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

Факультет программной инженерии и компьютерной техники

Базы данных

**Лабораторная работа №4**

Вариант 77

Выполнила:

Агаларова Айсел Нубарековна

Группа P3130

Проверил:

Перцев Тимофей Сергеевич

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?

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.НАИМЕНОВАНИЕ, Н\_ВЕДОМОСТИ.ЧЛВК\_ИД.  
   Фильтры (AND):  
   a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3.  
   b) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590.  
   c) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 153285.  
   Вид соединения: LEFT JOIN.
2. 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.ИД.  
   Фильтры: (AND)  
   a) Н\_ЛЮДИ.ИМЯ < Ярослав.  
   b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276.  
   Вид соединения: INNER JOIN.

**Реализация запросов на SQL:**

Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ.  
Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н\_ВЕДОМОСТИ.ЧЛВК\_ИД.  
Фильтры (AND):  
a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3.  
b) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590.  
c) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 153285.  
Вид соединения: LEFT JOIN.

SELECT

Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ,

Н\_ВЕДОМОСТИ.ЧЛВК\_ИД

FROM

Н\_ТИПЫ\_ВЕДОМОСТЕЙ

LEFT JOIN

Н\_ВЕДОМОСТИ ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ТВ\_ИД

WHERE

Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3 AND

Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590 AND

Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 153285;

Взаимоисключающее условие на Н\_ВЕДОМОСТИ.ЧЛВК\_ИД ---> Замена на AND на OR между этими фильтрами

SELECT

Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ,

Н\_ВЕДОМОСТИ.ЧЛВК\_ИД

FROM

Н\_ТИПЫ\_ВЕДОМОСТЕЙ

LEFT JOIN

Н\_ВЕДОМОСТИ ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ТВ\_ИД

WHERE

Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3 AND

(Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590 OR Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 153285);

2) 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.ИД.  
Фильтры: (AND)  
a) Н\_ЛЮДИ.ИМЯ < Ярослав.  
b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276.  
Вид соединения: INNER JOIN.

SELECT

Н\_ЛЮДИ.ОТЧЕСТВО,

Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД,

Н\_УЧЕНИКИ.ИД

FROM

Н\_ЛЮДИ

INNER JOIN

Н\_ОБУЧЕНИЯ ON Н\_ЛЮДИ.ИД = Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД

INNER JOIN

Н\_УЧЕНИКИ ON Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = Н\_УЧЕНИКИ.ЧЛВК\_ИД

WHERE

Н\_ЛЮДИ.ИМЯ < 'Ярослав' AND

Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276;

**Оптимизация запросов с помощью индексов:**  
  
1) Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ.  
Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н\_ВЕДОМОСТИ.ЧЛВК\_ИД.  
Фильтры (AND):  
a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3.  
b) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590.  
c) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 153285.  
Вид соединения: LEFT JOIN.

SELECT

Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ,

Н\_ВЕДОМОСТИ.ЧЛВК\_ИД

FROM

Н\_ТИПЫ\_ВЕДОМОСТЕЙ

LEFT JOIN

Н\_ВЕДОМОСТИ ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ТВ\_ИД

WHERE

Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3 AND

(Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590 OR Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 153285);

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Атрибут | Тип индекса | Назначение |
| Н\_ВЕДОМОСТИ | ТВ\_ИД | B-tree | |  | | --- | |  |  |  | | --- | | Ускоряет LEFT JOIN, так как позволяет **быстро найти все строки в** Н\_ВЕДОМОСТИ**, связанные с** ИД = 3 | |
| Н\_ВЕДОМОСТИ | ЧЛВК\_ИД | B-tree | Это фильтр по диапазону. С**разу находит** все значения, которые попадают в условие. Без индекса происходил бы перебор всей таблицы. |

DROP INDEX IF EXISTS ВЕД\_ТИП\_ИД\_ИX;

DROP INDEX IF EXISTS ВЕД\_ТВ\_ИД\_ИX;

DROP INDEX IF EXISTS ВЕД\_ЧЛВК\_ИД\_ИX;

CREATE INDEX ВЕД\_ТИП\_ИД\_ИX ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ (ИД);

CREATE INDEX ВЕД\_ТВ\_ИД\_ИX ON Н\_ВЕДОМОСТИ (ТВ\_ИД);

CREATE INDEX ВЕД\_ЧЛВК\_ИД\_ИX ON Н\_ВЕДОМОСТИ (ЧЛВК\_ИД);

2) 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.ИД.  
Фильтры: (AND)  
a) Н\_ЛЮДИ.ИМЯ < Ярослав.  
b) Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276.  
Вид соединения: INNER JOIN.

SELECT

Н\_ЛЮДИ.ОТЧЕСТВО,

Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД,

Н\_УЧЕНИКИ.ИД

FROM

Н\_ЛЮДИ

INNER JOIN

Н\_ОБУЧЕНИЯ ON Н\_ЛЮДИ.ИД = Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД

INNER JOIN

Н\_УЧЕНИКИ ON Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = Н\_УЧЕНИКИ.ЧЛВК\_ИД

WHERE

Н\_ЛЮДИ.ИМЯ < 'Ярослав' AND

Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276;

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Атрибут | Тип индекса | Назначение |
| Н\_ЛЮДИ | ИМЯ | B-tree | Ускоряет лексикографическое сравнение по алфавиту ИМЯ < 'Ярослав' |
| Н\_ОБУЧЕНИЯ | ЧЛВК\_ИД | B-tree | |  | | --- | |  |  |  | | --- | | Работает и при соединении (JOIN), и при фильтрации. Ускоряет поиск нужных записей в Н\_ОБУЧЕНИЯ по ЧЛВК\_ИД | |
| Н\_УЧЕНИКИ | ЧЛВК\_ИД | B-tree | Ускоряет соединение по нужному ЧЛВК\_ИД без полного сканирования таблицы |

DROP INDEX IF EXISTS ЛЮДИ\_ИМЯ\_ИX;

DROP INDEX IF EXISTS ОБУЧ\_ЧЛВК\_ИД\_ИX;

DROP INDEX IF EXISTS УЧЕНИКИ\_ЧЛВК\_ИД\_ИX;

CREATE INDEX ЛЮДИ\_ИМЯ\_ИX ON Н\_ЛЮДИ (ИМЯ);

CREATE INDEX ОБУЧ\_ЧЛВК\_ИД\_ИX ON Н\_ОБУЧЕНИЯ (ЧЛВК\_ИД);

CREATE INDEX УЧЕНИКИ\_ЧЛВК\_ИД\_ИX ON Н\_УЧЕНИКИ (ЧЛВК\_ИД);

**Возможные планы выполнения запросов:**

**Запрос 1:**

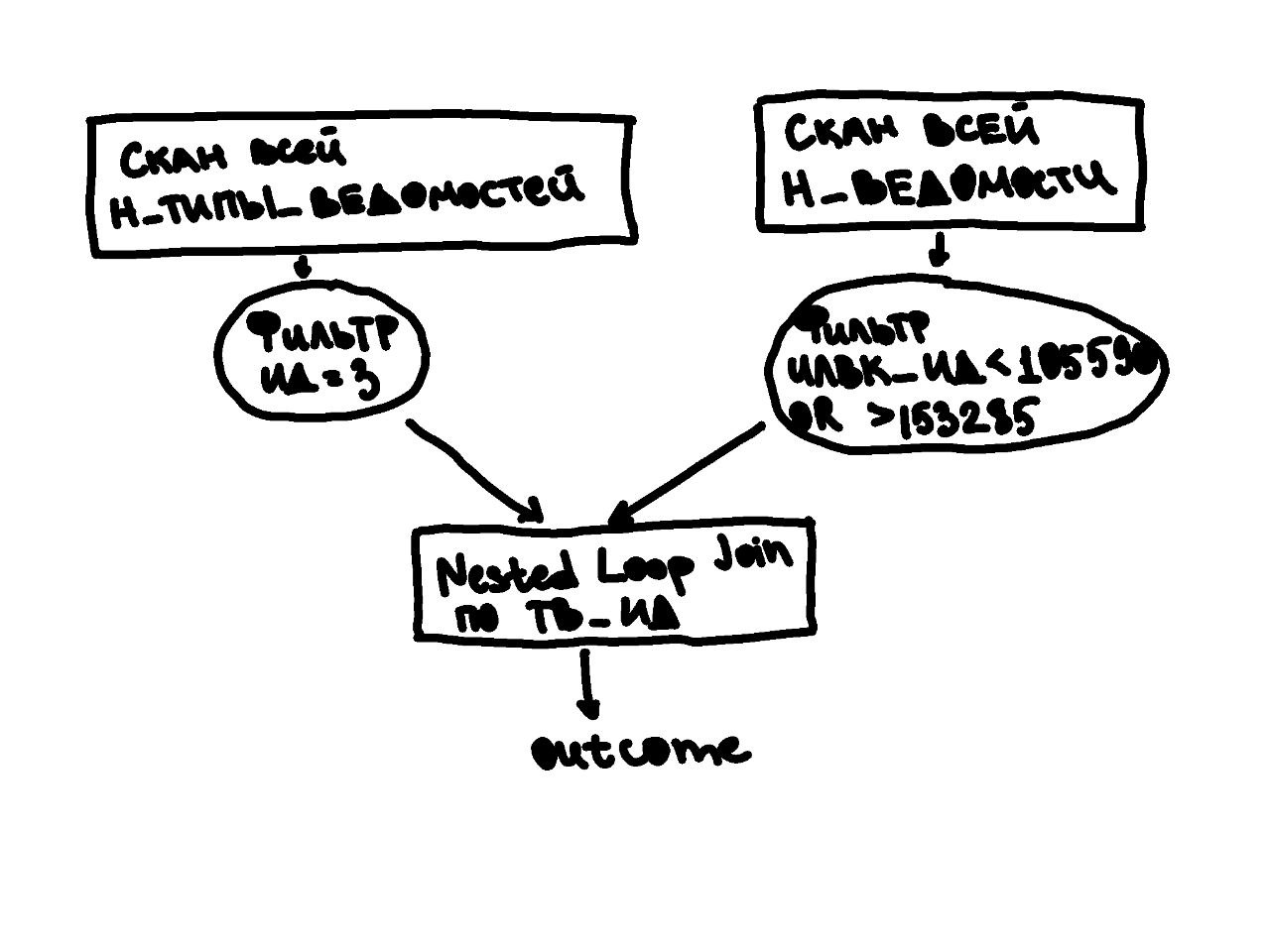
**План 1:**

**Полный скан таблицы** Н\_ТИПЫ\_ВЕДОМОСТЕЙ — выбираются все строки, затем фильтрация по ИД = 3.

**Полный скан таблицы** Н\_ВЕДОМОСТИ — выбираются все строки, затем фильтрация по ЧЛВК\_ИД < 105590 OR > 153285.

**Соединение таблиц** по полю ТВ\_ИД — используется Nested Loop Join.

Результат отправляется на вывод.

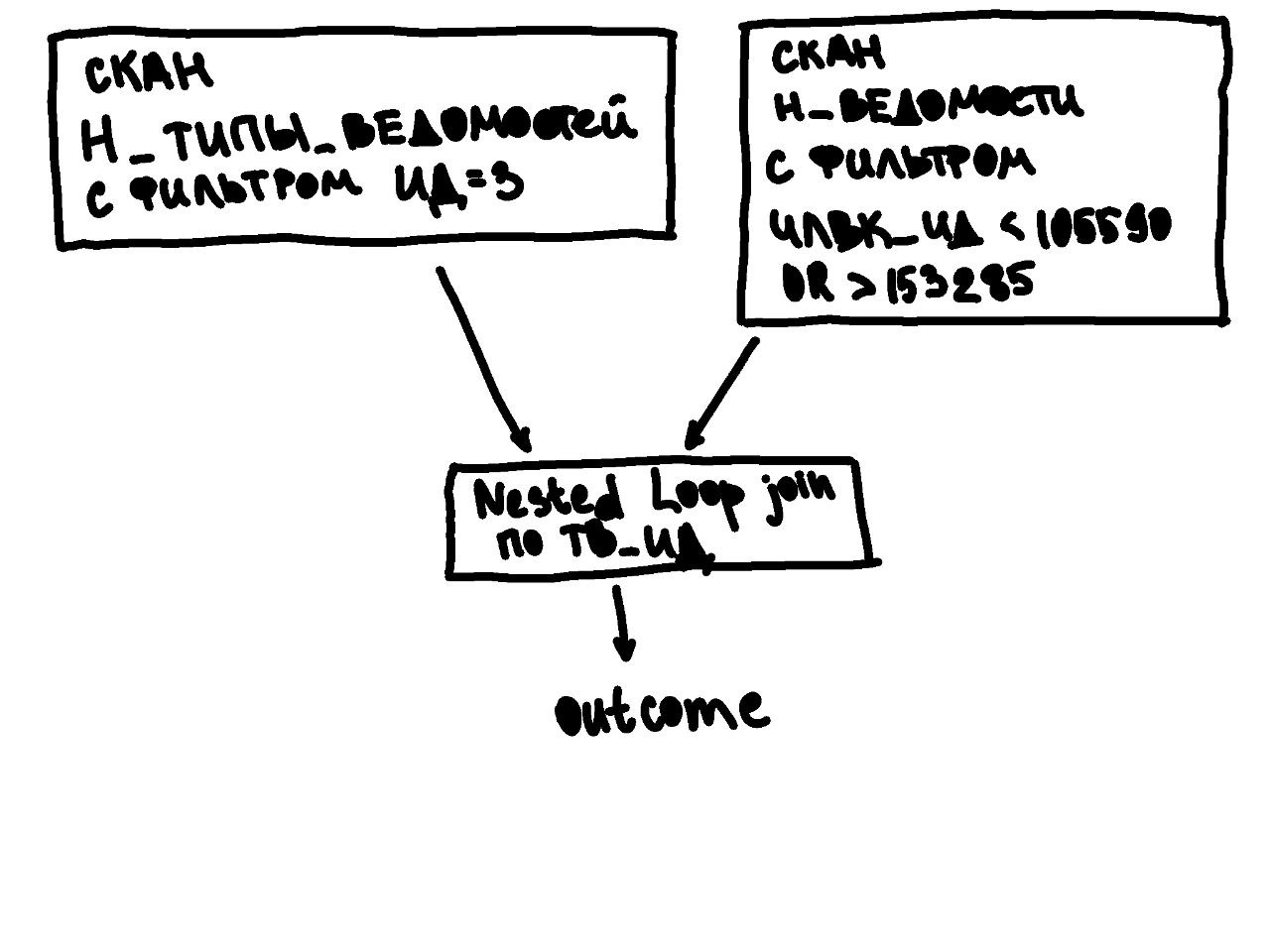


**План 2:**

**Фильтрация** Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = 3 выполняется сразу при чтении таблицы.

**Фильтрация** Н\_ВЕДОМОСТИ.ЧЛВК\_ИД < 105590 OR > 153285 также выполняется до соединения.

Таблицы соединяются по ТВ\_ИД через Nested Loop Join.



Оптимальным является второй план, так как в первом плане фильтрация выполняется **после соединения**, что ведёт к избыточной нагрузке на память и процессора, а во втором - соединяются уже **отфильтрованные подмножества**, что снижает нагрузку и ускоряет обработку.

При добавлении индексов планы выполнения запросов изменятся:

Вместо полного скана таблиц будет использоваться индексный скан

Nested Loops Join станет быстрее благодаря индексам.

**Запрос 2:**

**План 1:**

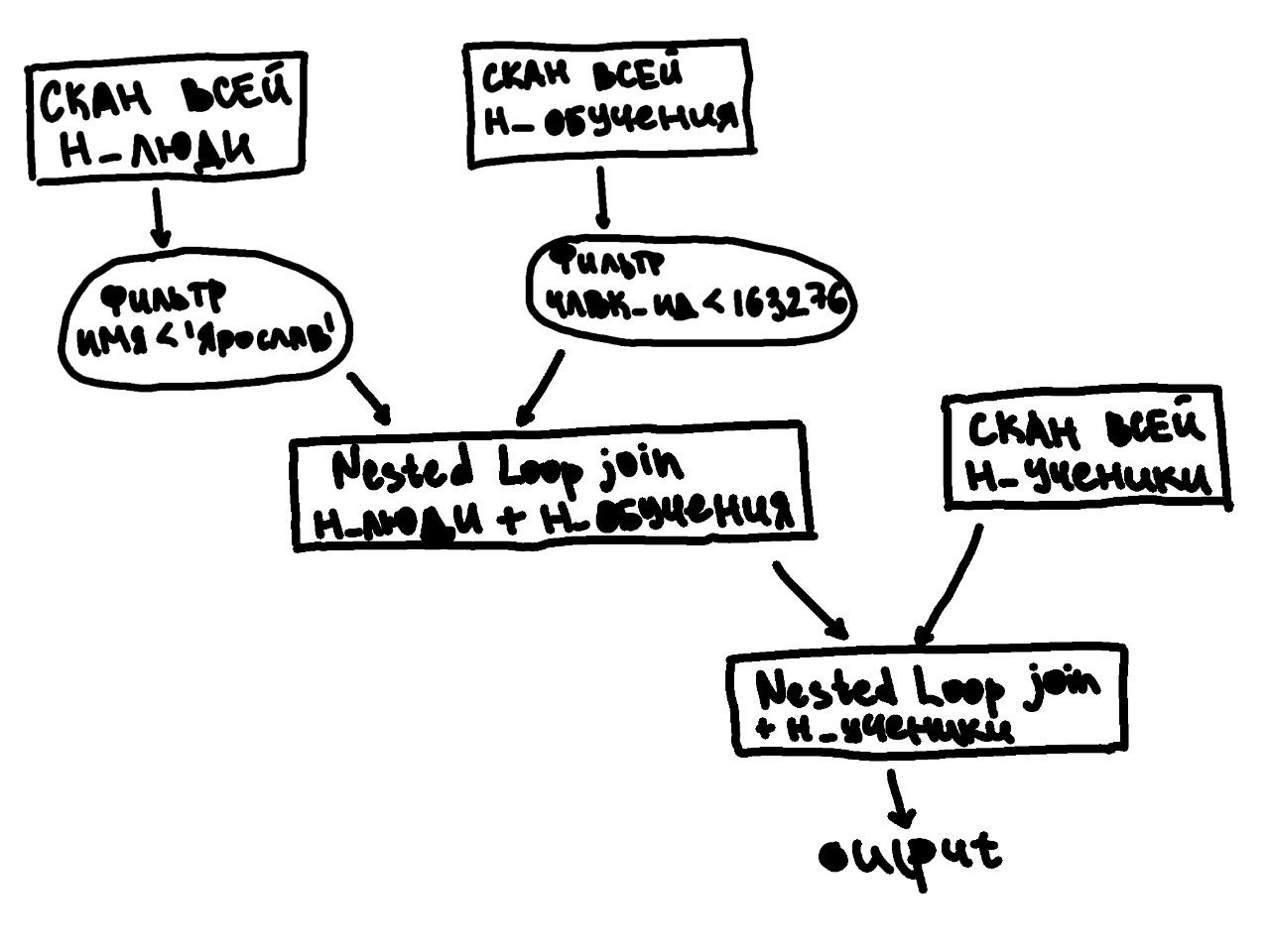
Полный скан таблицы Н\_ЛЮДИ с применением фильтра по условию "Н\_ЛЮДИ.ИМЯ < 'Ярослав'".

Полный скан таблицы Н\_ОБУЧЕНИЯ с применением фильтра по условию "Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276".

Соединение таблиц Н\_ЛЮДИ и Н\_ОБУЧЕНИЯ с использованием **Nested Loops Join** по атрибуту ЧЛВК\_ИД.

Полный скан таблицы Н\_УЧЕНИКИ.

Соединение результатов предыдущего этапа с таблицей Н\_УЧЕНИКИ с использованием **Nested Loops Join** по атрибуту ЧЛВК\_ИД.



**План 2:**

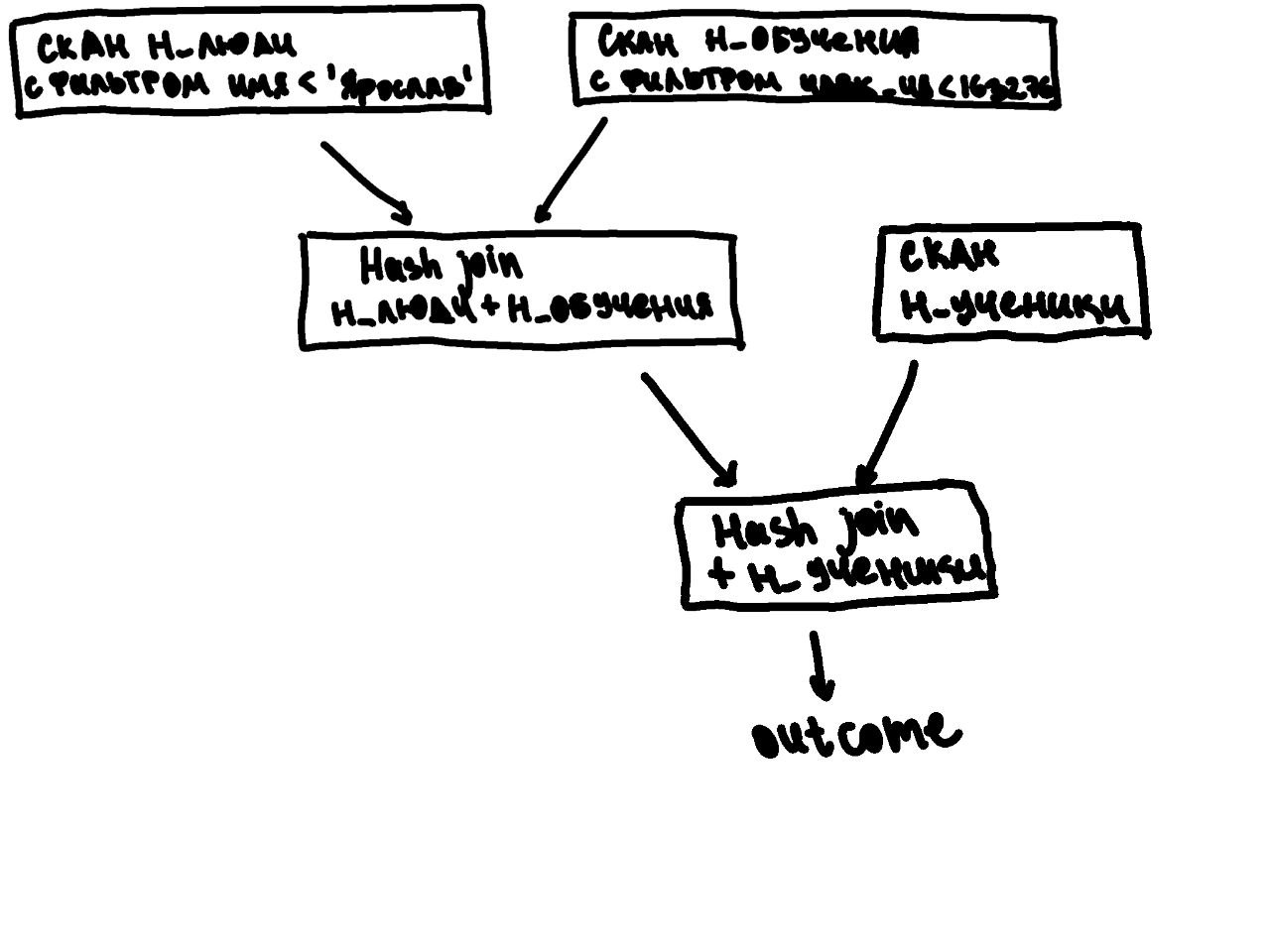
Полный скан таблицы Н\_ЛЮДИ с применением фильтра по условию "Н\_ЛЮДИ.ИМЯ < 'Ярослав'".

Полный скан таблицы Н\_ОБУЧЕНИЯ с применением фильтра по условию "Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД < 163276".

Соединение таблиц Н\_ЛЮДИ и Н\_ОБУЧЕНИЯ с использованием **Hash Join** по атрибуту ЧЛВК\_ИД.

Полный скан таблицы Н\_УЧЕНИКИ.

Соединение результатов предыдущего этапа с таблицей Н\_УЧЕНИКИ с использованием **Hash Join** по атрибуту ЧЛВК\_ИД.



Оптимальным является **План 1**, так как **Nested Loops Join предпочтительнее Hash Join** в случаях, когда обрабатываемые наборы данных **небольшие**, и заранее известно, что результаты фильтрации будут содержать малое количество строк.

При добавлении индексов планы выполнения запросов изменятся:

Вместо полного скана таблиц будет использоваться индексный скан

Nested Loops Join станет быстрее благодаря индексам.

**Вывод команды EXPLAIN ANALYZE**

**Запрос 1:**

Nested Loop (cost=225.17..1497.09 rows=434 width=422) (actual time=0.898..1.000 rows=187 loops=1)

-> Seq Scan on "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" (cost=0.00..1.04 rows=1 width=422) (actual time=0.015..0.015 rows=1 loops=1)

Filter: ("ИД" = 3)

Rows Removed by Filter: 2

-> Bitmap Heap Scan on "Н\_ВЕДОМОСТИ" (cost=225.17..1491.72 rows=434 width=8) (actual time=0.877..0.952 rows=187 loops=1)

Recheck Cond: ((("ЧЛВК\_ИД" < 105590) OR ("ЧЛВК\_ИД" > 153285)) AND ("ТВ\_ИД" = 3))

Heap Blocks: exact=26

-> BitmapAnd (cost=225.17..225.17 rows=434 width=0) (actual time=0.857..0.858 rows=0 loops=1)

-> BitmapOr (cost=109.93..109.93 rows=9216 width=0) (actual time=0.446..0.446 rows=0 loops=1)

-> Bitmap Index Scan on "ВЕД\_ЧЛВК\_FK\_IFK" (cost=0.00..4.30 rows=1 width=0) (actual time=0.002..0.002 rows=0 loops=1)

Index Cond: ("ЧЛВК\_ИД" < 105590)

-> Bitmap Index Scan on "ВЕД\_ЧЛВК\_FK\_IFK" (cost=0.00..105.42 rows=9216 width=0) (actual time=0.443..0.443 rows=9332 loops=1)

Index Cond: ("ЧЛВК\_ИД" > 153285)

-> Bitmap Index Scan on "ВЕД\_ТВ\_FK\_I" (cost=0.00..114.87 rows=10477 width=0) (actual time=0.373..0.373 rows=10075 loops=1)

Index Cond: ("ТВ\_ИД" = 3)

Planning Time: 0.748 ms

Execution Time: 1.108 ms

**Запрос 2:**

Hash Join (cost=423.09..1547.12 rows=22751 width=28) (actual time=7.503..18.582 rows=23544 loops=1)

Hash Cond: ("Н\_УЧЕНИКИ"."ЧЛВК\_ИД" = "Н\_ЛЮДИ"."ИД")

-> Seq Scan on "Н\_УЧЕНИКИ" (cost=0.00..809.11 rows=23311 width=8) (actual time=0.010..2.460 rows=23311 loops=1)

-> Hash (cost=360.65..360.65 rows=4995 width=28) (actual time=7.440..7.441 rows=5002 loops=1)

Buckets: 8192 Batches: 1 Memory Usage: 370kB

-> Hash Join (cost=227.71..360.65 rows=4995 width=28) (actual time=4.362..6.443 rows=5002 loops=1)

Hash Cond: ("Н\_ОБУЧЕНИЯ"."ЧЛВК\_ИД" = "Н\_ЛЮДИ"."ИД")

-> Seq Scan on "Н\_ОБУЧЕНИЯ" (cost=0.00..119.76 rows=5014 width=4) (actual time=0.007..0.908 rows=5019 loops=1)

Filter: ("ЧЛВК\_ИД" < 163276)

Rows Removed by Filter: 2

-> Hash (cost=163.97..163.97 rows=5099 width=24) (actual time=4.300..4.300 rows=5100 loops=1)

Buckets: 8192 Batches: 1 Memory Usage: 356kB

-> Seq Scan on "Н\_ЛЮДИ" (cost=0.00..163.97 rows=5099 width=24) (actual time=0.006..3.230 rows=5100 loops=1)

Filter: (("ИМЯ")::text < 'Ярослав'::text)

Rows Removed by Filter: 18

Planning Time: 1.608 ms

Execution Time: 19.711 ms

Вывод

В ходе выполнения лабораторной работы я освоила работу с реляционной алгеброй и научился строить планы выполнения запросов, а также их диаграммы. Я изучил различные виды индексов и узнала, как использовать их для оптимизации скорости выполнения запросов. Теперь я могу применять полученные знания для эффективной работы с базами данных и повышения производительности SQL-запросов.