Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники

**(ПИиКТ)**

A logo with blue dots and a red circle

Description automatically generated

## **ЛАБОРАТОРНАЯ РАБОТА №4**

По дисциплине

« Базы Данных »

Вариант № 3191

**Выполнил:**

Идрис Шуаибу

**Группа:**

P3131

**Проверил:**

Вербовой Александр Александрович

Санкт-Петербург 2025 г.

# **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\_ЛЮДИ, Н\_СЕССИЯ.  
   Вывести атрибуты: Н\_ЛЮДИ.ИМЯ, Н\_СЕССИЯ.УЧГОД.  
   Фильтры (AND):  
   a) Н\_ЛЮДИ.ОТЧЕСТВО > Владимирович.  
   b) Н\_СЕССИЯ.ИД > 14.  
   c) Н\_СЕССИЯ.ИД = 32199.  
   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.ДАТА, Н\_СЕССИЯ.ДАТА.  
   Фильтры (AND):  
   a) Н\_ЛЮДИ.ОТЧЕСТВО = Александрович.  
   b) Н\_ВЕДОМОСТИ.ДАТА > 2022-06-08.  
   Вид соединения: LEFT JOIN.

**Задание №1:**

Таблицы: Н\_ЛЮДИ, Н\_СЕССИЯ.  
Вывести атрибуты: Н\_ЛЮДИ.ИМЯ, Н\_СЕССИЯ.УЧГОД.  
Фильтры (AND):  
a) Н\_ЛЮДИ.ОТЧЕСТВО > Владимирович.  
b) Н\_СЕССИЯ.ИД > 14.  
c) Н\_СЕССИЯ.ИД = 32199.  
Вид соединения: INNER JOIN.

**Запрос:**

EXPLAIN ANALYZE

SELECT

Н\_ЛЮДИ.ИМЯ,

Н\_СЕССИЯ.УЧГОД

FROM

Н\_ЛЮДИ

INNER JOIN

Н\_СЕССИЯ ON Н\_ЛЮДИ.ИД = Н\_СЕССИЯ.ЧЛВК\_ИД

WHERE

Н\_ЛЮДИ.ОТЧЕСТВО > 'Владимирович'

AND Н\_СЕССИЯ.ИД > 14

AND Н\_СЕССИЯ.ИД = 32199;

|  |
| --- |
| **Result** |

Π(Н\_ЛЮДИ.ИМЯ, Н\_СЕССИЯ.УЧГОД)

σ (Н\_ЛЮДИ.ОТЧЕСТВО > 'Владимирович')

σ (Н\_СЕССИЯ.ИД > 14)

σ (Н\_СЕССИЯ.ИД = 32199)

⨝**θ** (Н\_ЛЮДИ.ИД = Н\_СЕССИЯ.ЧЛВК\_ИД)

Н\_ЛЮДИ Н\_СЕССИЯ

***fig:* Plan №1**

|  |
| --- |
| **Result** |

Π(Н\_ЛЮДИ.ИМЯ, Н\_СЕССИЯ.УЧГОД)

⨝**θ** (Н\_ЛЮДИ.ИД = Н\_СЕССИЯ.ЧЛВК\_ИД)

σ (Н\_ЛЮДИ.ОТЧЕСТВО > 'Владимирович') σ (Н\_СЕССИЯ.ИД > 14) ^ (Н\_СЕССИЯ.ИД = 32199)

Н\_ЛЮДИ Н\_СЕССИЯ

***fig:* Plan №2**

Выбранный план: **План №2**  
Этот план является более оптимальным, поскольку операции выбора (σ) применяются на самом раннем этапе — непосредственно к базовым таблицам. Это снижает объём данных, участвующих в операции соединения. Таким образом, уменьшается количество кортежей, передаваемых в ресурсоёмкую операцию соединения, что значительно повышает производительность, особенно при работе с большими объёмами данных.

**Индексы:**

Для ускорения выполнения запроса рекомендуется создать следующие индексы:

1. CREATE idx\_surname ON Н\_ЛЮДИ (ОТЧЕСТВО);

Н\_ЛЮДИ.ОТЧЕСТВО — индекс B-tree для ускорения фильтрации по диапазону (> 'Владимирович');

1. CREATE INDEX idx\_session\_id ON Н\_СЕССИЯ (ИД);

Н\_СЕССИЯ.ИД — индекс B-tree для ускорения поиска по равенству (= 32199) и диапазону (> 14);

1. CREATE INDEX idx\_person\_id ON Н\_СЕССИЯ (ЧЛВК\_ИД); Н\_СЕССИЯ.ЧЛВК\_ИД — индекс B-tree для ускорения соединения по внешнему ключу;
2. Н\_ЛЮДИ.ИД — используется в соединении и должен быть индексирован как первичный ключ.

**Влияние на план выполнения:**

После добавления индексов PostgreSQL заменит последовательные сканирования на **индексные сканирования(Index Scan)**, что существенно уменьшит количество обрабатываемых строк:

* Условия фильтрации будут обрабатываться с использованием индексов;
* Могут использоваться более эффективные алгоритмы соединения (например, Index Nested Loop Join);
* План может измениться с точки зрения порядка соединения таблиц;
* Общее время выполнения запроса уменьшится за счет снижения количества операций ввода-вывода и размера промежуточных результатов.

**Результат EXPLAIN ANALYZE:**

A computer screen with green text

AI-generated content may be incorrect.

**Задание №2:**

Таблицы: Н\_ЛЮДИ, Н\_ВЕДОМОСТИ, Н\_СЕССИЯ.  
Вывести атрибуты: Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА.  
Фильтры (AND):  
a) Н\_ЛЮДИ.ОТЧЕСТВО = Александрович.  
b) Н\_ВЕДОМОСТИ.ДАТА > 2022-06-08.  
Вид соединения: LEFT JOIN

**Запрос:**

EXPLAIN ANALYZE

SELECT

Н\_ЛЮДИ.ФАМИЛИЯ,

Н\_ВЕДОМОСТИ.ДАТА,

Н\_СЕССИЯ.ДАТА

FROM

Н\_ЛЮДИ

LEFT JOIN

Н\_ВЕДОМОСТИ ON Н\_ЛЮДИ.ИД = Н\_ВЕДОМОСТИ.ЧЛВК\_ИД

LEFT JOIN

Н\_СЕССИЯ ON Н\_ЛЮДИ.ИД = Н\_СЕССИЯ.ЧЛВК\_ИД

WHERE

Н\_ЛЮДИ.ОТЧЕСТВО = 'Александрович'

AND Н\_ВЕДОМОСТИ.ДАТА > '2022-06-08';

**Result**

Π (Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА)

Н\_СЕССИЯ ⟕ (Н\_ЛЮДИ.ИД = Н\_СЕССИЯ.ЧЛВК\_ИД)

⟕ (H\_ЛЮДИ.ИД = Н\_ВЕДОМОСТИ.ЧЛВК\_ИД)

σ(H\_ЛЮДИ.ОТЧЕСТВО = 'Александрович') σ(H\_ВЕДОМОСТИ.ДАТА > '2022-06-08)

Н\_ЛЮДИ Н\_ВЕДОМОСТИ

***fig:* Plan №1**

**Result**

Π (Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ДАТА)

Н\_СЕССИЯ ⟕ (Н\_ЛЮДИ.ИД = Н\_СЕССИЯ.ЧЛВК\_ИД)

σ(H\_ЛЮДИ.ОТЧЕСТВО = 'Александрович') σ(H\_ВЕДОМОСТИ.ДАТА > '2022-06-08)

⟕ (H\_ЛЮДИ.ИД = Н\_ВЕДОМОСТИ.ЧЛВК\_ИД)

Н\_ЛЮДИ Н\_ВЕДОМОСТИ

***fig:* Plan №2**

Выбранный план: **План №1**

**Пояснение:** План №1 использует стратегию **"ранней фильтрации" (selection pushdown)**, что позволяет значительно уменьшить количество обрабатываемых строк до выполнения соединения. Это снижает нагрузку на систему и ускоряет выполнение запроса.

**Индексы:**

а) **Таблица: Н\_ЛЮДИ**

* **Атрибут:** ОТЧЕСТВО
* **Индекс:** B-tree
* **Почему:** Используется в фильтре Н\_ЛЮДИ.ОТЧЕСТВО = 'Александрович'. Индекс по этому полю ускорит поиск нужных строк.
* CREATE INDEX idx\_people\_patronymic ON Н\_ЛЮДИ(ОТЧЕСТВО);

**б) Таблица: Н\_ВЕДОМОСТИ**

* **Атрибут:** ДАТА
* **Индекс:** B-tree
* **Почему:** Условие Н\_ВЕДОМОСТИ.ДАТА > '2022-06-08' позволит эффективно использовать B-tree для диапазонных запросов по дате.
* CREATE INDEX idx\_sheet\_date ON Н\_ВЕДОМОСТИ(ДАТА);

**в) Таблица: Н\_СЕССИЯ**

* **Атрибут:** ЧЛВК\_ИД
* **Индекс:** B-tree
* **Почему:** Используется в соединении с Н\_ЛЮДИ.ИД. Индекс ускорит поиск при соединении.
* CREATE INDEX idx\_session \_id ON Н\_СЕССИЯ(ЧЛВК\_ИД);

**г) Таблица: Н\_ВЕДОМОСТИ**

* **Атрибут:** ЧЛВК\_ИД
* **Индекс:** B-tree
* **Почему:** Также участвует в соединении с Н\_ЛЮДИ.ИД.
* **CREATE INDEX idx\_sheet\_person\_id ON Н\_ВЕДОМОСТИ(ЧЛВК\_ИД);**

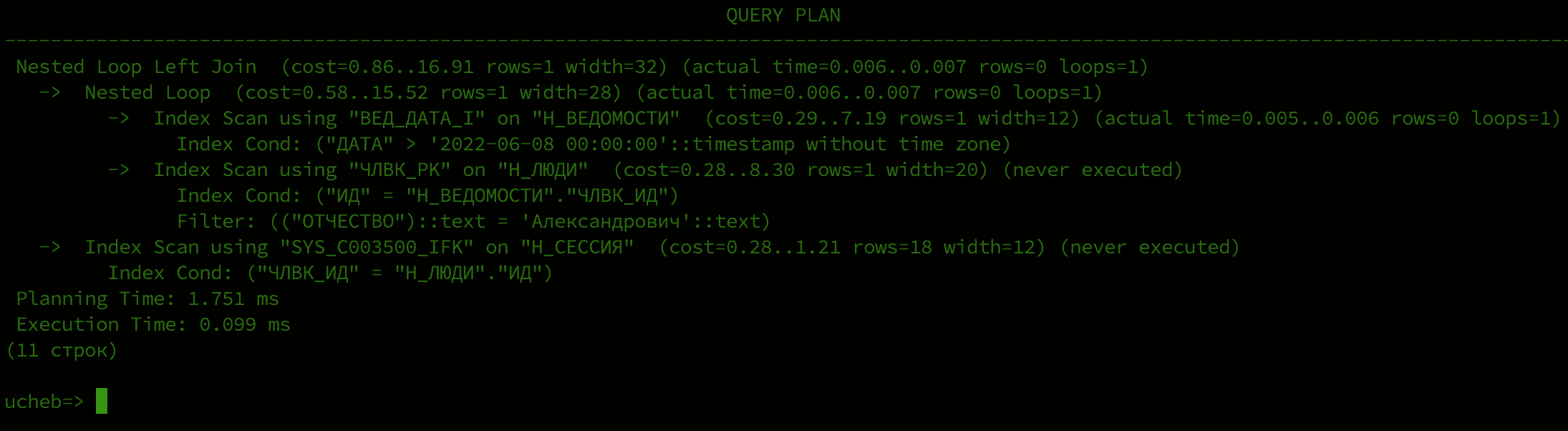
**Влияние на план выполнения:**

Без индексов PostreSQL использует полное сканирование таблиц (**Seq Scan**) для фильтрации и менее эффективные методы соединения, такие как **Nested Loop Join**. Это особенно замедляет выполнение запросов при работе с большими объемами данных.

Добавление индексов по ключевым атрибутам, таким как Н\_ЛЮДИ.ОТЧЕСТВО, Н\_СЕССИЯ.ИД и Н\_СЕССИЯ.ЧЛВК\_ИД, значительно повышает производительность. PostgreSQL получает возможность использовать **Index Scan** для фильтрации и более эффективные алгоритмы соединения, например, **Index Nested Loop Join** или **Hash Join**.

В результате план выполнения становится проще и быстрее: уменьшается количество операций чтения с диска, обрабатывается меньше строк, и общее время выполнения запроса существенно сокращается. Индексы позволяют извлекать только необходимые данные, ускоряя фильтрацию и соединение.

**Результат EXPLAIN ANALYZE:**

****

**Вывод**

В ходе выполнения лабораторной работы научился анализировать и визуализировать планы выполнения запросов с помощью реляционной алгебры. Осознал важность использования индексов в базах данных: без индексов PostgreSQL выполняет полный просмотр таблиц (Seq Scan), что приводит к низкой производительности, особенно при больших объемах данных. Индексы позволяют значительно ускорить выполнение запросов за счёт применения таких стратегий, как Index Scan и Index Nested Loop Join. Также я научился использовать команду EXPLAIN ANALYZE для оценки эффективности запросов и выбора оптимального плана выполнения.