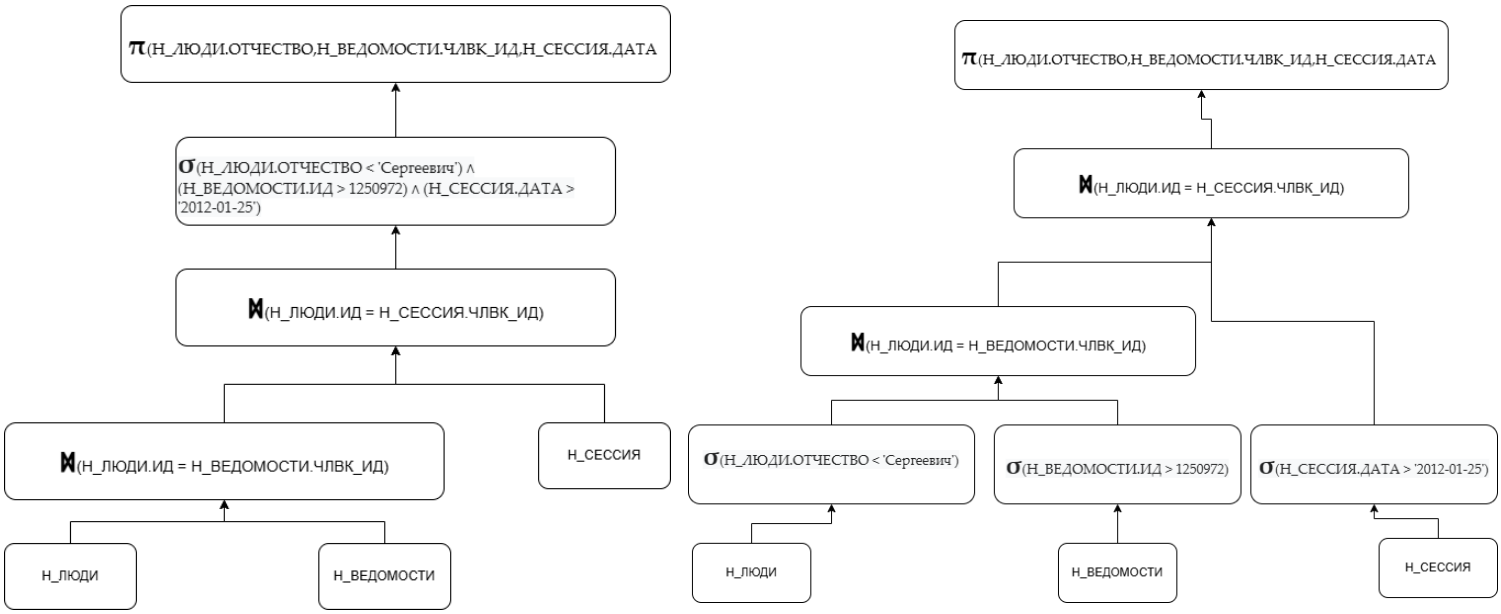
Возможные индексы

1. B-Tree INDEX для *Н_ЛЮДИ.ОТЧЕСТВО* для ускорения фильтрации *Н_ЛЮДИ.ОТЧЕСТВО < 'Сергеевич'.*
   В случае с операциями сравнения (<) выгодно использовать сбалансированное дерево для поиска за O(logN).

2. B-Tree INDEX для *Н_СЕССИЯ.ДАТА* для ускорения фильтрации *Н_СЕССИЯ.ДАТА > '2012-01-25'*
   В случае с операциями сравнения (>) выгодно использовать сбалансированное дерево для поиска за O(logN).

3. B-Tree INDEX для *Н_ВЕДОМОСТИ.ИД* для ускорения фильтрации *Н_ВЕДОМОСТИ.ИД > 1250972.*
   В случае с операциями сравнения (>) выгодно использовать сбалансированное дерево для поиска за O(logN). Генерируется автоматически (по умолчанию) для PK.

4. Hash INDEX для *Н_ВЕДОМОСТИ.ЧЛВК_ИД и Н_СЕССИЯ.ЧЛВК_ИД (отдельные)* для ускорения соединения таблиц.
   Hash INDEX выгодно использовать конкретно для этого примера, потому что используется лишь '=' при LEFT JOIN (нет операций сравнения, кроме равенства). Выполняется за O(1) – лучше, чем B-Tree в данном случае.



Оптимальный подход – второй, потому что в нем ресурсозатратная операция соединения выполняется на меньшем числе строк из-за сделанных изначально операций выборки.

Выводы

- Познакомился с концепцией индексов в PostgreSQL
- Узнал о работе оптимизатора и планировщика
- Получил общую информации о структуре БД (как хранится в файловой системе)