Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Санкт-Петербургский национальный исследовательский университет

ИТМО»

**факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА № 4**

по дисциплине

‘Базы данных’

Вариант №1721

*Выполнила:*

Студентка группы P3130

Вернер Кристина

Витальевна

*Преподаватель:*

Перцев Тимофей

Сергеевич



Санкт-Петербург, 2025

1. **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:  
   Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ.  
   Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н\_ВЕДОМОСТИ.ДАТА.  
   Фильтры (AND):   
   a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1.  
   b) Н\_ВЕДОМОСТИ.ДАТА < 2010-06-18.  
   Вид соединения: INNER JOIN.
2. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
   Таблицы: Н\_ЛЮДИ, Н\_ВЕДОМОСТИ, Н\_СЕССИЯ.  
   Вывести атрибуты: Н\_ЛЮДИ.ОТЧЕСТВО, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА.  
   Фильтры (AND):   
   a) Н\_ЛЮДИ.ФАМИЛИЯ = Иванов.  
   b) Н\_ВЕДОМОСТИ.ДАТА > 1998-01-05.  
   c) Н\_СЕССИЯ.УЧГОД < 2011/2012.  
   Вид соединения: INNER JOIN.
3. **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.ДАТА. --Фильтры (AND):  --a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1. --b) Н\_ВЕДОМОСТИ.ДАТА < 2010-06-18. --Вид соединения: INNER JOIN.  SELECT  Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ,  Н\_ВЕДОМОСТИ.ДАТА  FROM  Н\_ТИПЫ\_ВЕДОМОСТЕЙ  INNER JOIN  Н\_ВЕДОМОСТИ ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ТВ\_ИД  WHERE  Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1 AND  Н\_ВЕДОМОСТИ.ДАТА < '2010-06-18';  ---------------------------------------------------------  --Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным --условиям: --Таблицы: Н\_ЛЮДИ, Н\_ВЕДОМОСТИ, Н\_СЕССИЯ. --Вывести атрибуты: Н\_ЛЮДИ.ОТЧЕСТВО, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА. --Фильтры (AND):  --a) Н\_ЛЮДИ.ФАМИЛИЯ = Иванов. --b) Н\_ВЕДОМОСТИ.ДАТА > 1998-01-05. --c) Н\_СЕССИЯ.УЧГОД < 2011/2012. --Вид соединения: INNER JOIN.  SELECT Н\_ЛЮДИ.ОТЧЕСТВО, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА  FROM Н\_ЛЮДИ  INNER JOIN Н\_ВЕДОМОСТИ ON Н\_ЛЮДИ.ИД = Н\_ВЕДОМОСТИ.ЧЛВК\_ИД  INNER JOIN Н\_СЕССИЯ ON Н\_ВЕДОМОСТИ.СЭС\_ИД = Н\_СЕССИЯ.СЭС\_ИД  WHERE Н\_ЛЮДИ.ФАМИЛИЯ = 'Иванов' AND Н\_ВЕДОМОСТИ.ДАТА = '1998-01-05' AND Н\_СЕССИЯ.УЧГОД < '2011/2012'; |

1. **Уменьшение выполнения времени 1 запроса**

|  |
| --- |
| Индексы, добавление которых уменьшит время выполнения запроса:  a) На таблице Н\_ТИПЫ\_ВЕДОМОСТЕЙ:   * Индекс на атрибуте ИД (B-tree) Это ускорит фильтрацию строк с условием "Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1".   b) На таблице Н\_ВЕДОМОСТИ:   * Индекс на атрибуте ДАТА (B-tree) Это ускорит фильтрацию строк с условием "Н\_ВЕДОМОСТИ.ДАТА < '2010-06-18'". * Индекс на атрибуте ТВ\_ИД (B-tree) Это ускорит соединение таблиц по атрибуту ТВ\_ИД.   Добавление индексов на указанные столбцы позволит ускорить поиск нужных записей по фильтрам в запросе.  Возможные планы выполнения запросов без индексов:  **План 1:**   * Полный скан таблицы Н\_ТИПЫ\_ВЕДОМОСТЕЙ. * Фильтрация строк по условию Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1. * Для каждой строки:   + Полный скан таблицы Н\_ВЕДОМОСТИ с фильтрацией Н\_ВЕДОМОСТИ.ДАТА < '2010-06-18'.   + Соединение с использованием **Nested Loops Join** по условию Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ТВ\_ИД.   Недостаток: Н\_ВЕДОМОСТИ просматривается многократно — по числу строк в Н\_ТИПЫ\_ВЕДОМОСТЕЙ.  **План 2 :**   * Полный скан таблицы Н\_ВЕДОМОСТИ с фильтрацией по условию Н\_ВЕДОМОСТИ.ДАТА < '2010-06-18'. * Полный скан таблицы Н\_ТИПЫ\_ВЕДОМОСТЕЙ с фильтрацией по условию Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1. * Соединение двух отфильтрованных наборов данных с использованием **Nested Loops Join** по полю ТВ\_ИД.   Преимущество: отфильтрованные данные до соединения → меньше строк обрабатывается на этапе JOIN.  Оптимальный план:   * План 2, потому что фильтрация данных выполняется до соединения таблиц, что уменьшает количество строк для обработки.   При добавлении индексов:  CREATE INDEX idx\_тип\_ид ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ (ИД);  CREATE INDEX idx\_ведомости\_дата ON Н\_ВЕДОМОСТИ (ДАТА);  CREATE INDEX idx\_ведомости\_твид ON Н\_ВЕДОМОСТИ (ТВ\_ИД);  планы выполнения запросов изменятся:   * Вместо полного скана таблиц будет использоваться индексный скан. * Nested Loops Join станет быстрее благодаря индексу на атрибуте ТВ\_ИД. |

1. **План выполнения 1 запроса**

|  |
| --- |
| EXPLAIN ANALYZE  SELECT  Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ,  Н\_ВЕДОМОСТИ.ДАТА  FROM  Н\_ТИПЫ\_ВЕДОМОСТЕЙ  INNER JOIN  Н\_ВЕДОМОСТИ ON Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД = Н\_ВЕДОМОСТИ.ТВ\_ИД  WHERE  Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1 AND  Н\_ВЕДОМОСТИ.ДАТА < '2010-06-18';  QUERY PLAN  ---------------------------------------------------------------------------------------------------------------------  Nested Loop (cost=828.76..6700.66 rows=65465 width=426) (actual time=0.014..0.015 rows=0 loops=1)  -> Seq Scan on "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" (cost=0.00..1.04 rows=1 width=422) (actual time=0.013..0.014 rows=0 loops=1)  Filter: ("ИД" < 1)  Rows Removed by Filter: 3  -> Bitmap Heap Scan on "Н\_ВЕДОМОСТИ" (cost=828.76..6044.97 rows=65465 width=12) (never executed)  Recheck Cond: ("Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."ИД" = "ТВ\_ИД")  Filter: ("ДАТА" < '2010-06-18 00:00:00'::timestamp without time zone)  -> Bitmap Index Scan on "ВЕД\_ТВ\_FK\_I" (cost=0.00..812.40 rows=74147 width=0) (never executed)  Index Cond: ("ТВ\_ИД" = "Н\_ТИПЫ\_ВЕДОМОСТЕЙ"."ИД")  Planning Time: 1.174 ms  Execution Time: 0.105 ms  (11 строк) |

1. **Уменьшение выполнения времени 2 запроса**

|  |
| --- |
| Индексы, добавление которых уменьшит время выполнения запроса:  a) На таблице Н\_ЛЮДИ:   * Индекс на атрибуте ФАМИЛИЯ (B-tree). Это ускорит фильтрацию строк с условием "Н\_ЛЮДИ.ФАМИЛИЯ = 'Иванов'".   b) На таблице Н\_ВЕДОМОСТИ:   * Индекс на атрибуте ДАТА (B-tree). Это ускорит фильтрацию строк с условием "Н\_ВЕДОМОСТИ.ДАТА = '1998-01-05'".   c) На таблице Н\_СЕССИЯ:   * Индекс на атрибуте УЧГОД (B-tree). Это ускорит фильтрацию строк с условием "Н\_СЕССИЯ. УЧГОД= '2011/2012'".   Добавление индексов на указанные столбцы позволит ускорить поиск нужных записей по фильтрам в запросе.  Возможные планы выполнения запросов без индексов:  План 1:   1. Полный скан таблицы Н\_ЛЮДИ  Применяется фильтр: ФАМИЛИЯ = 'Иванов' 2. Для каждой найденной строки:   Полный скан таблицы Н\_ВЕДОМОСТИ  Проверяется:  ЧЛВК\_ИД = ИД (условие JOIN)  ДАТА > '1998-01-05' (фильтр)   1. Для каждой найденной строки ведомости:   Полный скан таблицы Н\_СЕССИЯ  Проверяется:  СЭС\_ИД = СЭС\_ИД (JOIN)  УЧГОД < '2011/2012' (фильтр)  Недостаток: три вложенных цикла (Nested Loops Join) и множество полных сканов — медленно.  План 2 (более оптимальный):   1. Полный скан таблицы Н\_ЛЮДИ с фильтрацией ФАМИЛИЯ = 'Иванов' 2. Полный скан таблицы Н\_ВЕДОМОСТИ с фильтрацией ДАТА > '1998-01-05' 3. Полный скан таблицы Н\_СЕССИЯ с фильтрацией УЧГОД < '2011/2012' 4. Выполняются JOIN-ы:   Н\_ЛЮДИ.ИД = Н\_ВЕДОМОСТИ.ЧЛВК\_ИД  Н\_ВЕДОМОСТИ.СЭС\_ИД = Н\_СЕССИЯ.СЭС\_ИД  Преимущество: фильтрация выполняется до соединения, количество строк меньше, JOIN быстрее.  Оптимальный план 2:   * начала отбрасываем ненужные строки. * JOIN работает уже с меньшими таблицами → быстрее.   При добавлении индексов:  CREATE INDEX idx\_люди\_фамилия ON Н\_ЛЮДИ (ФАМИЛИЯ); -- для фильтрации 'Иванов'  CREATE INDEX idx\_ведомости\_челид ON Н\_ВЕДОМОСТИ (ЧЛВК\_ИД); -- для соединения  CREATE INDEX idx\_ведомости\_дата ON Н\_ВЕДОМОСТИ (ДАТА); -- для фильтра  CREATE INDEX idx\_ведомости\_сэсид ON Н\_ВЕДОМОСТИ (СЭС\_ИД); -- для соединения  CREATE INDEX idx\_сессия\_сэсид ON Н\_СЕССИЯ (СЭС\_ИД); -- для соединения  CREATE INDEX idx\_сессия\_учгод ON Н\_СЕССИЯ (УЧГОД); -- для фильтра  планы выполнения запросов изменятся:   * Используются Index Scan вместо Full Scan * Используется Index Nested Loops Join: * по ЧЛВК\_ИД и СЭС\_ИД — соединение ускоряется * Все фильтры (ФАМИЛИЯ, ДАТА, УЧГОД) — обрабатываются индексами * Итого: ускорение в разы, особенно при больших объёмах данных |

1. **План выполнения 2 запроса**

|  |
| --- |
| EXPLAIN ANALYZE  SELECT Н\_ЛЮДИ.ОТЧЕСТВО, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА  FROM Н\_ЛЮДИ  INNER JOIN Н\_ВЕДОМОСТИ ON Н\_ЛЮДИ.ИД = Н\_ВЕДОМОСТИ.ЧЛВК\_ИД  INNER JOIN Н\_СЕССИЯ ON Н\_ВЕДОМОСТИ.СЭС\_ИД = Н\_СЕССИЯ.СЭС\_ИД  WHERE Н\_ЛЮДИ.ФАМИЛИЯ = 'Иванов'  AND Н\_ВЕДОМОСТИ.ДАТА > '1998-01-05'  AND Н\_СЕССИЯ.УЧГОД < '2011/2012';  2nd QUERY PLAN  ------------------------------------------------------------------------------------------------------------------  Gather (cost=1248.43..6868.51 rows=1082 width=36) (actual time=3.276..27.466 rows=1533 loops=1)  Workers Planned: 2  Workers Launched: 2  -> Hash Join (cost=248.43..5760.31 rows=451 width=36) (actual time=3.881..21.196 rows=511 loops=3)  Hash Cond: ("Н\_ВЕДОМОСТИ"."СЭС\_ИД" = "Н\_СЕССИЯ"."СЭС\_ИД")  -> Hash Join (cost=86.24..5586.65 rows=796 width=32) (actual time=0.243..18.463 rows=682 loops=3)  Hash Cond: ("Н\_ВЕДОМОСТИ"."ЧЛВК\_ИД" = "Н\_ЛЮДИ"."ИД")  -> Parallel Seq Scan on "Н\_ВЕДОМОСТИ" (cost=0.00..5262.54 rows=90540 width=16) (actual time=0.009..12.126 rows=72416 loops=3)  Filter: ("ДАТА" > '1998-01-05 00:00:00'::timestamp without time zone)  Rows Removed by Filter: 1731  -> Hash (cost=85.68..85.68 rows=45 width=24) (actual time=0.155..0.156 rows=45 loops=3)  Buckets: 1024 Batches: 1 Memory Usage: 11kB  -> Bitmap Heap Scan on "Н\_ЛЮДИ" (cost=4.63..85.68 rows=45 width=24) (actual time=0.091..0.139 rows=45 loops=3)  Recheck Cond: (("ФАМИЛИЯ")::text = 'Иванов'::text)  Heap Blocks: exact=35  -> Bitmap Index Scan on "ФАМ\_ЛЮД" (cost=0.00..4.62 rows=45 width=0) (actual time=0.079..0.080 rows=45 loops=3)  Index Cond: (("ФАМИЛИЯ")::text = 'Иванов'::text)  -> Hash (cost=117.90..117.90 rows=3543 width=12) (actual time=2.500..2.501 rows=3543 loops=3)  Buckets: 4096 Batches: 1 Memory Usage: 184kB  -> Seq Scan on "Н\_СЕССИЯ" (cost=0.00..117.90 rows=3543 width=12) (actual time=0.015..1.824 rows=3543 loops=3)  Filter: (("УЧГОД")::text < '2011/2012'::text)  Rows Removed by Filter: 209  Planning Time: 1.159 ms  Execution Time: 27.626 ms  (24 строки) |

1. **Вывод**

В ходе выполнения лабораторной работы я освоила работу с реляционной алгеброй и научилась строить планы выполнения запросов, а также их диаграммы. Я изучила 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