**Федеральное государственное автономное образовательное учреждение высшего образования**

**Национальный исследовательский университет ИТМО**

Факультет программной инженерии и компьютерной техники

Направление подготовки: 09.03.04. Системное и прикладное программное обеспечение

Дисциплина «Базы данных»

**Отчет**

**По лабораторной работе №4**

**Вариант №21355**

Студент

Бушмелев Константин Алексеевич,   
группа P3118

Преподаватель

Николаев Владимир Вячеславович

г. Санкт-Петербург, 2022 г.

Оглавление

[Оглавление 2](#_Toc134689587)

[Задание 2](#_Toc134689588)

[SQL запросы 3](#_Toc134689589)

[Индексирование 3](#_Toc134689590)

[Планы выполнения запросов 6](#_Toc134689591)

[Вывод команды EXPLAIN ANALYZE 13](#_Toc134689592)

[Заключение 14](#_Toc134689593)

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.НАИМЕНОВАНИЕ < Экзаменационный лист.  
   b) Н\_ВЕДОМОСТИ.ДАТА = 2022-06-08.  
   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.ИМЯ, Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД, Н\_УЧЕНИКИ.ИД.  
   Фильтры: (AND)  
   a) Н\_ЛЮДИ.ИД = 152862.  
   b) Н\_ОБУЧЕНИЯ.НЗК = 933232.  
   c) Н\_УЧЕНИКИ.НАЧАЛО > 2011-11-21.  
   Вид соединения: INNER JOIN.

SQL запросы

SELECT Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н\_ВЕДОМОСТИ.ЧЛВК\_ИД

FROM Н\_ТИПЫ\_ВЕДОМОСТЕЙ

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

WHERE Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ < 'Экзаменационный лист'

AND Н\_ВЕДОМОСТИ.ДАТА = DATE('2022-06-08');

SELECT Н\_ЛЮДИ.ИМЯ, Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД, Н\_УЧЕНИКИ.ИД

FROM Н\_ЛЮДИ

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

INNER JOIN Н\_УЧЕНИКИ ON Н\_ОБУЧЕНИЯ.ЧЛВК\_ИД = Н\_УЧЕНИКИ.ИД

WHERE Н\_ЛЮДИ.ИД = 152862

AND Н\_ОБУЧЕНИЯ.НЗК = '933232'

AND Н\_УЧЕНИКИ.НАЧАЛО > DATE('2011-11-21');

Индексирование

1. CREATE INDEX НАИМЕНОВАНИЕ\_ИНДЕКС ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ

USING btree(НАИМЕНОВАНИЕ);

Эффективно использовать данный тип индекса, так как:

1. Операция направлена на запрос данных;
2. Запрос выдает несколько строк, а не одну;
3. Данный атрибут используется в фильтрации с помощью WHERE;
4. Запрос использует оператор «<»;
5. Данный атрибут редко будет изменяться, а значит, индекс не придется менять часто;

CREATE INDEX ДАТА\_ИНДЕКС ON Н\_ВЕДОМОСТИ

USING hash (ДАТА);

Эффективно использовать данный тип индекса, так как:

1. Операция направлена на запрос данных;
2. Запрос выдает несколько строк, а не одну;
3. Данный атрибут используются в фильтрации с помощью WHERE;
4. Запрос использует операторы «=»;
5. Данный атрибут редко будет изменяться, а значит, индекс не придется менять часто;
6. Большой объем данных не выбирается, так как ведомости получаются только за один день.

CREATE INDEX ТВ\_ИД\_ИНДЕКС

ON Н\_ВЕДОМОСТИ

USING hash (ТВ\_ИД);

Эффективно использовать данный тип индекса, так как:

1. В операции участвует соединение таблиц;
2. Данные столбцы используются в JOIN;
3. Все значения атрибута уникальны, так как являются внешним ключом, а значит, используют единственное значение.
4. Используется прямое сравнение.

CREATE INDEX ИД\_ИНДЕКС

ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ

USING hash (ИД);

Эффективно использовать данный тип индекса, так как:

1. В операции участвует соединение таблиц;
2. Данные столбцы используются в JOIN;
3. Все значения атрибута уникальны, так как являются главным ключом, а значит, используют единственное значение;
4. Используется прямое сравнение.
5. CREATE INDEX НАЧАЛО\_ИНДЕКС

ON Н\_УЧЕНИКИ (НАЧАЛО);

Эффективно использовать данный тип индекса, так как:

1. Операция направлена на запрос данных;
2. Данный атрибут используются в фильтрации с помощью WHERE;
3. Запрос использует оператор «>»;
4. Данный атрибут редко будет изменяться, а значит, индекс не придется менять часто;

CREATE INDEX ИД\_ИНДЕКС ON Н\_ЛЮДИ

USING hash (ИД);

Эффективно использовать данный тип индекса, так как:

1. Операция направлена на запрос данных;
2. Данные столбцы используются в фильтрации с помощью WHERE;
3. Данные атрибуты редко будут изменяться, а значит, индекс не придется менять часто;
4. Все значения атрибута уникальны, так как являются главным ключом, а значит, используют единственное значение;
5. Используется прямое сравнение.

CREATE INDEX ЧЛВК\_ИД\_ИНДЕКС

ON Н\_ОБУЧЕНИЯ

USING hash (ЧЛВК\_ИД);

Эффективно использовать данный тип индекса, так как:

1. В операции участвует соединение таблиц;
2. Данные столбцы используются в JOIN;
3. Все значения атрибута уникальны, так как являются внешним ключом, а значит, используют единственное значение;
4. Используется прямое сравнение.

CREATE INDEX НЗК\_ИНДЕКС ON Н\_ОБУЧЕНИЯ

USING hash (НЗК);

Эффективно использовать данный тип индекса, так как:

1. Операция направлена на запрос данных;
2. Данные столбцы используются в фильтрации с помощью WHERE;
3. Данные атрибуты редко будут изменяться, а значит, индекс не придется менять часто;
4. Все значения атрибута уникальны, а значит, используют единственное значение;
5. Используется прямое сравнение.

Планы выполнения запросов

Первый запрос

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

1. Базовый план исполнения

Изображение выглядит как текст, снимок экрана, Шрифт, линия

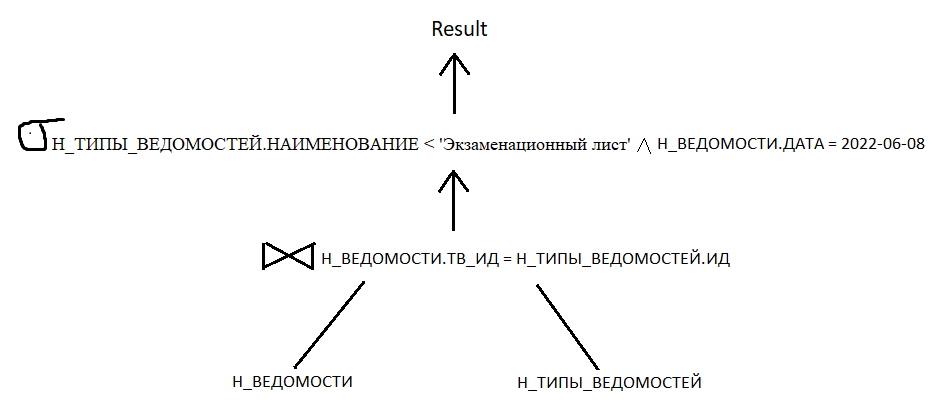
Автоматически созданное описание

2. Коммутативность

Изображение выглядит как линия, текст, диаграмма, График

Автоматически созданное описание

3. Закон №4 - перемена мест соединения и фильтрации



4. Объединение фильтров

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

5. Проекция перед соединением и фильтрами

Изображение выглядит как текст, линия, снимок экрана, диаграмма

Автоматически созданное описание

6. Проекция в конце

Изображение выглядит как линия, текст, диаграмма, График

Автоматически созданное описание

7. Оптимальный план

Этот план является оптимальным, так как:

1. Проекция сделана раньше соединения;
2. Выборка сделана раньше соединения;
3. Минимизирован размер промежуточных данных.

Второй запрос

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

8. Базовый план исполнения

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

9. Коммутативность

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

10. Ассоциативность

Изображение выглядит как текст, диаграмма, снимок экрана, линия

Автоматически созданное описание

11. Композиция

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

12. Фильтрация перед соединением таблиц

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Автоматически созданное описание

13. Проекция перед фильтрацией и соединением таблиц

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

14.Проекция в конце

Оптимальным является план №13, так как:

1. Проекция сделана раньше соединения;
2. Выборка сделана раньше соединения;
3. План левосторонний;
4. Минимизирован размер промежуточных данных.

Использование индексов положительно повлияет на план выполнения запроса, так как выборка и соединение таблиц будет происходить быстрее.

На схеме плана это не отобразится, но из-за конвейерной обработки ставить индекс имеет смысл только на самую нижнюю выборку в каждой таблице.

Вывод команды EXPLAIN ANALYZE

Первый запрос

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Второй запрос

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Заключение

В ходе лабораторной работы я узнал о методах оптимизации SQL-запроса, научился писать индексы и хэш-индексы, составлять всевозможные планы исполнения запросов и выбирать оптимальный из них.