

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ

«Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ
ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине

«ИНФОРМАЦИОННЫЕ СИСТЕМЫ И БАЗЫ ДАННЫХ»

вариант 11221

Выполнил:

Студент группы Р33312
Бинов Д.Е.

Преподаватель:
Наумова Н.А.

Санкт-Петербург, 2024

Текст задания

Введите вариант: 11221

Внимание! У разных вариантов разный 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:

Таблицы: Н_ОЦЕНКИ, Н_ВЕДОМОСТИ.

Вывести атрибуты: Н_ОЦЕНКИ.КОД, Н_ВЕДОМОСТИ.ИД.

Фильтры (AND):

- a) Н_ОЦЕНКИ.КОД < неявка.
- b) Н_ВЕДОМОСТИ.ЧЛВК_ИД = 117219.

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.ИД.

Фильтры (AND):

- a) Н_ЛЮДИ.ИД > 100865.
- b) Н_ВЕДОМОСТИ.ИД > 1457443.
- c) Н_СЕССИЯ.ДАТА = 2002-01-04.

Вид соединения: INNER JOIN.

1 запрос:

```
ucheb=> SELECT Н_ОЦЕНКИ.КОД, Н_ВЕДОМОСТИ.ИД
  FROM Н_ОЦЕНКИ
 INNER JOIN Н_ВЕДОМОСТИ ON Н_ОЦЕНКИ.КОД = Н_ВЕДОМОСТИ.ОЦЕНКА
 WHERE Н_ОЦЕНКИ.КОД < 'неявка'
 [AND Н_ВЕДОМОСТИ.ЧЛВК_ИД = 117219;
   КОД | ИД
 -----
 3 | 66986
 зачет | 67370
 5 | 64440
 3 | 64461
 3 | 67002
 3 | 67019
 3 | 67236
 4 | 67253
 зачет | 67350
 зачет | 67181
 зачет | 67287
 зачет | 67161
 зачет | 67302
 зачет | 67318
 3 | 67038
 4 | 67053
 3 | 64407
```

С использованием EXPLAIN ANALYZE:

```
QUERY PLAN
-----
Hash Join  (cost=1.49..200.82 rows=51 width=9) (actual time=0.047..0.083 rows=31 loops=1)
  Hash Cond: ((Н_ВЕДОМОСТИ"."ОЦЕНКА")::text = ("Н_ОЦЕНКИ"."КОД")::text)
    -> Index Scan using "ВЕД_ЧЛВК_FK_I FK" on "Н_ВЕДОМОСТИ" (cost=0.29..199.37 rows=65 width=10) (actual time=0.010..0.035 rows=31 loops=1)
        Index Cond: ("ЧЛВК_ИД" = 117219)
    -> Hash  (cost=1.11..1.11 rows=7 width=5) (actual time=0.025..0.025 rows=7 loops=1)
        Buckets: 1024  Batches: 1  Memory Usage: 9kB
        -> Seq Scan on "Н_ОЦЕНКИ"  (cost=0.00..1.11 rows=7 width=5) (actual time=0.012..0.015 rows=7 loops=1)
            Filter: (("КОД")::text < 'неявка'::text)
            Rows Removed by Filter: 2
Planning Time: 0.372 ms
Execution Time: 0.127 ms
(11 строк)

~
~
~
~
~
~
--More--(END)
```

2 запрос

```
учеб=> SELECT Н_ЛЮДИ.ФАМИЛИЯ, Н_ВЕДОМОСТИ.ЧЛВК_ИД, Н_СЕССИЯ.ИД
FROM Н_ЛЮДИ
INNER JOIN Н_ВЕДОМОСТИ ON Н_ВЕДОМОСТИ.ЧЛВК_ИД = Н_ЛЮДИ.ИД
INNER JOIN Н_СЕССИЯ ON Н_СЕССИЯ.ЧЛВК_ИД = Н_ВЕДОМОСТИ.ЧЛВК_ИД
WHERE Н_ЛЮДИ.ИД > 100865
    AND Н_ВЕДОМОСТИ.ИД > 1457443
    AND Н_СЕССИЯ.ДАТА = '2002-01-04';
ФАМИЛИЯ | ЧЛВК_ИД | ИД
-----+-----+-----
(0 строк)
```

ucheb=>

С использованием EXPLAIN ANALYZE:

```
----- ,  
 QUERY PLAN  
  
Nested Loop  (cost=0.58..179.47 rows=4 width=24) (actual time=0.472..0.473 rows=0 loops=1)  
  > Nested Loop  (cost=0.28..159.42 rows=4 width=28) (actual time=0.076..0.462 rows=4 loops=1)  
    > Seq Scan on "Н_СЕССИЯ"  (cost=0.00..117.90 rows=5 width=8) (actual time=0.065..0.443 rows=7 loops=1)  
      Filter: ("ДАТА" = '2002-04-08 00:00:00'::timestamp without time zone)  
      Rows Removed by Filter: 3745  
    > Index Scan using "ЧЛВК_РК" on "Н_ЛЮДИ"  (cost=0.28..8.30 rows=1 width=20) (actual time=0.002..0.002 rows=1 loops=7)  
      Index Cond: ((\"ИД\" = \"Н_СЕССИЯ\".\"ЧЛВК_ИД\") AND (\"ИД\" > 100865))  
  > Index Scan using "ВЕД_ЧЛВК_FK_1FK" on "Н_ВЕДОМОСТИ"  (cost=0.29..5.00 rows=1 width=4) (actual time=0.002..0.002 rows=0 loops=4)  
    Index Cond: ("ЧЛВК_ИД" = "Н_ЛЮДИ"."ИД")  
    Filter: ("ИД" > 1457443)  
Planning Time: 0.699 ms  
Execution Time: 0.514 ms  
(12 строк)
```

Создание индексов

1 запрос:

```
CREATE INDEX idx_code ON Н_ОЦЕНКИ using btree(КОД);
CREATE INDEX idx_vedomosti_human_id ON Н_ВЕДОМОСТИ using hash(ЧЛВК_ИД);
```

Для таблицы Н_ОЦЕНКИ для атрибута КОД имеет смысл создать индекс, так как он используется в фильтре. Имеет смысл использовать индекс на основе b-tree дерева, так как в фильтре используется оператор <. Создание данного индекса улучшит выборку данных при использовании WHERE.

Для таблицы Н_ВЕДОМОСТИ для атрибута ЧЛВК_ИД имеет смысл создать индекс, так как он используется в фильтре. Имеет смысл использовать индекс на основе hash, так как в фильтре используется оператор =. Используя такой индекс мы снизим время поиска при использовании WHERE.

2 запрос:

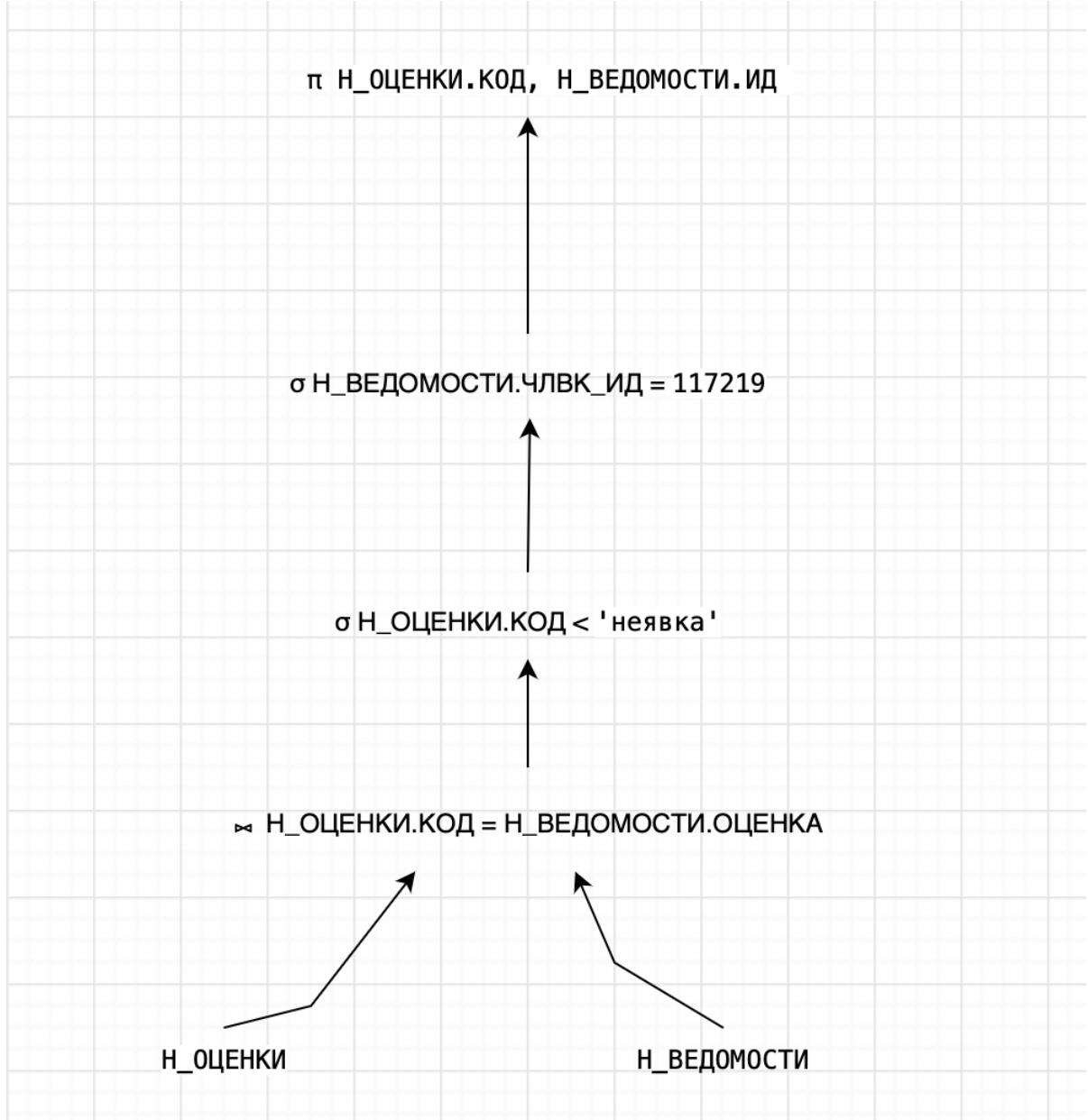
```
CREATE INDEX idx_human_id ON Н_ЛЮДИ using btree(ИД);
CREATE INDEX idx_vedomosti_id ON Н_ВЕДОМОСТИ using btree(ИД);
```

Для таблицы Н_ЛЮДИ для атрибута ИД имеет смысл создать индекс, так как он используется в фильтре. Имеет смысл использовать индекс на основе b-tree дерева, так как мы используем оператор >. Создание данного индекса улучшить выборку данных при использовании WHERE.

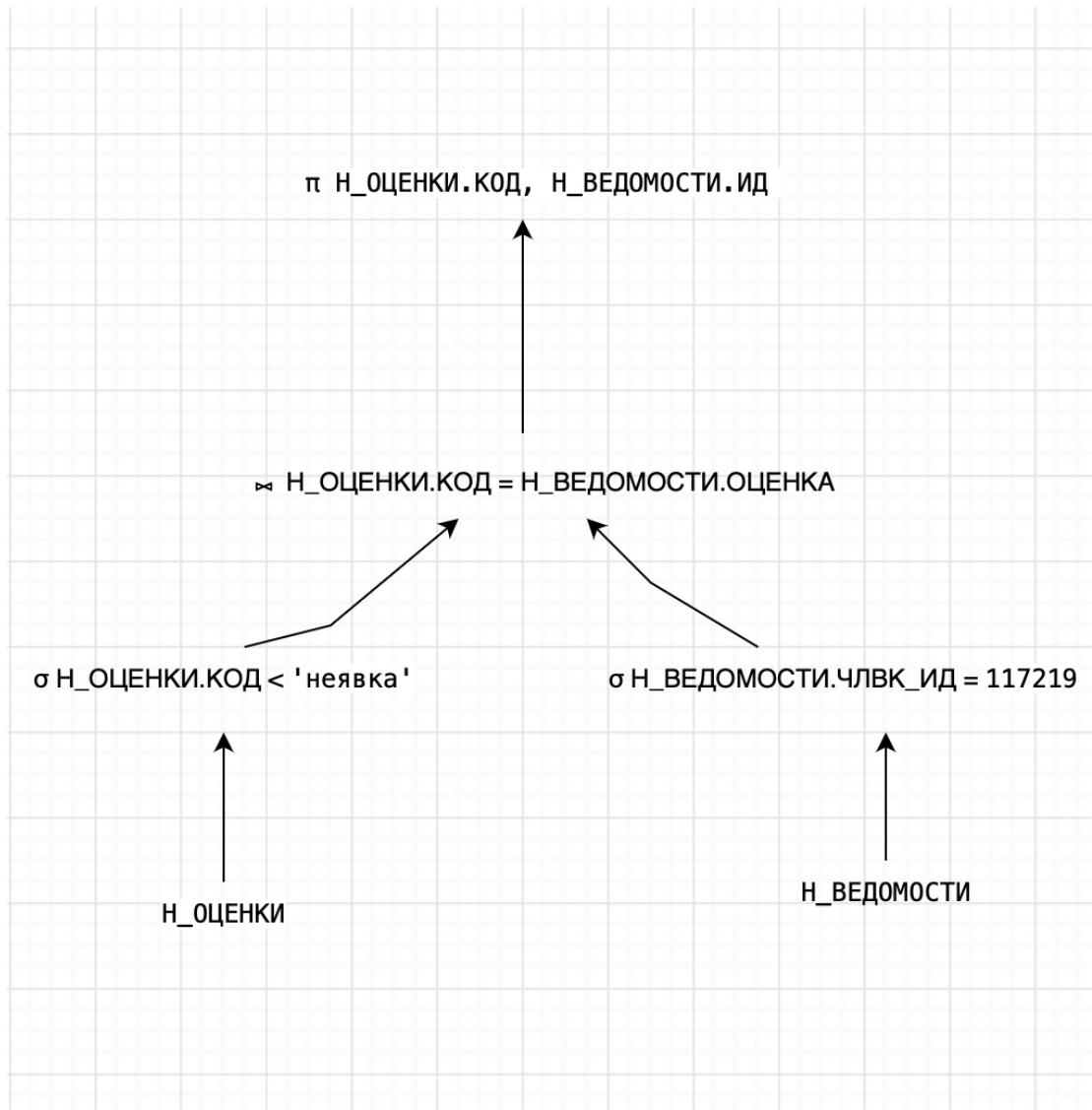
Для таблицы Н_ВЕДОМОСТИ для атрибута ИД имеет смысл создать индекс, так как он используется в фильтре. Имеет смысл использовать индекс на основе b-tree дерева, так как в фильтре используется оператор >. Используя такой индекс мы снизим время поиска с линейного до логарифмического ($O(n) \rightarrow O(\log N)$) при выполнении выборки данных при использовании WHERE.

1 запрос. План выполнения.

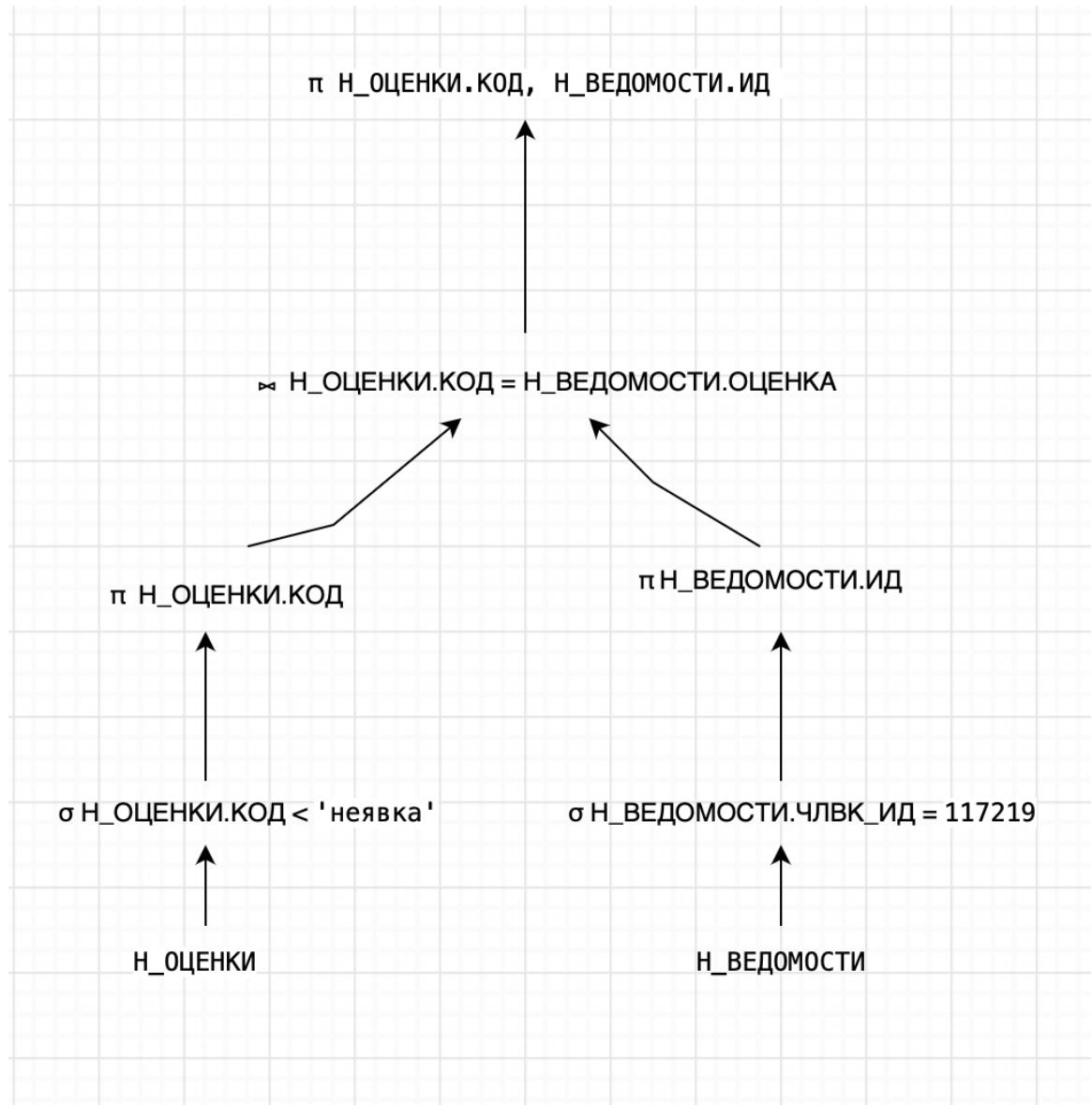
- 1) Сначала происходит соединение отношений, далее последовательная выборка, результатом будет проекция двух атрибутов



2) Сначала происходит операция выборки, после соединение двух отношений, результатом будет проекция двух атрибутов.

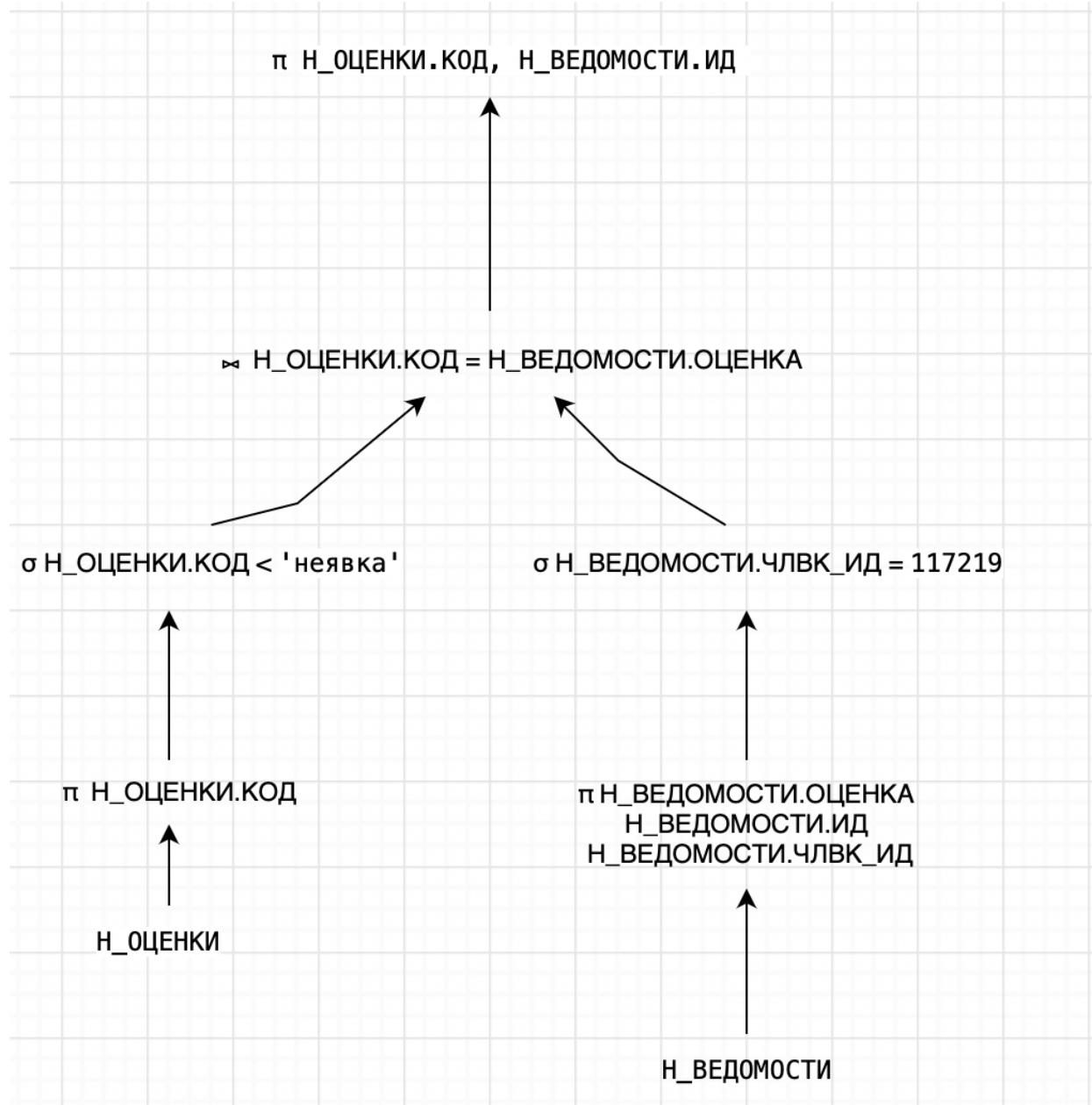


3) Сначала производится выборка для двух отношений, далее получается проекция для двух отношений из атрибутов, необходимых для выборки, соединения и итоговой проекции, далее соединение двух отношений, результатом является проекция двух атрибутов.

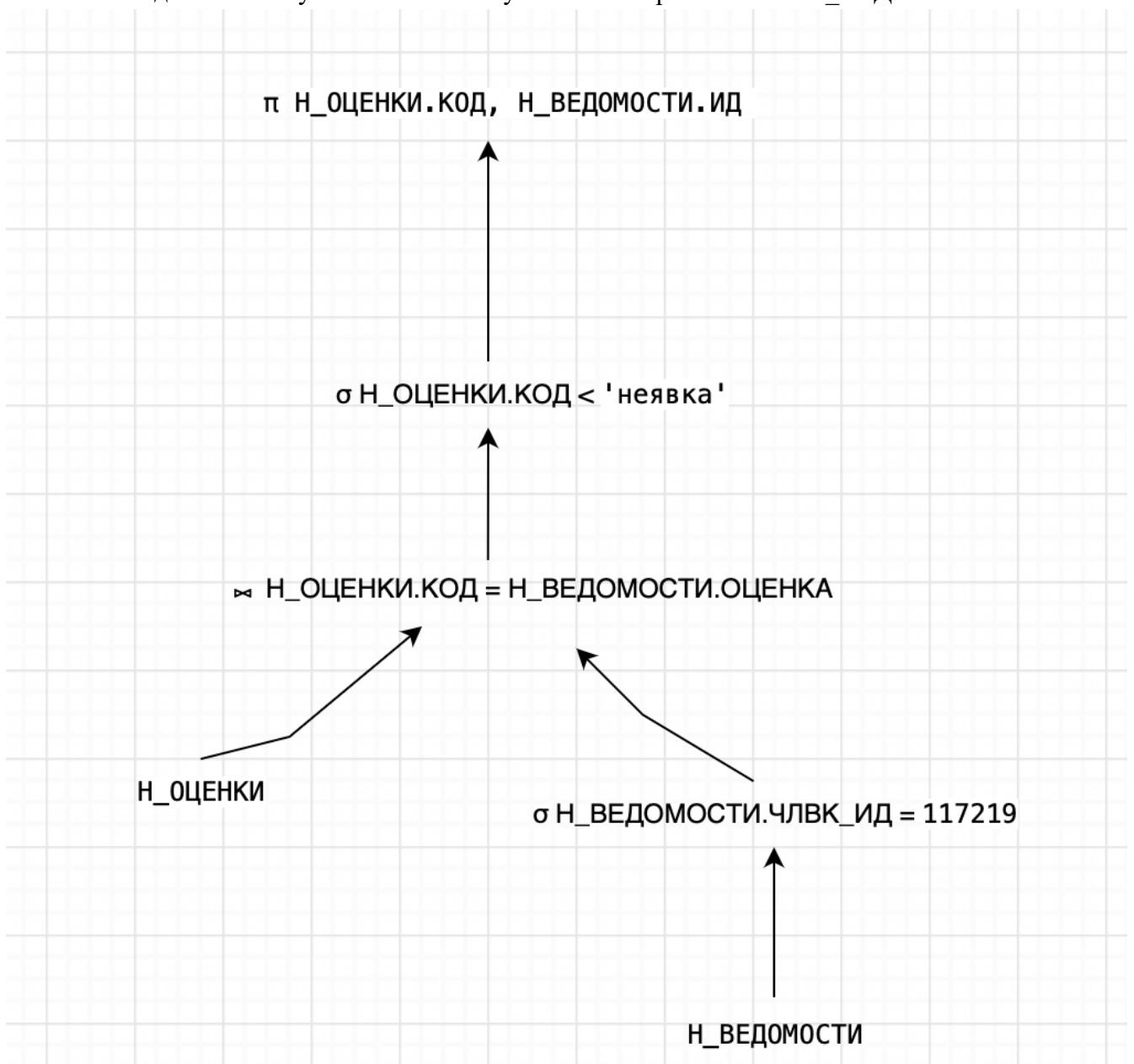


4) Для начала получается проекция для двух отношений из атрибутов, необходимых для выборки, соединения и итоговой проекции, далее производится выборка и соединение.

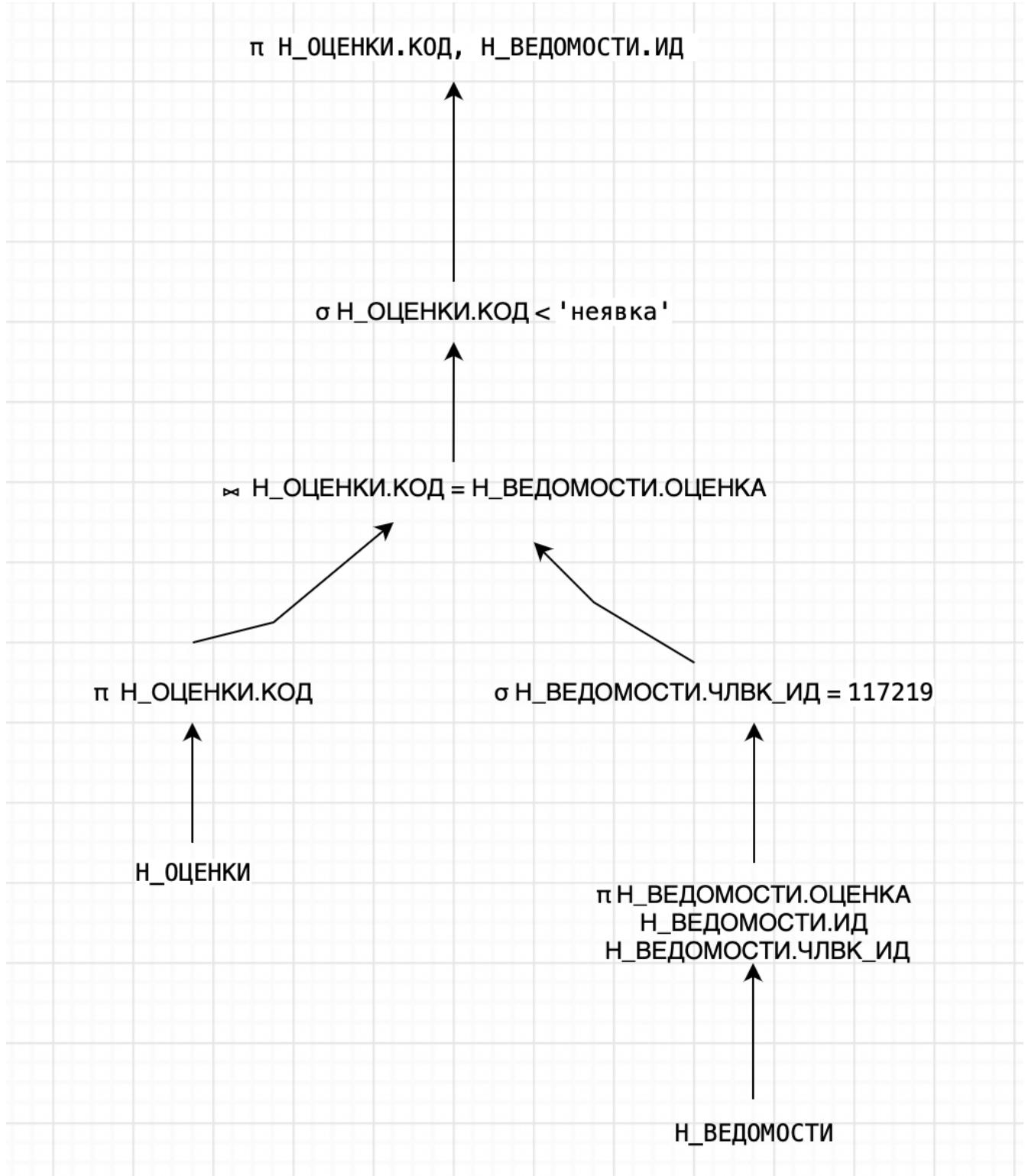
Результатом является проекция двух атрибутов.



- 5) Для начала получается выборка для одного из отношений из атрибутов, далее происходит соединение таблиц, после которого производится выборка для результата соединения. Результатом является проекция двух атрибутов. В данном плане атрибуты выборки после соединения могут меняться – могут быть выборки из ветки Н_ВЕДОМОСТИ.



6) Для начала получается проекция для двух отношений из атрибутов, необходимых для выборки, соединения и итоговой проекции, далее производится выборка для одного из отношений, а после соединения двух отношений, после которого производится выборка для результата отношения. Результатом является проекция двух атрибутов. В данном плане атрибуты выборки после соединения могут меняться – могут быть выборки из ветки Н_ВЕДОМОСТИ.

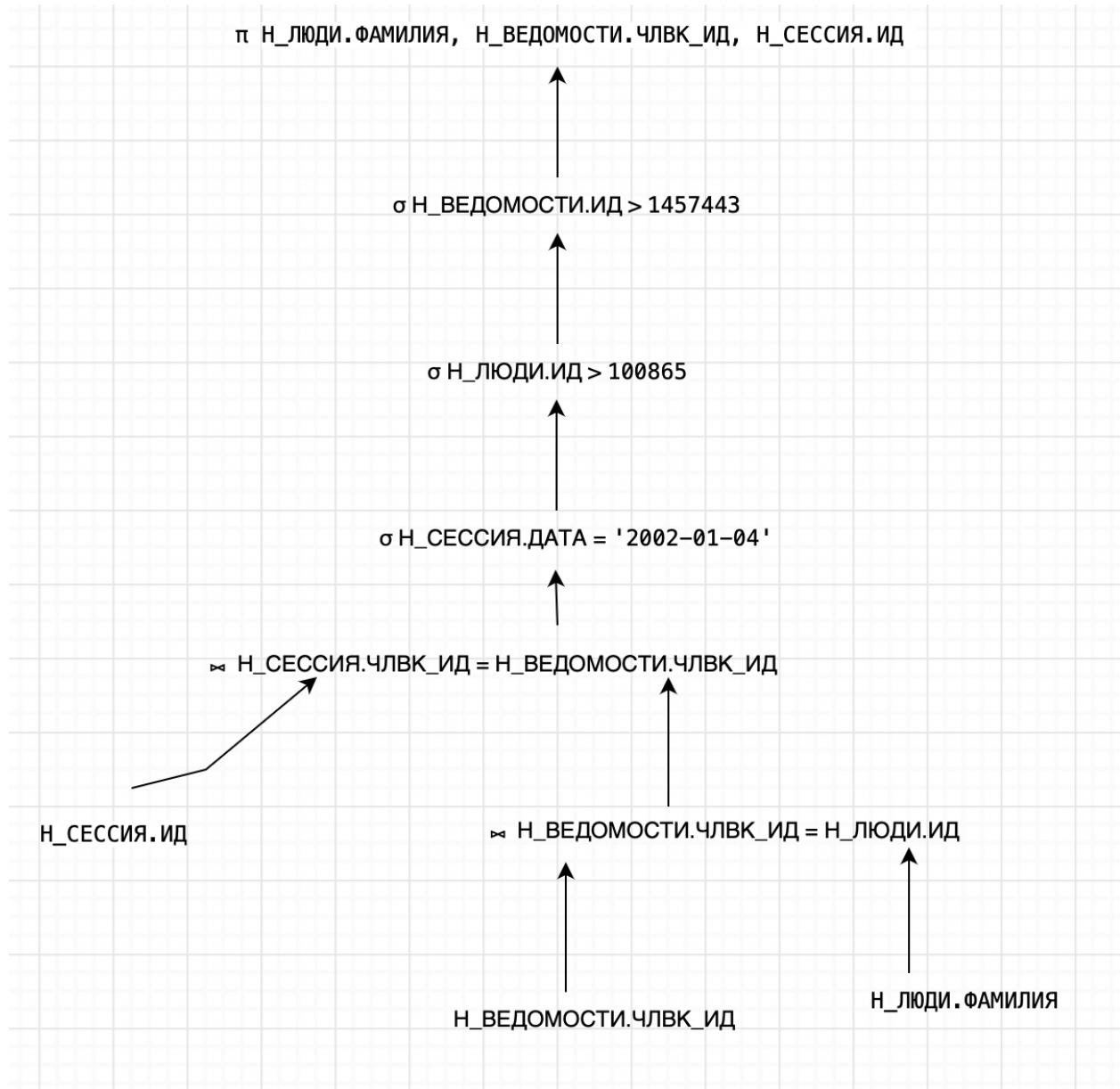


Из составленных возможных планов выполнения запроса лучшим является четвертый, поскольку в нем изначально получаются проекции и производятся выборки, а уже после этого выполняются соединение. Это позволяет уменьшить размер хранимых данных.

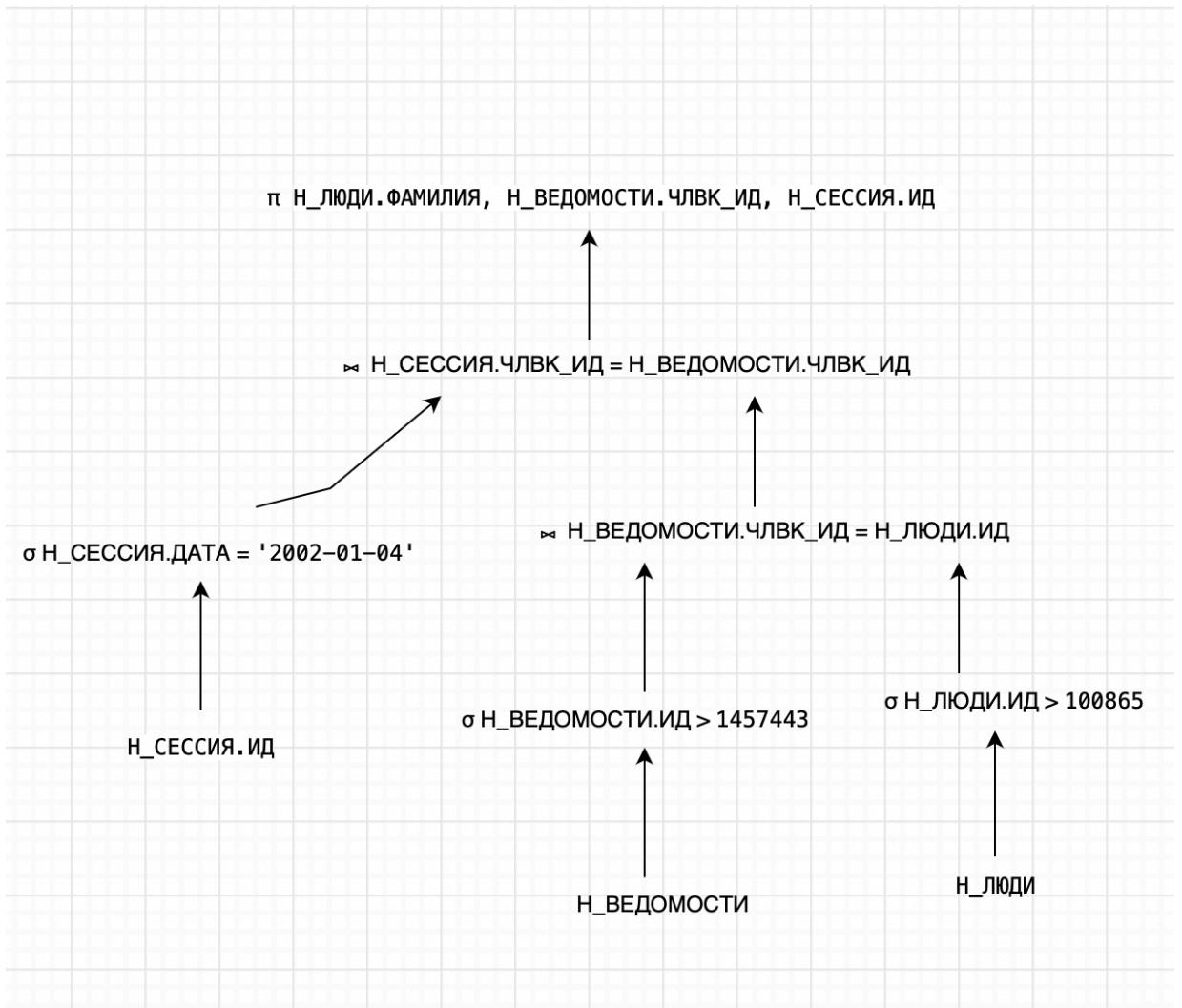
При создании индексов четвертый план останется оптимальным, при этом даже ускорится за счёт ускорения поиска. Также довольно эффективным будет третий план, потому что скорость поиска в нем увеличится. В нем также соединение происходит после выборки, что позволяет ускорить выполнение запроса, но, в отличие от четвертого плана, в нем выборка будет происходить по всей таблице целиком, а не по отдельным проекциям.

2 запрос. План выполнения.

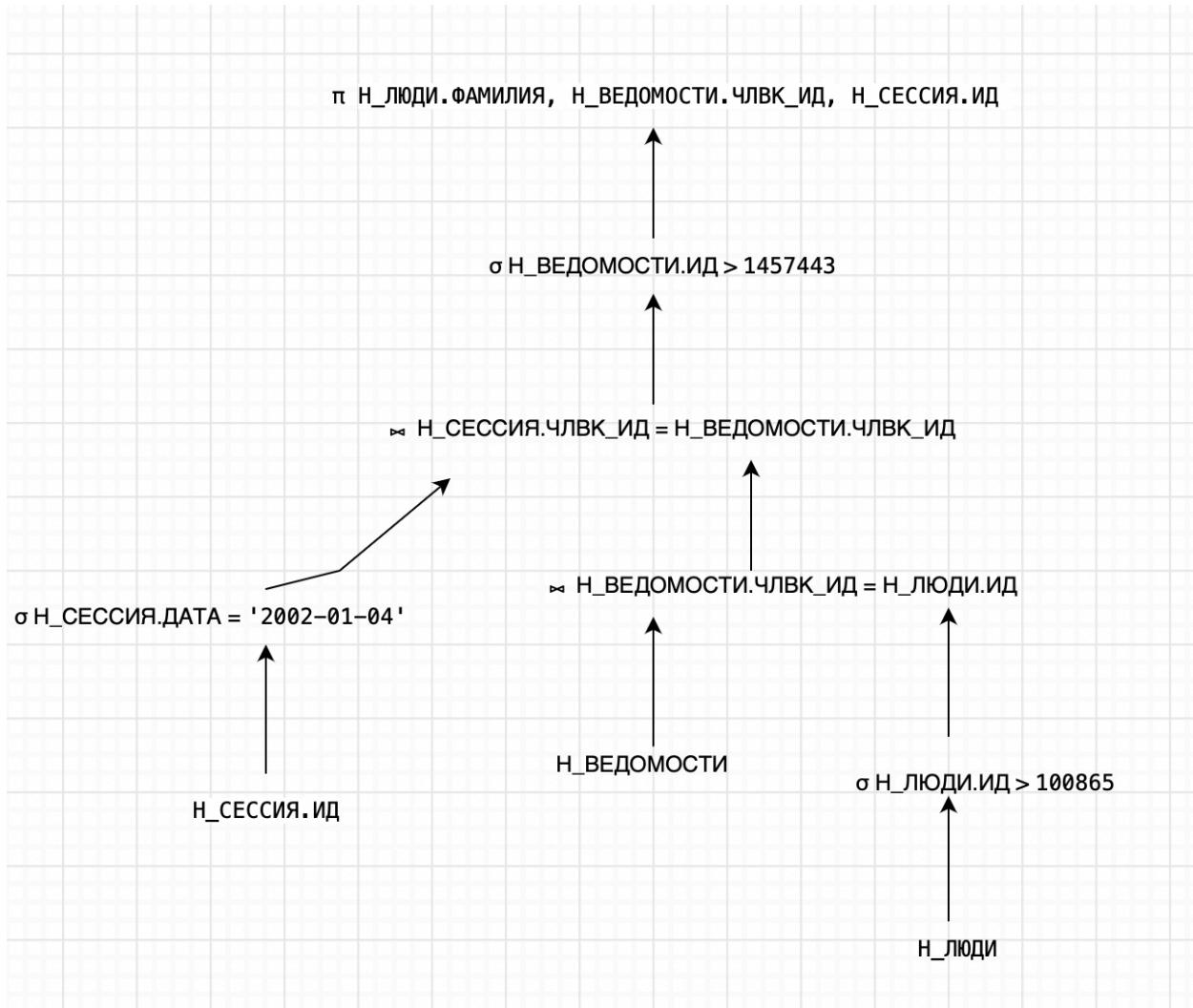
- 1) Соединение двух отношений, далее результат соединения соединяется с третьим отношением, а для результата двух соединений последовательно производится выборка. Результатом является проекция трех атрибутов.



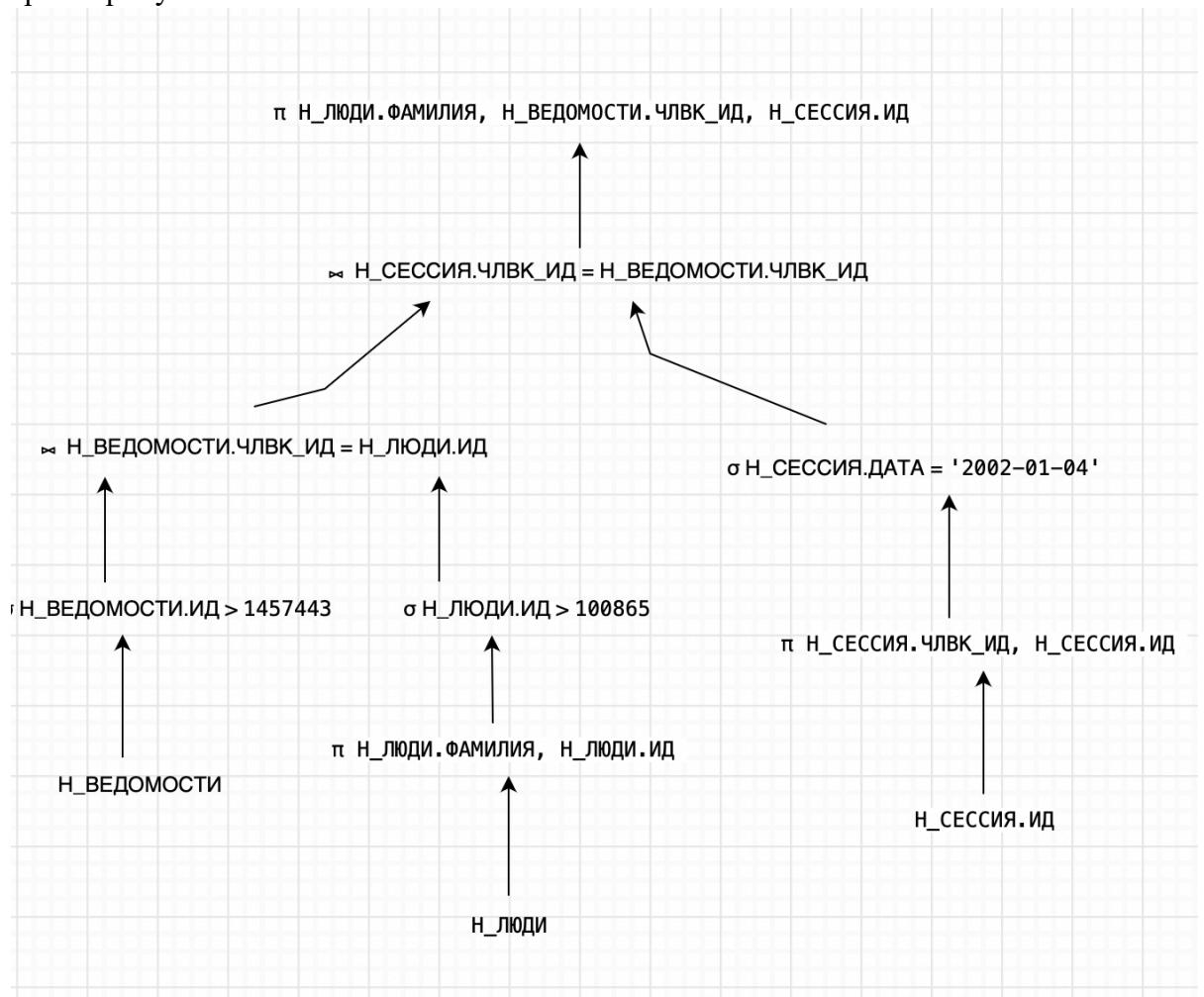
- 2) Для начала производится выборка для первых двух отношений, далее производится соединение этих двух отношений, а после результат соединения соединяется третьим отношением. Результатом является проекция трёх атрибутов.



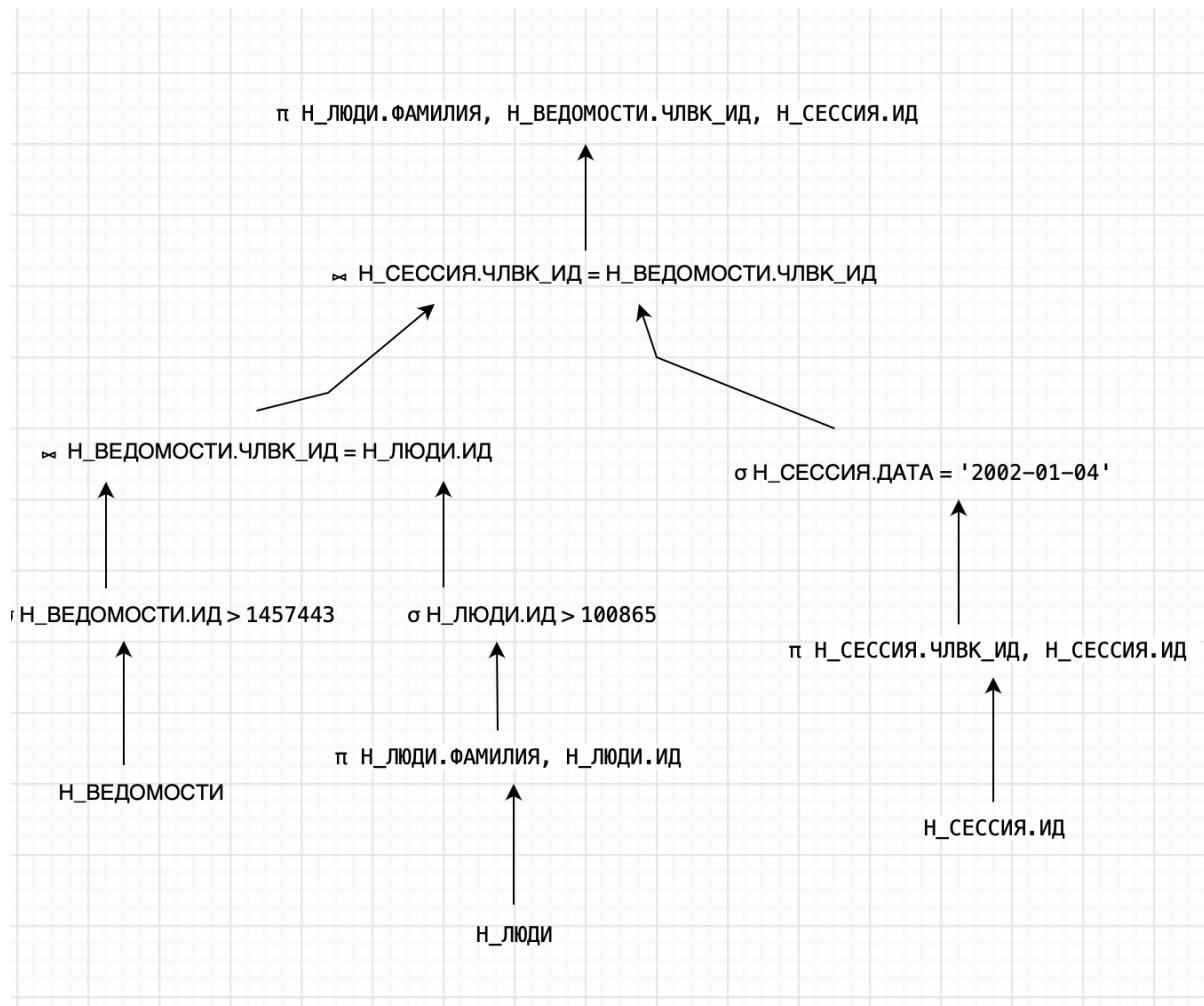
- 3) Сначала производится выборка для одного отношения, далее производится соединение первых двух отношений, а после результат соединения соединяется с третьим отношением. Происходит выборка по результату соединения. Результатом является проекция трёх атрибутов. На первом шаге выборка может быть у ветки Н_ОБУЧЕНИЯ.



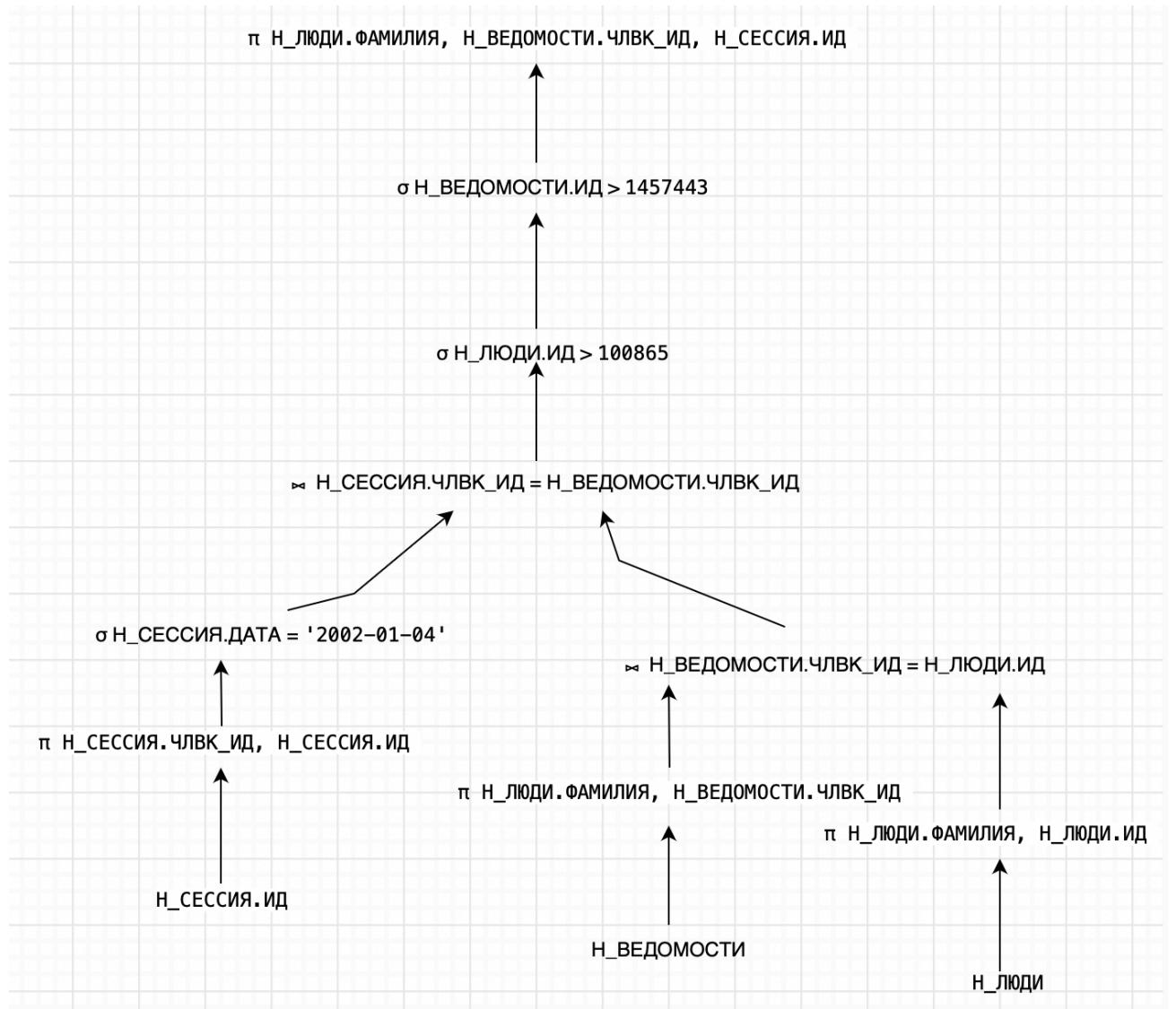
- 4) Сначала получается проекция для двух отношений, после по ним производится выборка и соединение. Далее результат соединения соединяется с третьим отношением, для которого перед соединением была получена проекция. Результатом является проекция трёх атрибутов.



5) Сначала формируются выборки для каждого отношения, далее строятся проекции. После чего данные проекции соединяются. Результатом является проекция трёх атрибутов.



6) Получаются проекции для двух отношений, после чего эти отношения соединяются, а результат соединения соединяется с третьим отношением, для которого перед соединением тоже была получена проекция. Далее производится последовательная выборка. Результатом является проекция трёх атрибутов.



Из составленных возможных планов выполнения запросы лучшим является четвертый, поскольку в нем изначально получаются проекции и производятся выборки, а уже после этого выполняется соединение. Это позволяет уменьшить размер хранимых данных.

При создании индексов четвертый план останется оптимальным, при этом даже ускорится за счёт ускорения поиска. Также довольно эффективным будет пятый план, потому что скорость поиска в нем увеличится. В нем также соединение происходит после выборки, что позволяет ускорить выполнение запроса, но, в отличие от четвертого плана, в нем выборка будет происходить по всей таблице целиком, а не по отдельным проекциям.

Вывод: узнал, что такое индексы, какие индексы бывают и как они помогают уменьшить время выполнения запроса.