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

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4

Вариант №228

Выполнил: Пивоваров Р. Н.

Группа: P3131

Проверил:

Коновалов А. А.

Г. Санкт-Петербург, 2025 г.

Оглавление

Задание **2**

Запросы **3**

Индексы для Запросов **4**

Планы выполнения для запросов. **4**

вывод команды EXPLAIN ANALYZE **8**

Заключение **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:
   Н_ТИПЫ_ВЕДОМОСТЕЙ, Н_ВЕДОМОСТИ.
   Вывести атрибуты: Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н_ВЕДОМОСТИ.ИД.
   Фильтры (AND):
   a) Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД < 3.
   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.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.УЧГОД.
   Фильтры (AND):
   a) Н_ЛЮДИ.ОТЧЕСТВО > Сергеевич.
   b) Н_ВЕДОМОСТИ.ДАТА < 1998-01-05.
   c) Н_СЕССИЯ.ИД > 14369.
   Вид соединения: LEFT JOIN.

Запросы

```sql
-- 1
SELECT Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н_ВЕДОМОСТИ.ИД FROM Н_ТИПЫ_ВЕДОМОСТЕЙ
INNER JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ТВ_ИД = Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД
WHERE Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД < 3 AND Н_ВЕДОМОСТИ.ДАТА > '2010-06-18';
```

Рисунок 1: запрос 1

```sql
-- 2
SELECT Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.УЧГОД FROM  Н_ЛЮДИ
LEFT JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
LEFT JOIN Н_СЕССИЯ ON Н_СЕССИЯ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
WHERE Н_ЛЮДИ.ОТЧЕСТВО > 'Сергеевич' AND Н_ВЕДОМОСТИ.ДАТА < '1998-01-05' AND Н_СЕССИЯ.ИД > 14369;
```

Рисунок 2: запрос 2

Запрос 2 не работает, потому что в задание указано некорректное условие. Нас просят вывести строки в который дата ведомости < 1998-01-05, но это минимальное значение даты которое есть в таблице Н_ВЕДОМОСТИ, т.е. строк соответствующих условию нет.

```
ucheb=> SELECT MIN(H_ВЕДОМОСТИ.ДАТА) FROM H_ВЕДОМОСТИ;
         min
---------------------
 1998-01-05 00:00:00
(1 строка)
```

Рисунок 3: запрос на вывод мин. Даты ведомости.

Индексы для Запросов

Запрос 1:

- Добавление индекса для ТИПЫ_ВЕДОМОСТЕЙ.ИД не имеет смысла т.к. таблица содержит только 3 кортежа(строки).
- А добавления индекса для Н_ВЕДОМОСТИ.ДАТА поможет ускорит выполнения запроса. Наиболее подходящим будет B-tree index т.е. запрос работает с оператором >. Добавление этого индекса поможет быстро найти минимальное значение и очень быстро найти все последующие значения.
- Из-за небольших размеров таблицы ТИПЫ_ВЕДОМОСТЕЙ.ИД индексация Н_ВЕДОМОСТИ.ИД не требуется.

Запрос 2:

- В этом запросе будет полезно для всех столбцов, задействованных в where, а именно: Н_ЛЮДИ.ОТЧЕСТВО, Н_ВЕДОМОСТИ.ДАТА, Н_СЕССИЯ.ИД. для всех этих столбцов идеально подойдет B-tree index, потому что все они подвержены операторам порядка(<, >, = и т.д.).
- Т.к. в Н_ВЕДОМОСТИ и Н_СЕССИЯ по несколько записей на 1 человека, то добавления индекса для этих столбцов поможет ускорит выполнения запроса. Используем для этого обычный index т.к. нам нужна просто группировка значений.

Планы выполнения для запросов.

Запрос 1:

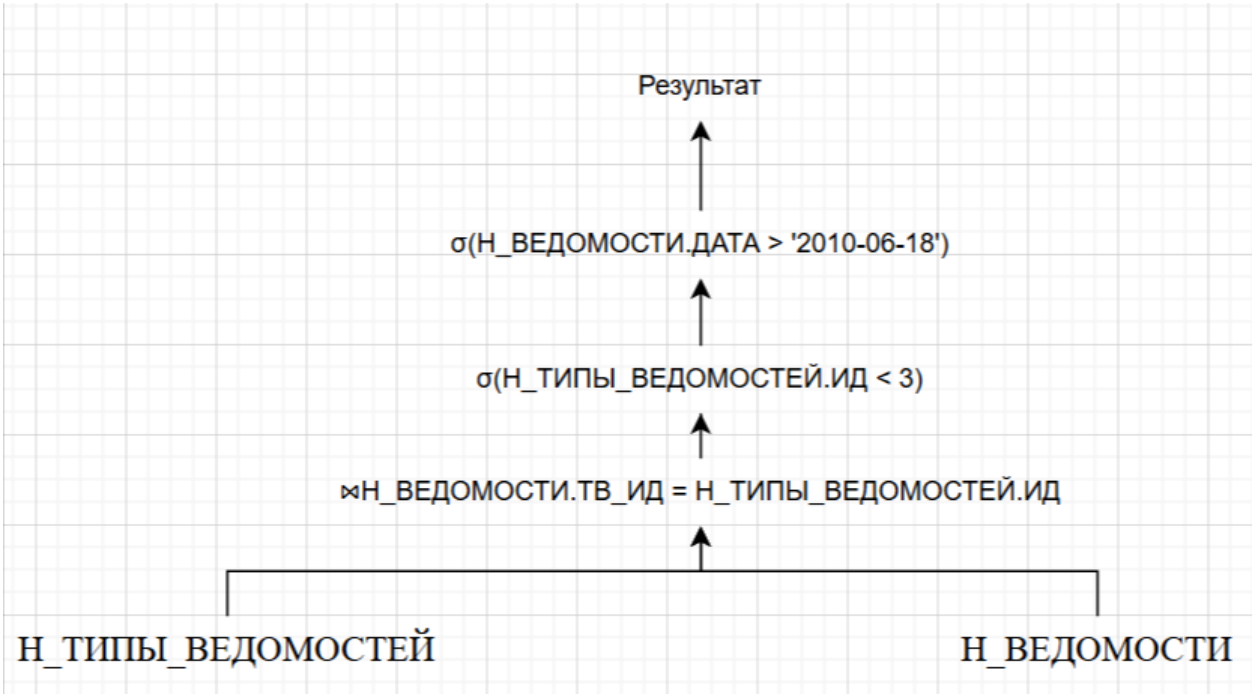

Рисунок 4 наивный план выполнения для запроса 1

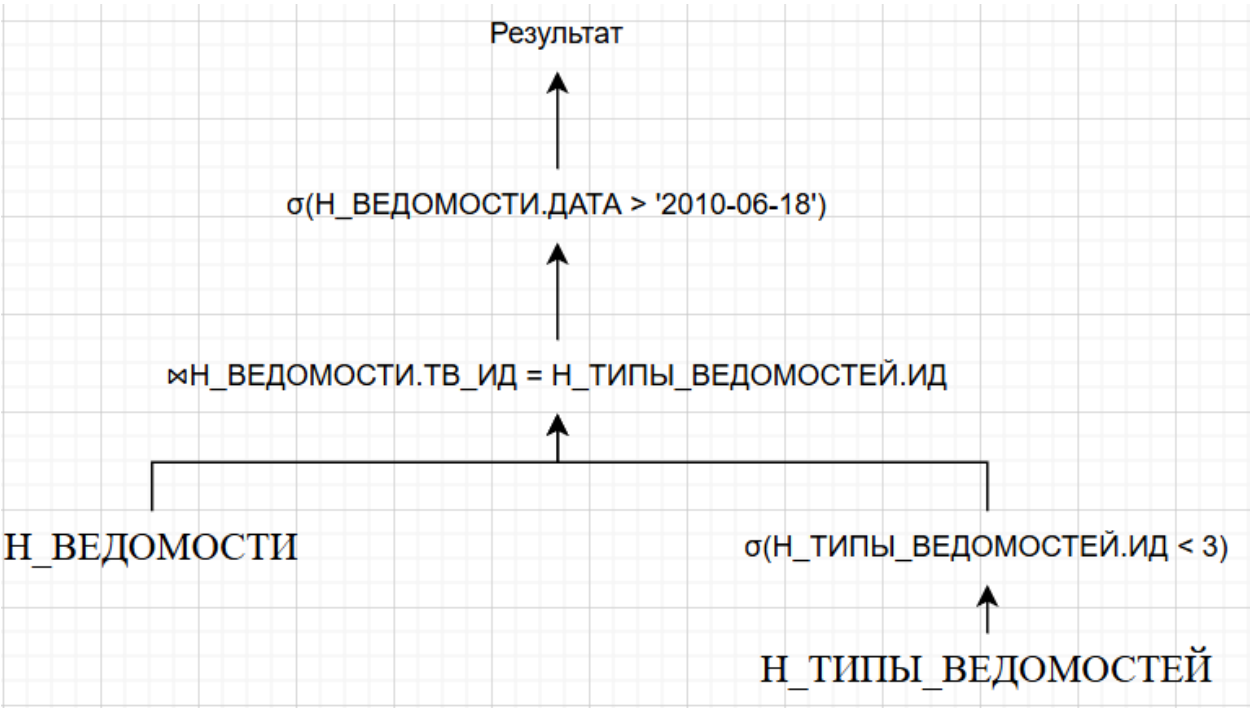

Рисунок 5 альтернативный план для запроса 1

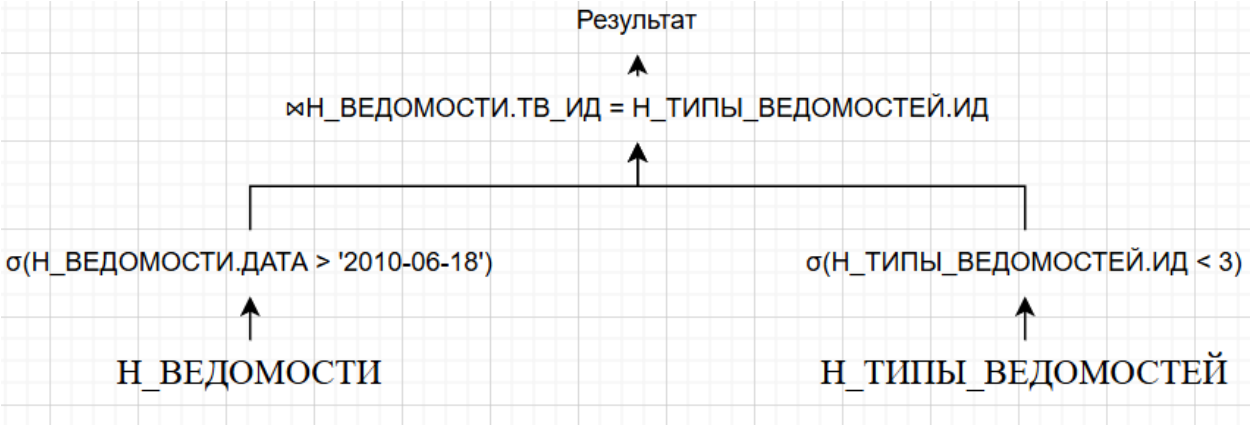

Рисунок 6 предлагаемая мной оптимальная схема для запроса 1

Я предлагаю выбрать нужные значения тип_ведомостей.ид перед соединением, потому что таблица н_типы_ведомостей гораздо меньше, чем н_ведомости и сокращение значений н_ведомости поможет избежать лишних обращений в н_ведомости и повторного обхода этих значений уже в объеденной таблице как в 1 плане. Как можно заметить ниже применение операции выборки оставляет меньше значений, чем операция соединения, поэтому оно должно выполняться раньше.

```
ucheb=> select count(*) from H_ВЕДОМОСТИ as ved where ved.ДАТА > '2010-06-18';
 count
-------
 25731
(1 строка)

ucheb=> SELECT count(*) FROM H_ТИПЫ_ВЕДОМОСТЕЙ
INNER JOIN H_ВЕДОМОСТИ ON H_ВЕДОМОСТИ.ТВ_ИД = H_ТИПЫ_ВЕДОМОСТЕЙ.ИД
WHERE H_ТИПЫ_ВЕДОМОСТЕЙ.ИД < 3
ucheb-> ;
 count
--------
 212365
(1 строка)
```

Запрос 2:

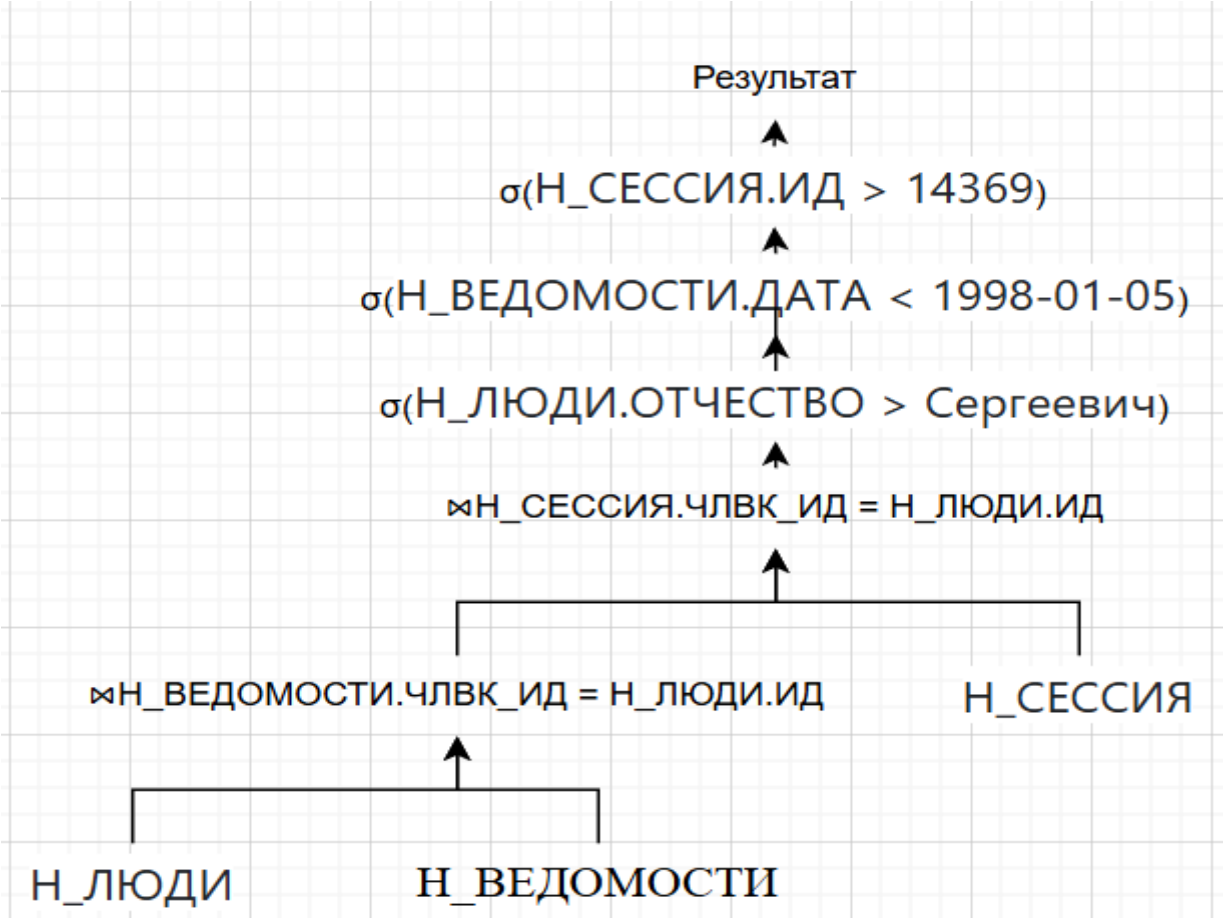

Рисунок 7 наивная схема

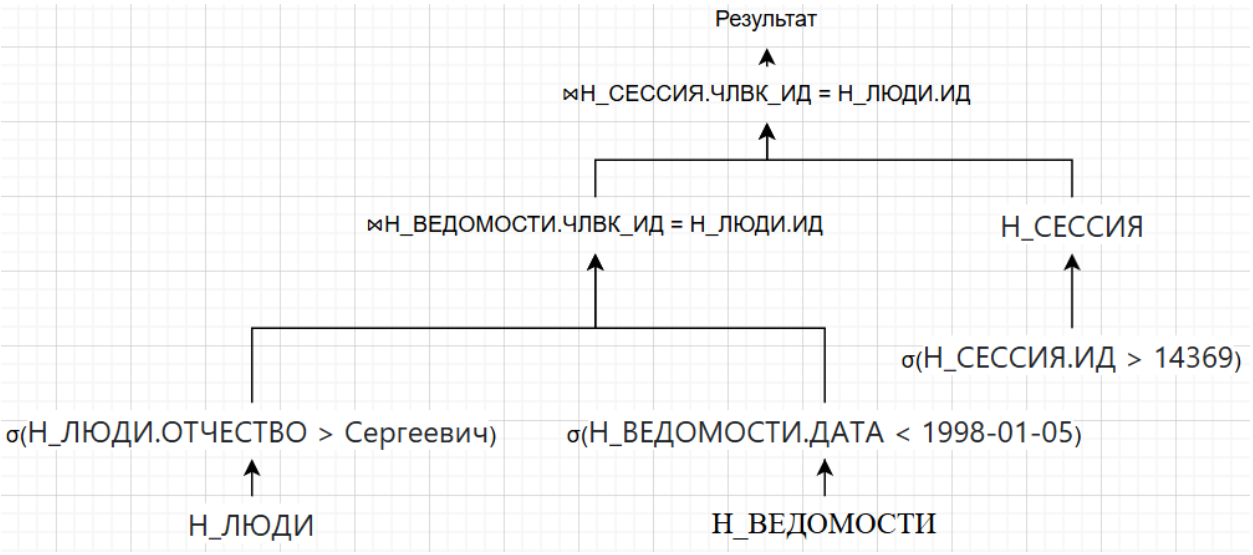

Рисунок 8 оптимальная схема

Т.к. тут все таблицы больших объёмов, то я решил, что будет быстрее сначала применять операцию выборки к таблице, и только после - операцию соединения.

При добавлении индексов планы не изменятся, однако будут работать быстрее.

вывод команды EXPLAIN ANALYZE

```
ucheb=> EXPLAIN ANALYZE SELECT Н_ТИПЫ_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н_ВЕДОМОСТИ.ИД FROM Н_ТИПЫ_ВЕДОМОСТЕЙ
INNER JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ТВ_ИД = Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД
WHERE Н_ТИПЫ_ВЕДОМОСТЕЙ.ИД < 3 AND Н_ВЕДОМОСТИ.ДАТА > '2010-06-18';
                                                                  QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------------------
 Hash Join  (cost=298.63..4891.76 rows=8657 width=422) (actual time=0.963..11.468 rows=22463 loops=1)
   Hash Cond: ("Н_ВЕДОМОСТИ"."ТВ_ИД" = "Н_ТИПЫ_ВЕДОМОСТЕЙ"."ИД")
   ->  Bitmap Heap Scan on "Н_ВЕДОМОСТИ"  (cost=297.58..4726.23 rows=25972 width=8) (actual time=0.905..5.083 rows=25731 loops=1)
         Recheck Cond: ("ДАТА" > '2010-06-18 00:00:00'::timestamp without time zone)
         Heap Blocks: exact=651
         ->  Bitmap Index Scan on "ВЕД_ДАТА_I"  (cost=0.00..291.08 rows=25972 width=0) (actual time=0.815..0.815 rows=25731 loops=1)
               Index Cond: ("ДАТА" > '2010-06-18 00:00:00'::timestamp without time zone)
   ->  Hash  (cost=1.04..1.04 rows=1 width=422) (actual time=0.027..0.028 rows=2 loops=1)
         Buckets: 1024  Batches: 1  Memory Usage: 9kB
         ->  Seq Scan on "Н_ТИПЫ_ВЕДОМОСТЕЙ"  (cost=0.00..1.04 rows=1 width=422) (actual time=0.016..0.017 rows=2 loops=1)
               Filter: ("ИД" < 3)
               Rows Removed by Filter: 1
 Planning Time: 1.071 ms
 Execution Time: 12.565 ms
(14 строк)
```

Рисунок 9 для запроса 1

```
ucheb=> EXPLAIN ANALYZE SELECT Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.УЧГОД FROM  Н_ЛЮДИ
LEFT JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
LEFT JOIN Н_СЕССИЯ ON Н_СЕССИЯ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
WHERE Н_ЛЮДИ.ОТЧЕСТВО > 'Сергеевич' AND Н_ВЕДОМОСТИ.ДАТА < '1998-01-05' AND Н_СЕССИЯ.ИД > 14369;
                                                                  QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------------------
 Nested Loop  (cost=0.86..16.98 rows=1 width=30) (actual time=0.006..0.007 rows=0 loops=1)
   Join Filter: ("Н_ВЕДОМОСТИ"."ЧЛВК_ИД" = "Н_СЕССИЯ"."ЧЛВК_ИД")
   ->  Nested Loop  (cost=0.58..15.53 rows=1 width=24) (actual time=0.005..0.006 rows=0 loops=1)
         ->  Index Scan using "ВЕД_ДАТА_I" on "Н_ВЕДОМОСТИ"  (cost=0.29..7.19 rows=1 width=4) (actual time=0.004..0.005 rows=0 loops=1)
               Index Cond: ("ДАТА" < '1998-01-05 00:00:00'::timestamp without time zone)
         ->  Index Scan using "ЧЛВК_РК" on "Н_ЛЮДИ"  (cost=0.28..8.30 rows=1 width=20) (never executed)
               Index Cond: ("ИД" = "Н_ВЕДОМОСТИ"."ЧЛВК_ИД")
               Filter: (("ОТЧЕСТВО")::text > 'Сергеевич'::text)
   ->  Index Scan using "SYS_C003500_IFK" on "Н_СЕССИЯ"  (cost=0.28..1.36 rows=8 width=14) (never executed)
         Index Cond: ("ЧЛВК_ИД" = "Н_ЛЮДИ"."ИД")
         Filter: ("ИД" > 14369)
 Planning Time: 1.222 ms
 Execution Time: 0.067 ms
(13 строк)
```

Рисунок 10 для запроса 2

```
ucheb=> EXPLAIN ANALYZE SELECT Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.УЧГОД FROM  Н_ЛЮДИ
LEFT JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
LEFT JOIN Н_СЕССИЯ ON Н_СЕССИЯ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
WHERE Н_ЛЮДИ.ОТЧЕСТВО > 'Сергеевич' AND Н_ВЕДОМОСТИ.ДАТА > '1998-01-05';
                                                                  QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------------------
 Hash Join  (cost=293.11..8175.03 rows=18257 width=30) (actual time=4.100..63.154 rows=16808 loops=1)
   Hash Cond: ("Н_ВЕДОМОСТИ"."ЧЛВК_ИД" = "Н_ЛЮДИ"."ИД")
   ->  Seq Scan on "Н_ВЕДОМОСТИ"  (cost=0.00..6884.50 rows=217295 width=4) (actual time=0.013..36.582 rows=217247 loops=1)
         Filter: ("ДАТА" > '1998-01-05 00:00:00'::timestamp without time zone)
         Rows Removed by Filter: 5193
   ->  Hash  (cost=287.73..287.73 rows=430 width=30) (actual time=4.056..4.062 rows=643 loops=1)
         Buckets: 1024  Batches: 1  Memory Usage: 45kB
         ->  Hash Right Join  (cost=169.35..287.73 rows=430 width=30) (actual time=2.969..3.929 rows=643 loops=1)
               Hash Cond: ("Н_СЕССИЯ"."ЧЛВК_ИД" = "Н_ЛЮДИ"."ИД")
               ->  Seq Scan on "Н_СЕССИЯ"  (cost=0.00..108.52 rows=3752 width=14) (actual time=0.007..0.350 rows=3752 loops=1)
               ->  Hash  (cost=163.97..163.97 rows=430 width=20) (actual time=2.951..2.955 rows=430 loops=1)
                     Buckets: 1024  Batches: 1  Memory Usage: 31kB
                     ->  Seq Scan on "Н_ЛЮДИ"  (cost=0.00..163.97 rows=430 width=20) (actual time=0.008..2.845 rows=430 loops=1)
                           Filter: (("ОТЧЕСТВО")::text > 'Сергеевич'::text)
                           Rows Removed by Filter: 4688
 Planning Time: 0.577 ms
 Execution Time: 63.971 ms
(17 строк)
```

Рисунок 11 для запроса 2, модифицированного, чтобы он что-то выводил.

Заключение

В результате выполнения лабораторной работы я узнал как СУБД интерпретирует запросы. Научился оптимизировать свои запросы, составлять план их выполнения.