Отчет

**по лабораторной работе №4**

**Вариант - 9871**

по дисциплине «Базы данных»

Выполнил: Караганов П.Э., группа P3110

Преподаватель: Гаврилов А. В.

Николаев В. В.

Оглавление

[Текст задания 3](#_Toc196311895)

[Запрос №1 3](#_Toc196311896)

[Индексы: 4](#_Toc196311897)

[Результат EXPLAIN ANALYSE: 5](#_Toc196311898)

[Запрос №2 5](#_Toc196311899)

[Индексы: 6](#_Toc196311900)

[Результат EXPLAIN ANALYSE: 6](#_Toc196311901)

[Вывод 6](#_Toc196311902)

# Текст задания

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\_ЛЮДИ, Н\_СЕССИЯ.

Вывести атрибуты: Н\_ЛЮДИ.ИМЯ, Н\_СЕССИЯ.УЧГОД.

Фильтры (AND):

a) Н\_ЛЮДИ.ОТЧЕСТВО < Георгиевич.

b) Н\_СЕССИЯ.ЧЛВК\_ИД < 151200.

c) Н\_СЕССИЯ.ЧЛВК\_ИД < 100622.

Вид соединения: RIGHT JOIN.

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

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

Вывести атрибуты: Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ОБУЧЕНИЯ.НЗК, Н\_УЧЕНИКИ.ИД.

Фильтры: (AND)

a) Н\_ЛЮДИ.ОТЧЕСТВО > Александрович.

b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = 105590.

Вид соединения: INNER JOIN.

# Запрос №1

Таблицы: Н\_ЛЮДИ, Н\_СЕССИЯ.

Вывести атрибуты: Н\_ЛЮДИ.ИМЯ, Н\_СЕССИЯ.УЧГОД.

Фильтры (AND):

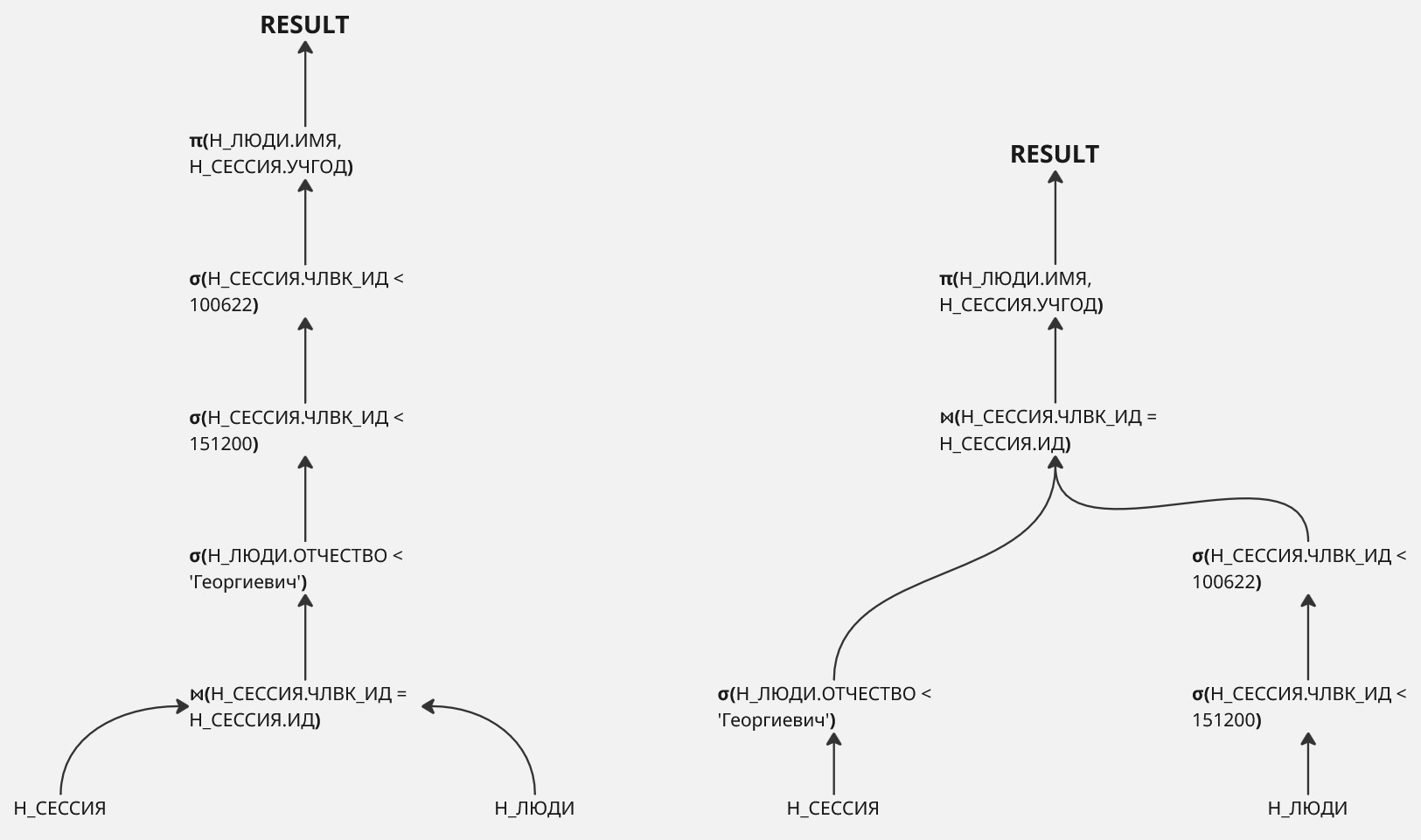
a) Н\_ЛЮДИ.ОТЧЕСТВО < Георгиевич.

b) Н\_СЕССИЯ.ЧЛВК\_ИД < 151200.

c) Н\_СЕССИЯ.ЧЛВК\_ИД < 100622.

Вид соединения: RIGHT JOIN.

SELECT P."ИМЯ", S."УЧГОД"  
FROM "Н\_СЕССИЯ" S  
RIGHT JOIN "Н\_ЛЮДИ" P ON S."ЧЛВК\_ИД" = P."ИД"  
WHERE P."ОТЧЕСТВО" < 'Георгиевич'  
 AND S."ЧЛВК\_ИД" < 151200  
 AND S."ЧЛВК\_ИД" < 100622;



Второй план более оптимальнее, потому что в нём сначала происходит фильтрация и только потом объединение таблиц, по сравнению с первым, где сначала происходит объединение таблиц и только потом фильтрация, тем самым загромождая память лишними данными

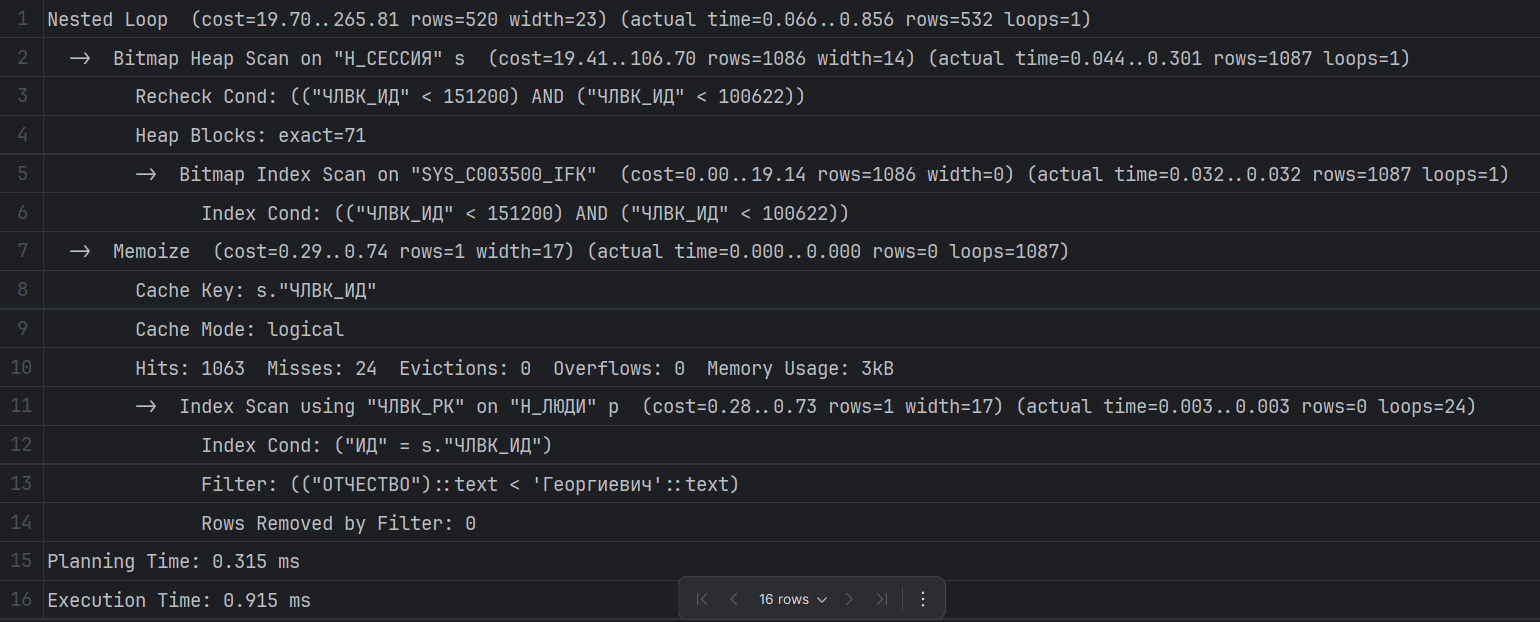
## Индексы:

CREATE INDEX "ИНДЕКС\_ЛЮДИ\_ОТЧЕСТВО" ON "Н\_ЛЮДИ" USING BTREE ("ОТЧЕСТВО");  
CREATE INDEX "ИНДЕКС\_СЕССИЯ\_ЧЛВК\_ИД" ON "Н\_СЕССИЯ" USING BTREE ("ЧЛВК\_ИД");  
CREATE INDEX "ИНДЕКС\_ЛЮДИ\_ИД" ON "Н\_ЛЮДИ" USING HASH ("ИД");

Эти индексы должны ускорить работу нашего запроса, так как в нём используются атрибуты данных таблиц и операторы сравнения. Также будет быстрее происходить соединение таблиц. В нашем запросе есть сравнение в атрибуте «Н\_ЛЮДИ».«ОТЧЕСТВО» поэтому там лучше использовать btree. Также мы сравниваем «Н\_СЕССИЯ»."ЧЛВК\_ИД" поэтому там тоже нужно создать btree индекс. В JOIN мы делаем прямое сравнение с "Н\_ЛЮДИ"."ИД", поэтому здесь поможет hash-индекс для быстродействия.

При добавлении индексов планы выполнения запросов изменятся, так как будет происходить Index scan

## Результат EXPLAIN ANALYSE:



# Запрос №2

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

Вывести атрибуты: Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ОБУЧЕНИЯ.НЗК, Н\_УЧЕНИКИ.ИД.

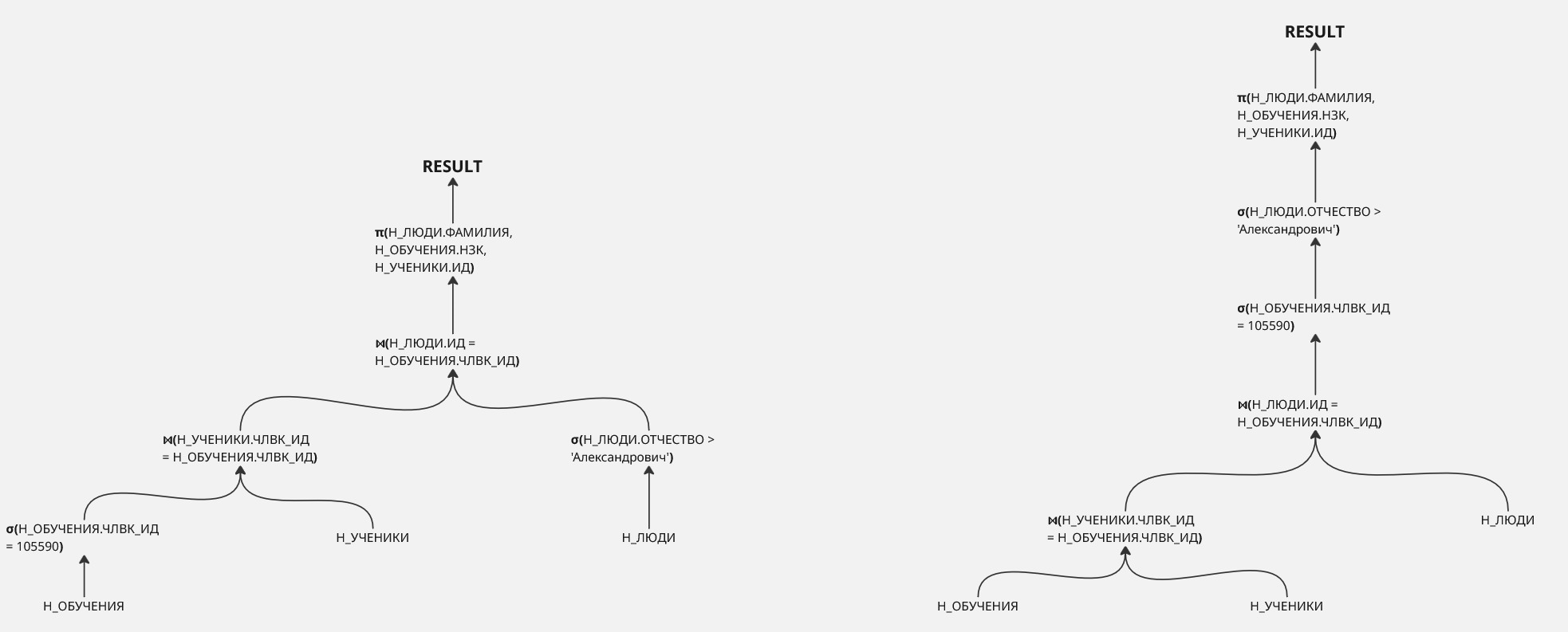
Фильтры: (AND)

a) Н\_ЛЮДИ.ОТЧЕСТВО > Александрович.

b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = 105590.

Вид соединения: INNER JOIN.

SELECT P."ФАМИЛИЯ", E."НЗК", S."ИД"  
FROM "Н\_УЧЕНИКИ" S  
JOIN "Н\_ОБУЧЕНИЯ" E ON S."ЧЛВК\_ИД" = E."ЧЛВК\_ИД"  
JOIN "Н\_ЛЮДИ" P ON P."ИД" = E."ЧЛВК\_ИД"  
WHERE P."ОТЧЕСТВО" > 'Александрович'  
 AND E."ЧЛВК\_ИД" = 105590



Левый план (левостороннее дерево) оптимальнее правого так как там происходит сначала фильтрация, а только потом соединение тем самым в итоговую таблицу идёт меньше данных. В правом плане мы сначала соединяем все таблицы и только потом начинаем выборку.

## Индексы:

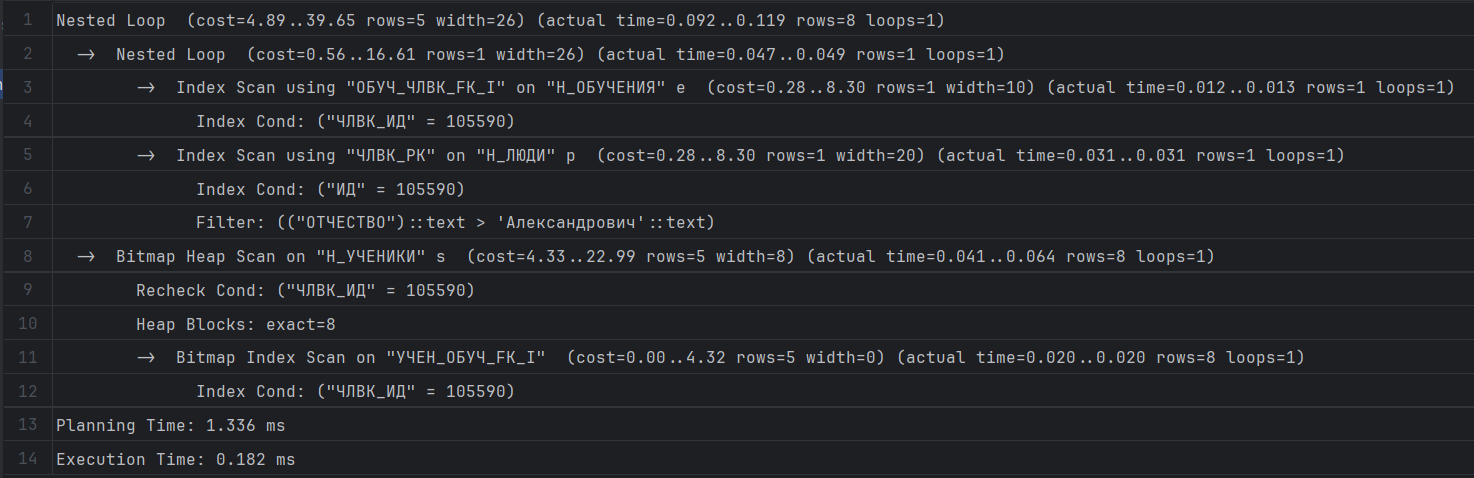
CREATE INDEX "idx\_люди\_отчество" ON "Н\_ЛЮДИ" USING BTREE ("ОТЧЕСТВО");  
CREATE INDEX "idx\_обучения\_члвк\_ид" ON "Н\_ОБУЧЕНИЯ" USING HASH ("ЧЛВК\_ИД");  
CREATE INDEX "idx\_люди\_ид" ON "Н\_ЛЮДИ" USING HASH ("ИД");

CREATE INDEX "idx\_ученики\_ид" ON "Н\_УЧЕНИКИ" USING HASH ("ЧЛВК\_ИД");

Добавление этих индексов должно ускорить выполнение запросов. Первый индекс должен оптимизировать запросы типа Н\_ЛЮДИ.ОТЧЕСТВО > 'Александрович'. Второй, третий и четвёртый индексы должны обеспечить быстродействие при прямых сравнениях, а значит JOIN-ы и запросы типа Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = 105590 будут работать быстрее.

План поменяется из-за добавления индексов, Nested Loop станет быстрее работать из-за Index Scan и Bitmap Index Scan

## **Результат EXPLAIN ANALYSE:**



# Вывод

За время выполнения лабораторной ям понял что такое индексы в SQL, как работают btree и hash индексы, как работают соединения, что такое планы выполнения запроса и какие scan бывают