Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Лабораторная работа №4

По дисциплине

Базы Данных

Выполнил студент группы P3117:

Кудрявцева Руслана Сергеевна

Преподаватель:

Чупанов Аликылыч Алибекович

Санкт-Петербург 2025 г.

**Оглавление**

[Задание 3](#__RefHeading___Toc3039_3602381616)

[Запрос №1 4](#__RefHeading___Toc3041_3602381616)

[Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ. Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД, Н\_ВЕДОМОСТИ.ДАТА. Фильтры (AND): a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ = Ведомость. b) Н\_ВЕДОМОСТИ.ДАТА < 1998-01-05. c) Н\_ВЕДОМОСТИ.ДАТА > 1998-01-05. Вид соединения: INNER JOIN. 4](#__RefHeading___Toc3043_3602381616)

[Индексы: 5](#__RefHeading___Toc2795_1612188114)

[Результат EXPLAIN ANALYSE: 5](#__RefHeading___Toc2797_1612188114)

[Запрос №2 6](#__RefHeading___Toc3045_3602381616)

[Индексы: 6](#__RefHeading___Toc2795_16121881141)

[Результат EXPLAIN ANALYSE: 7](#__RefHeading___Toc2797_16121881141)

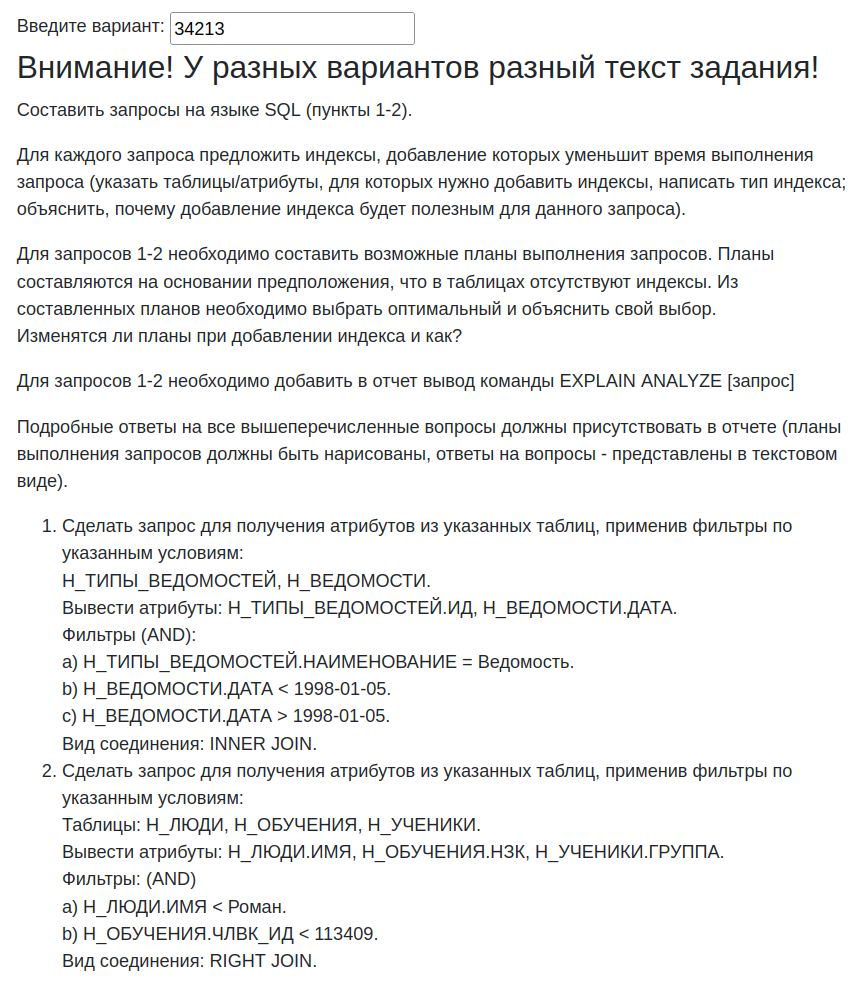
[Вывод: Во время выполнения данной лабораторной работы я научилась оптимизировать запросы, составлять наиболее выгодный план выполнения запросов, используя для этого подходящие виды индексов. 7](#__RefHeading___Toc3047_3602381616)

# Задание

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

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.



# Запрос №1

# Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ. Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД, Н\_ВЕДОМОСТИ.ДАТА. Фильтры (AND): a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ = Ведомость. b) Н\_ВЕДОМОСТИ.ДАТА < 1998-01-05. c) Н\_ВЕДОМОСТИ.ДАТА > 1998-01-05. Вид соединения: INNER JOIN.

|  |
| --- |
| SELECT "Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."ИД", "Н\_ВЕДОМОСТИ"."ДАТА" FROM "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" INNER JOIN "Н\_ВЕДОМОСТИ" ON "Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."ИД" = "Н\_ВЕДОМОСТИ"."ВЕД\_ИД" WHERE "Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."НАИМЕНОВАНИЕ" = 'Ведомость' AND "Н\_ВЕДОМОСТИ"."ДАТА" > '1998-01-05' AND "Н\_ВЕДОМОСТИ"."ДАТА" < '1998-01-05'; |



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

## **Индексы:**

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

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 втором случае используется операторы сравнения „>“ и „<“, так что эффективнее использовать btree. В первом и последнем случаях используется прямое сравнение, так что эффективнее использовать хэш-индекс.

Во всех случаеях ключ создается на поле, по которому идет фильтрация. В первом индексе идет соединение JOIN от Н\_ТИПЫ\_ВЕДОМОСТЕЙ к Н\_ВЕДОМОСТИ, и именно там происходит фильтрация по поиску ВЕД\_ИД. Так же и с остальными.

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

## **Результат EXPLAIN ANALYSE:**

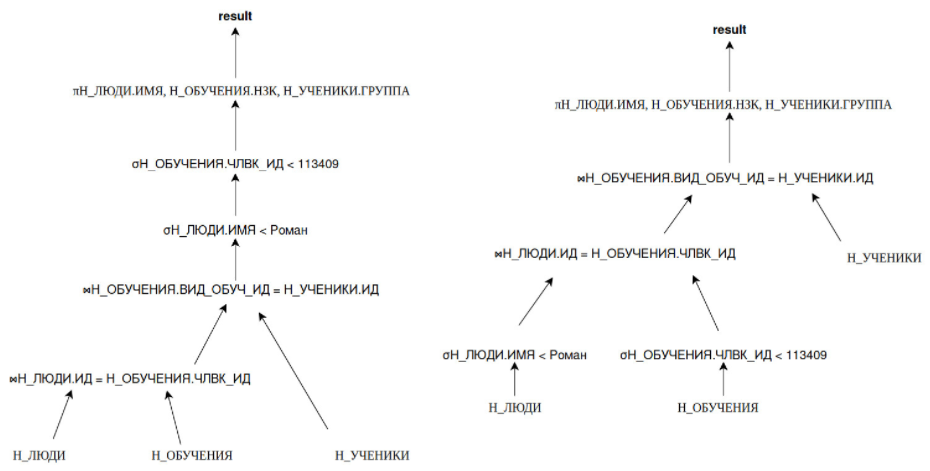
|  |
| --- |
| Nested Loop (cost=15.69..2530.64 rows=1 width=12)  Join Filter: ("Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."ИД" = "Н\_ВЕДОМОСТИ"."ВЕД\_ИД")  -> Seq Scan on "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" (cost=0.00..1.04 rows=1 width=4)  Filter: (("НАИМЕНОВАНИЕ")::text = 'Ведомость'::text)  -> Bitmap Heap Scan on "Н\_ВЕДОМОСТИ" (cost=15.69..2515.70 rows=1112 width=12)  Recheck Cond: (("ДАТА" > '1998-01-05 00:00:00'::timestamp without time zone) AND ("ДАТА" < '1998-01-05 00:00:00'::timestamp without time zone))  -> Bitmap Index Scan on "ВЕД\_ДАТА\_I" (cost=0.00..15.42 rows=1112 width=0)  Index Cond: (("ДАТА" > '1998-01-05 00:00:00'::timestamp without time zone) AND ("ДАТА" < '1998-01-05 00:00:00'::timestamp without time zone)) |

# Запрос №2

Таблицы: Н\_ЛЮДИ, Н\_ОБУЧЕНИЯ, Н\_УЧЕНИКИ.  
Вывести атрибуты: Н\_ЛЮДИ.ИМЯ, Н\_ОБУЧЕНИЯ.НЗК, Н\_УЧЕНИКИ.ГРУППА.

Фильтры: (AND)  
a) Н\_ЛЮДИ.ИМЯ < Роман.  
b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 113409.  
Вид соединения: RIGHT JOIN.

|  |
| --- |
| SELECT "Н\_ЛЮДИ"."ИМЯ", "Н\_ОБУЧЕНИЯ"."НЗК", "Н\_УЧЕНИКИ"."ГРУППА" 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"."ИМЯ" < 'Роман' AND "Н\_ОБУЧЕНИЯ"."ЧЛВК\_ИД" < 113409; |

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\_Н\_ЛЮДИ\_ИМЯ" ON "Н\_ЛЮДИ" USING btree("ИМЯ"); CREATE INDEX "ИНДЕКС\_Н\_ОБУЧЕНИЯ\_ЧЛВК\_ИД" ON "Н\_ОБУЧЕНИЯ" USING btree("ЧЛВК\_ИД"); CREATE INDEX "ИНДЕКС\_Н\_УЧЕНИКИ\_ИД" ON "Н\_УЧЕНИКИ" USING hash("ИД"); CREATE INDEX "ИНДЕКС\_Н\_ОБУЧЕНИЯ\_ВИД\_ОБУЧЕНИЯ\_ИД" ON "Н\_ОБУЧЕНИЯ" USING btree("ВИД\_ОБУЧ\_ИД"); |

Добавление этих индексов должно ускорить выполнение запроса, так как по перечисленным полям происходит выборка с использованием операторов сравнения. Быстрее будет происходить соединение таблиц и фильтрация.

Для полей ИМЯ и ЧЛВК\_ИД используются операторы „<“, поэтому здесь нужен именно btree, так как он поддерживает поиск по диапазонам. Для ВИД\_ОБУЧ\_ИД также подходит btree, потому что он используется при соединении, а btree универсален и эффективен при равенстве. Для поля ИД в таблице Н\_УЧЕНИКИ, где используется точное сравнение, можно использовать hash, так как в этом случае он будет быстрее. Во всех случаях индексы создаются на поля, по которым идёт фильтрация или соединение в запросе.

При добавлении индексов планы выполнения запросов изменятся, так как будет происходить индексный или хеш-скан. Использование Hash Join и Index Scan ускоряет выполнение за счёт сокращения объема перебираемых строк.

## **Результат EXPLAIN ANALYSE:**

|  |
| --- |
| Nested Loop (cost=79.33..288.26 rows=351 width=23)  -> Hash Join (cost=79.04..267.91 rows=351 width=23)  Hash Cond: ("Н\_ЛЮДИ"."ИД" = "Н\_ОБУЧЕНИЯ"."ЧЛВК\_ИД")  -> Seq Scan on "Н\_ЛЮДИ" (cost=0.00..163.97 rows=4277 width=17)  Filter: (("ИМЯ")::text < 'Роман'::text)  -> Hash (cost=73.79..73.79 rows=420 width=14)  -> Bitmap Heap Scan on "Н\_ОБУЧЕНИЯ" (cost=11.54..73.79 rows=420 width=14)  Recheck Cond: ("ЧЛВК\_ИД" < 113409)  -> Bitmap Index Scan on "ОБУЧ\_ЧЛВК\_FK\_I" (cost=0.00..11.43 rows=420 width=0)  Index Cond: ("ЧЛВК\_ИД" < 113409)  -> Memoize (cost=0.30..3.88 rows=1 width=8)  Cache Key: "Н\_ОБУЧЕНИЯ"."ВИД\_ОБУЧ\_ИД"  Cache Mode: logical  -> Index Scan using "УЧЕН\_PK" on "Н\_УЧЕНИКИ" (cost=0.29..3.87 rows=1 width=8)  Index Cond: ("ИД" = "Н\_ОБУЧЕНИЯ"."ВИД\_ОБУЧ\_ИД") |

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