Национальный исследовательский университет ИТМО

Факультет Программной Инженерии и Компьютерной техники

Лабораторная работа №4
Вариант №37646275

Выполнила:
Брель Мария Владимировна
Группа P3107
Преподаватели:
Байрамова Хумай
Николаев Владимир Вячеславович

Санкт-Петербург
2024

Оглавление

Задание:.....3
Запрос №1:.....4
Индексы:.....4
Результат EXPLAIN ANALYSE:.....5
Запрос №2:.....6
Индексы:.....6
Результат EXPLAIN ANALYSE:.....7
Вывод:.....8

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).

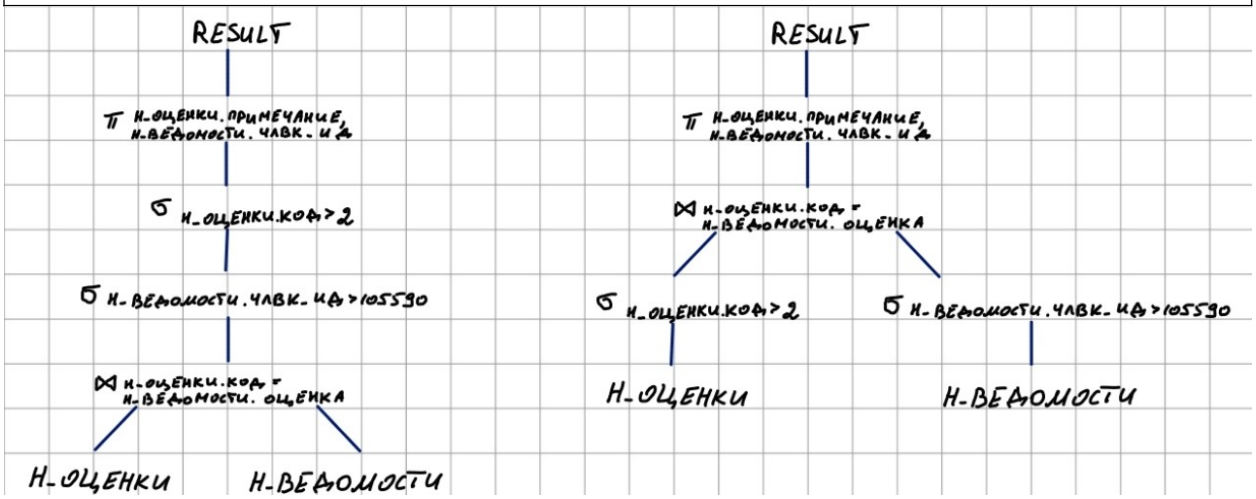
1. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:
Таблицы: H_ОЦЕНКИ, H_ВЕДОМОСТИ.
Вывести атрибуты: H_ОЦЕНКИ.ПРИМЕЧАНИЕ, H_ВЕДОМОСТИ.ЧЛВК_ИД.
Фильтры (AND):
a) H_ОЦЕНКИ.КОД > 2.
b) H_ВЕДОМОСТИ.ЧЛВК_ИД > 105590.
Вид соединения: INNER JOIN.

2. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:
Таблицы: H_ЛЮДИ, H_ОБУЧЕНИЯ, H_УЧЕНИКИ.
Вывести атрибуты: H_ЛЮДИ.ИМЯ, H_ОБУЧЕНИЯ.НЗК, H_УЧЕНИКИ.НАЧАЛО.
Фильтры: (AND)
a) H_ЛЮДИ.ОТЧЕСТВО = Георгиевич.
b) H_ОБУЧЕНИЯ.НЗК > 001000.
c) H_УЧЕНИКИ.НАЧАЛО = 1996-09-01.
Вид соединения: INNER JOIN.

Запрос №1:

Таблицы: H_ОЦЕНКИ, H_ВЕДОМОСТИ.
Вывести атрибуты: H_ОЦЕНКИ.ПРИМЕЧАНИЕ, H_ВЕДОМОСТИ.ЧЛВК_ИД.
Фильтры (AND):
a) H_ОЦЕНКИ.КОД > 2.
b) H_ВЕДОМОСТИ.ЧЛВК_ИД > 105590.
Вид соединения: INNER JOIN.

```
SELECT "H_ОЦЕНКИ"."ПРИМЕЧАНИЕ", "H_ВЕДОМОСТИ"."ЧЛВК_ИД" FROM
"H_ВЕДОМОСТИ"
JOIN "H_ОЦЕНКИ" ON "H_ОЦЕНКИ"."КОД" = "H_ВЕДОМОСТИ"."ОЦЕНКА"
WHERE "H_ОЦЕНКИ"."КОД" IN ('3','4','5') AND "H_ВЕДОМОСТИ"."ЧЛВК_ИД">105590;
```



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

Индексы:

```
CREATE INDEX "ИНДЕКС_ОЦЕНКИ_КОД" ON "H_ОЦЕНКИ" USING btree("КОД");
CREATE INDEX "ИНДЕКС_ВЕДОМОСТИ_ЧЛВК_ИД" ON "H_ВЕДОМОСТИ" USING
btree("ЧЛВК_ИД");
CREATE INDEX "ИНДЕКС_ВЕДОМОСТИ_ОЦЕНКА" ON "H_ВЕДОМОСТИ" USING
hash("ОЦЕНКА");
```

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 первых двух случаях используются операторы сравнения „>“ и „<“, так что эффективнее использовать btree. В третьем 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.

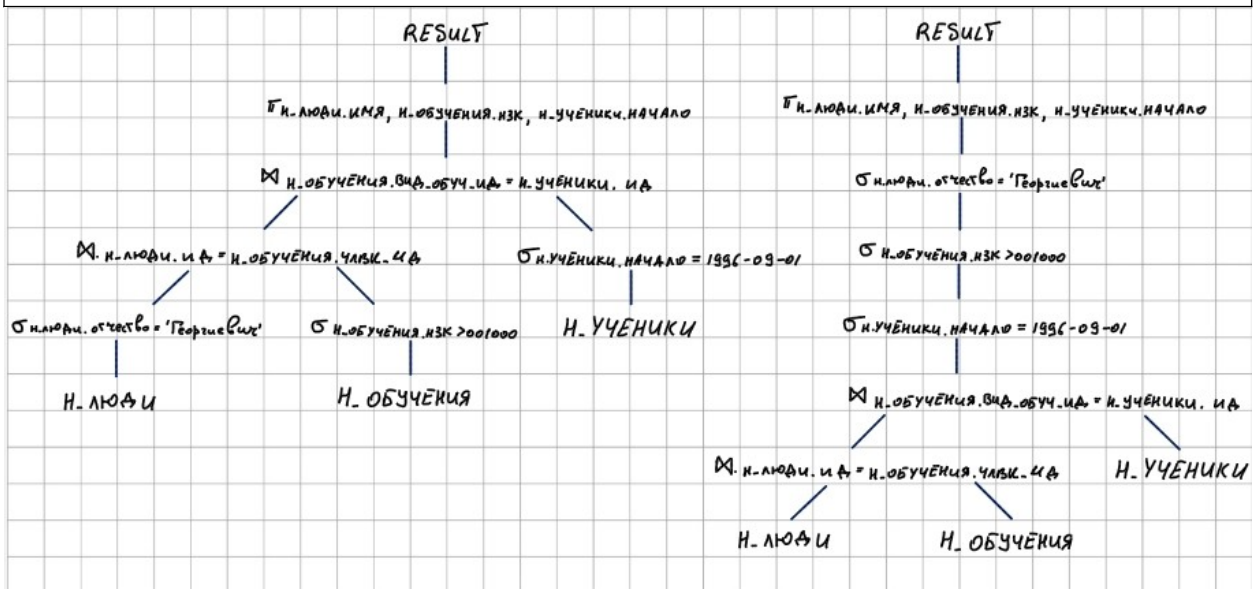
Результат EXPLAIN ANALYSE:

```
Hash Join  (cost=1.16..7706.27 rows=74125 width=26) (actual time=0.080..143.217 rows=107434 loops=1)
  Hash Cond: (("Н_ВЕДОМОСТИ"."ОЦЕНКА")::text = ("Н_ОЦЕНКИ"."КОД")::text)
  ->  Seq Scan on "Н_ВЕДОМОСТИ"  (cost=0.00..6846.50 rows=222375 width=10) (actual time=0.023..64.523 rows=222413 loops=1)
        Filter: ("ЧЛВК_ИД" > 105590)
        Rows Removed by Filter: 27
  ->  Hash  (cost=1.12..1.12 rows=3 width=27) (actual time=0.030..0.032 rows=3 loops=1)
        Buckets: 1024  Batches: 1  Memory Usage: 9kB
        ->  Seq Scan on "Н_ОЦЕНКИ"  (cost=0.00..1.12 rows=3 width=27) (actual time=0.009..0.011 rows=3 loops=1)
              Filter: (("КОД")::text = ANY ('{3,4,5}'::text[]))
              Rows Removed by Filter: 6
 Planning Time: 2.091 ms
 Execution Time: 152.891 ms
```

Запрос №2:

Таблицы: Н_ЛЮДИ, Н_ОБУЧЕНИЯ, Н_УЧЕНИКИ.
Вывести атрибуты: Н_ЛЮДИ.ИМЯ, Н_ОБУЧЕНИЯ.НЗК, Н_УЧЕНИКИ.НАЧАЛО.
Фильтры: (AND)
a) Н_ЛЮДИ.ОТЧЕСТВО = Георгиевич.
b) Н_ОБУЧЕНИЯ.НЗК > 001000.
c) Н_УЧЕНИКИ.НАЧАЛО = 1996-09-01.
Вид соединения: INNER JOIN.

```
SELECT "Н_ЛЮДИ"."ИМЯ", "Н_ОБУЧЕНИЯ"."НЗК", "Н_УЧЕНИКИ"."НАЧАЛО" FROM
"Н_ЛЮДИ"
JOIN "Н_ОБУЧЕНИЯ" ON "Н_ЛЮДИ"."ИД" = "Н_ОБУЧЕНИЯ"."ЧЛВК_ИД"
JOIN "Н_УЧЕНИКИ" ON "Н_ОБУЧЕНИЯ"."ВИД_ОБУЧ_ИД" = "Н_УЧЕНИКИ"."ИД"
WHERE "Н_ЛЮДИ"."ОТЧЕСТВО" = 'Георгиевич' AND "Н_ОБУЧЕНИЯ"."НЗК" > '001000'
AND "Н_УЧЕНИКИ"."НАЧАЛО" = '1996-09-01';
```



Оптимальным является план №1, 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
btree("ЧЛВК_ИД");
CREATE INDEX "ИНДЕКС_ОБУЧЕНИЯ_ВИД_ОБУЧЕНИЯ_ИД" ON "Н_ОБУЧЕНИЯ" USING
btree("ВИД_ОБУЧ_ИД");
CREATE INDEX "ИНДЕКС_ОБУЧЕНИЯ_НЗК" ON "Н_ОБУЧЕНИЯ" USING btree("НЗК");


CREATE INDEX "ИНДЕКС_УЧЕНИКИ_ИД" ON "Н_УЧЕНИКИ" USING btree("ИД");
CREATE INDEX "ИНДЕКС_УЧЕНИКИ_НАЧАЛО" ON "Н_УЧЕНИКИ" USING hash("НАЧАЛО");
```

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 первом и последнем случаях происходит прямое сравнение, так что эффективнее использовать хэш-индексы. В остальных случаях используются операторы сравнения „>“ и „<“, так что эффективнее использовать btree.

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=0.85..257.57 rows=1 width=27) (actual time=0.161..13.270 rows=21 loops=1)
   ->  Merge Join  (cost=0.57..199.22 rows=75 width=18) (actual time=0.045..3.922 rows=4387 loops=1)
         Merge Cond: ("Н_ОБУЧЕНИЯ"."ВИД_ОБУЧ_ИД" = "Н_УЧЕНИКИ"."ИД")
         ->  Index Scan using "ОБУЧ_ВО_FK_I" on "Н_ОБУЧЕНИЯ"  (cost=0.28..180.40 rows=4387 width=14) (actual time=0.032..2.812 rows=4387 loops=1)
               Filter: (("НЗК")::text > '001000'::text)
               Rows Removed by Filter: 634
         ->  Index Scan using "УЧЕН_РК" on "Н_УЧЕНИКИ"  (cost=0.29..2722.68 rows=400 width=12) (actual time=0.009..0.011 rows=2 loops=1)
               Filter: ("НАЧАЛО" = '1996-09-01 00:00:00'::timestamp without time zone)
   ->  Index Scan using "ЧЛВК_РК" on "Н_ЛЮДИ"  (cost=0.28..0.78 rows=1 width=17) (actual time=0.002..0.002 rows=0 loops=4387)
         Index Cond: ("ИД" = "Н_ОБУЧЕНИЯ"."ЧЛВК_ИД")
         Filter: (("ОТЧЕСТВО")::text = 'Георгиевич'::text)
         Rows Removed by Filter: 1
 Planning Time: 1.757 ms
 Execution Time: 13.347 ms
```

Вывод: в ходе выполнения данной лабораторной работы я научилась оптимизировать запросы, продумывать более выгодные планы выполнения запросов и использовать для этого разные виды индексов.