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.

**Содержание**

[Задание 3](#_Toc198384106)

[Решение 4](#_Toc198384107)

[Выводы 8](#_Toc198384108)

### 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'*

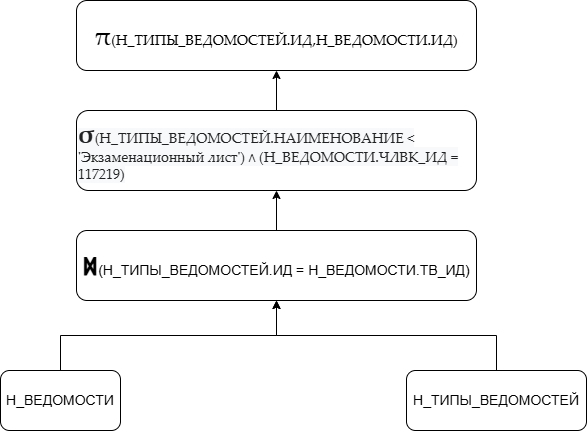
В случае с операциями сравнения и строками выгодно использовать сбалансированное дерево для поиска за O(logN).

1. 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.

1. 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\_PK" **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).

1. 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).

1. B-Tree INDEX для *Н\_ВЕДОМОСТИ.ИД* для ускорения фильтрации

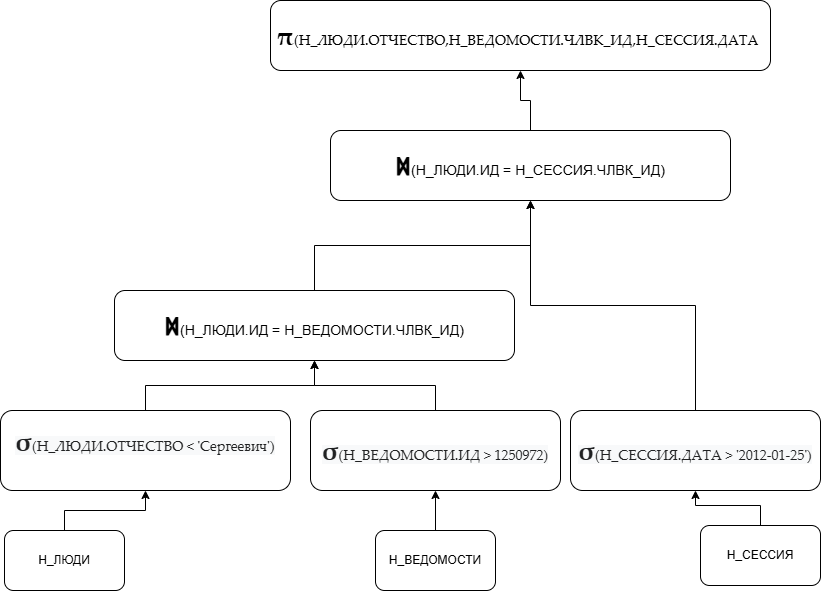
*Н\_ВЕДОМОСТИ.ИД > 1250972.*

В случае с операциями сравнения (>) выгодно использовать сбалансированное дерево для поиска за O(logN). Генерируется автоматически (по умолчанию) для PK.

1. 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)