Факультет Программной Инженерии и Компьютерной техники

**Лабораторная работа №4**

**Вариант №81642751**

Выполнил:

Чусовлянов Максим Сергеевич

made by mosh

Группа Р3107

Преподаватели:

Байрамова Хумай

Николаев Владимир Вячеславович

**Оглавление**

[Задание: 3](#__RefHeading___Toc1941_1931527120)

[Запрос №1: 4](#__RefHeading___Toc3985_2680889073)

[Индексы: 4](#__RefHeading___Toc2795_1612188114)

[Результат EXPLAIN ANALYSE: 5](#__RefHeading___Toc2797_1612188114)

[Запрос №2: 6](#__RefHeading___Toc3985_26808890731)

[Индексы: 6](#__RefHeading___Toc2795_16121881141)

[Результат EXPLAIN ANALYSE: 7](#__RefHeading___Toc2797_16121881141)

[Вывод: 8](#__RefHeading___Toc973_1545597831)

# **Задание:**

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?

Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:

Таблицы: 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.ДАТА = 2022-06-08.

c) Н\_ВЕДОМОСТИ.ДАТА < 2022-06-08.

Вид соединения: LEFT JOIN.

Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:

Таблицы: Н\_ЛЮДИ, Н\_ОБУЧЕНИЯ, Н\_УЧЕНИКИ.

Вывести атрибуты: Н\_ЛЮДИ.ОТЧЕСТВО, Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД, Н\_УЧЕНИКИ.НАЧАЛО.

Фильтры: (AND)

a) Н\_ЛЮДИ.ОТЧЕСТВО = Сергеевич.

b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = 163276.

Вид соединения: RIGHT JOIN.

# **Запрос №1:** Таблицы: Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ.

Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД, Н\_ВЕДОМОСТИ.ИД.

Фильтры (AND):

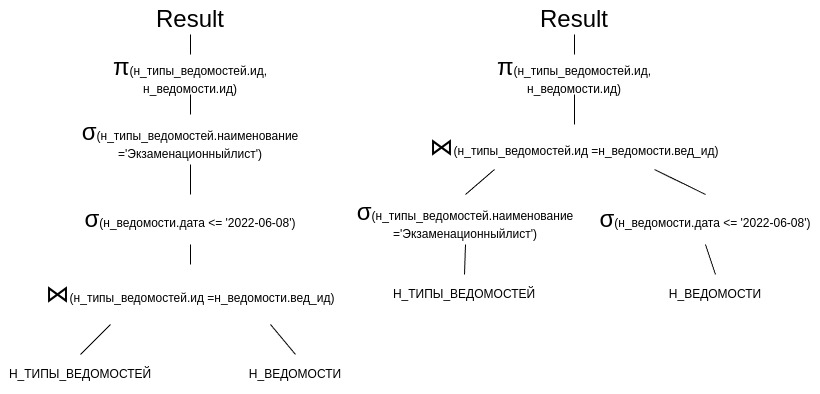
a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ = Экзаменационный лист.

b) Н\_ВЕДОМОСТИ.ДАТА = 2022-06-08.

c) Н\_ВЕДОМОСТИ.ДАТА < 2022-06-08.

Вид соединения: LEFT JOIN.

|  |
| --- |
| SELECT Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД, Н\_ВЕДОМОСТИ.ИД FROM Н\_ТИПЫ\_ВЕДОМОСТЕЙ LEFT JOIN Н\_ВЕДОМОСТИ ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ВЕД\_ИД WHERE Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ = 'Экзаменационный лист' AND Н\_ВЕДОМОСТИ.ДАТА <= '2022-06-08'; |



Оптимальным является план №2, так как он производит объединение таблиц по ранее выбранным атрибутам, а не по таблицам целиком.

## **Индексы:**

|  |
| --- |
| CREATE INDEX "ИНДЕКС\_ТИПЫ\_ВЕДОМОСТЕЙ\_ИД" ON "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" USING hash("ИД"); CREATE INDEX "ИНДЕКС\_ВЕДОМОСТИ\_ИД" ON "Н\_ВЕДОМОСТИ" USING hash("ИД"); CREATE INDEX "ИНДЕКС\_ТИПЫ\_ВЕДОМОСТЕЙ\_НАИМ" ON "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" USING hash("НАИМЕНОВАНИЕ"); CREATE INDEX "ИНДЕКС\_ВЕДОМОСТИ\_ДАТА" ON "Н\_ВЕДОМОСТИ" USING btree("ДАТА"); |

Добавление этих индексов должно ускорить выполнение запросов, так как по перечисленным полям происходит выборка с использованием оператора сравнения. Так же быстрее будет происходить соединение таблиц. В последнем случае используются операторы сравнения „>“ и „<“, так что эффективнее использовать btree. В первых трех случаях используется прямое сравнение, так что эффективнее использовать хэш-индекс.

При добавлении индексов планы выполнения запросов изменятся, так как будет происходить индексный скан и Hash Join станет быстрее благодаря индексам.

## **Результат EXPLAIN ANALYSE:**

|  |
| --- |
| Hash Join (cost=1.05..5468.88 rows=1 width=8) (actual time=22.249..22.251 rows=0 loops=3)  Hash Cond: ("Н\_ВЕДОМОСТИ"."ВЕД\_ИД" = "Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."ИД")  -> Parallel Seq Scan on "Н\_ВЕДОМОСТИ" (cost=0.00..5224.54 rows=92683 width=8) (actual time=0.016..14.163 rows=74147 loops=3)  Filter: ("ДАТА" <= '2022-06-08 00:00:00'::timestamp without time zone)  -> Hash (cost=1.04..1.04 rows=1 width=4) (actual time=0.028..0.029 rows=1 loops=3)  Buckets: 1024 Batches: 1 Memory Usage: 9kB  -> Seq Scan on "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" (cost=0.00..1.04 rows=1 width=4) (actual time=0.016..0.017 rows=1 loops=3)  Filter: (("НАИМЕНОВАНИЕ")::text = 'Экзаменационный лист'::text)  Rows Removed by Filter: 2  Planning Time: 0.311 ms  Execution Time: 28.339 ms |

# **Запрос №2:** Таблицы: Н\_ЛЮДИ, Н\_ОБУЧЕНИЯ, Н\_УЧЕНИКИ.

Вывести атрибуты: Н\_ЛЮДИ.ОТЧЕСТВО, Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД, Н\_УЧЕНИКИ.НАЧАЛО.

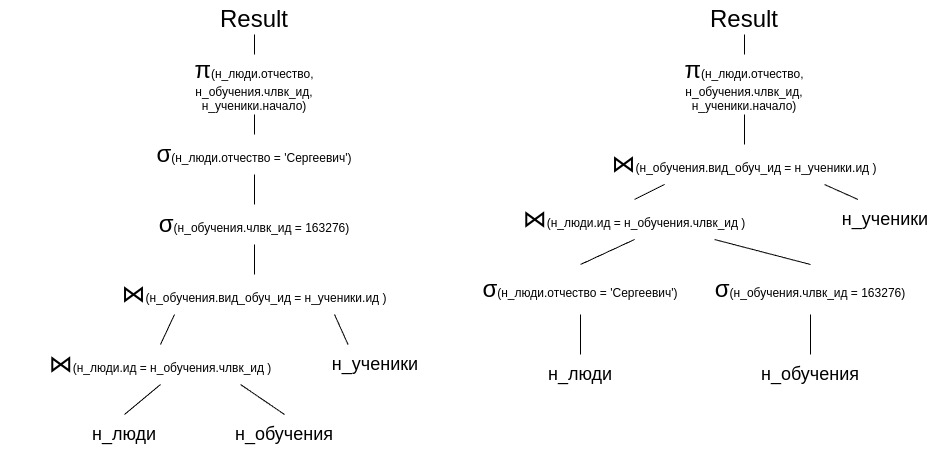
Фильтры: (AND)

a) Н\_ЛЮДИ.ОТЧЕСТВО = Сергеевич.

b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = 163276.

Вид соединения: RIGHT JOIN.

|  |
| --- |
| SELECT "Н\_ЛЮДИ"."ОТЧЕСТВО", "Н\_ОБУЧЕНИЯ"."ЧЛВК\_ИД", "Н\_УЧЕНИКИ"."НАЧАЛО" FROM "Н\_ЛЮДИ" RIGHT JOIN "Н\_ОБУЧЕНИЯ" ON "Н\_ЛЮДИ"."ИД" = "Н\_ОБУЧЕНИЯ"."ЧЛВК\_ИД" RIGHT JOIN "Н\_УЧЕНИКИ" ON "Н\_ОБУЧЕНИЯ"."ВИД\_ОБУЧ\_ИД" = "Н\_УЧЕНИКИ"."ИД" WHERE "Н\_ЛЮДИ"."ОТЧЕСТВО" = 'Сергеевич' AND "Н\_ОБУЧЕНИЯ"."ЧЛВК\_ИД" = 163276; |

1. 
2. Оптимальным является план №2, так как он производит объединение таблиц по ранее выбранным атрибутам, а не по таблицам целиком.

## **Индексы:**

|  |
| --- |
| CREATE INDEX "ИНДЕКС\_ЛЮДИ\_ОТЧЕСТВО" ON "Н\_ЛЮДИ" USING hash("ОТЧЕСТВО"); CREATE INDEX "ИНДЕКС\_ОБУЧЕНИЯ\_ЧЛВК\_ИД" ON "Н\_ОБУЧЕНИЯ" USING hash("ЧЛВК\_ИД"); CREATE INDEX "ИНДЕКС\_УЧЕНИКИ\_ИД" ON "Н\_УЧЕНИКИ" USING hash("ИД"); CREATE INDEX "ИНДЕКС\_ОБУЧЕНИЯ\_ВИД\_ОБУЧЕНИЯ\_ИД" ON "Н\_ОБУЧЕНИЯ" USING hash("ВИД\_ОБУЧ\_ИД"); CREATE INDEX "ИНДЕКС\_ЛЮДИ\_ИД" ON "Н\_ЛЮДИ" USING hash("ИД"); |

Добавление этих индексов должно ускорить выполнение запросов, так как по перечисленным полям происходит выборка с использованием оператора сравнения. Так же быстрее будет происходить соединение таблиц. Во всех случаях происходит прямое сравнение, так что эффективнее использовать хэш-индексы.

При добавлении индексов планы выполнения запросов изменятся, так как будет происходить индексный скан и Nested Loop Join станет быстрее благодаря индексам.

## **Результат EXPLAIN ANALYSE:**

|  |
| --- |
| Nested Loop (cost=8.88..17.16 rows=1 width=32) (actual time=0.054..0.055 rows=0 loops=1)  -> Merge Join (cost=8.60..8.85 rows=1 width=12) (actual time=0.042..0.044 rows=1 loops=1)  Merge Cond: ("Н\_УЧЕНИКИ"."ИД" = "Н\_ОБУЧЕНИЯ"."ВИД\_ОБУЧ\_ИД")  -> Index Scan using "УЧЕН\_PK" on "Н\_УЧЕНИКИ" (cost=0.29..2664.40 rows=23311 width=12) (actual time=0.007..0.008 rows=2 loops=1)  -> Sort (cost=8.31..8.32 rows=1 width=8) (actual time=0.029..0.030 rows=1 loops=1)  Sort Key: "Н\_ОБУЧЕНИЯ"."ВИД\_ОБУЧ\_ИД"  Sort Method: quicksort Memory: 25kB  -> Index Scan using "ОБУЧ\_ЧЛВК\_FK\_I" on "Н\_ОБУЧЕНИЯ" (cost=0.28..8.30 rows=1 width=8) (actual time=0.013..0.014 rows=1 loops=1)  Index Cond: ("ЧЛВК\_ИД" = 163276)  -> Index Scan using "ЧЛВК\_PK" on "Н\_ЛЮДИ" (cost=0.28..8.30 rows=1 width=24) (actual time=0.008..0.008 rows=0 loops=1)  Index Cond: ("ИД" = 163276)  Filter: (("ОТЧЕСТВО")::text = 'Сергеевич'::text)  Rows Removed by Filter: 1  Planning Time: 0.412 ms  Execution Time: 0.102 ms |

# **Вывод:** Во время выполнения данной лабораторной работы я научился оптимизировать запросы, составлять наиболее выгодный план выполнения запросов, используя для этого подходящие виды индексов.