Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Факультет «Информационных технологий и анализа больших данных»

**ОТЧЕТ**

**по практической работе № 10**

**по теме «Оптимизация запросов, часть 1.**

**Последовательное сканирование.**

**Индексы. Индексные структуры.**

**Алгоритмы доступа к данным. Битовые карты.»**

**по дисциплине Б.1.1.3.4 Системы управления базами данных**

**Вариант 1**

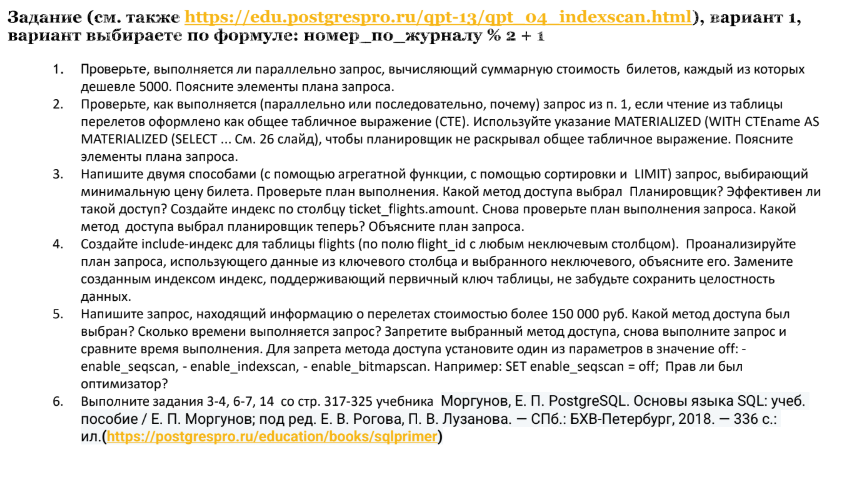
Выполнил: студент группы ИД23-1

Маслов Александр Николаевич

Проверил преподаватель: Цветков И.Е.

**Москва – 2025**

**Задания:**

****

**1.**

EXPLAIN (ANALYZE, VERBOSE)

SELECT SUM(amount)

FROM ticket\_flights

WHERE amount < 5000;

**Изображение выглядит как текст, Шрифт, снимок экрана, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

Данный запрос выполняется параллельно на это указывают такие строки как:

* **Gather -** собирает результаты от параллельных воркеров.
* **Workers Planned: 2 -** планировщик решил использовать 2 параллельных воркера.
* **Workers Launched: 2 -** было запущено 2 воркера — значит, параллелизм сработал.

**Finalize Aggregate -** финальный шаг агрегации: собирает частичные суммы, посчитанные параллельными воркерами, и агрегирует их в окончательное значение.

**2.**

EXPLAIN (ANALYZE, VERBOSE)

WITH t AS MATERIALIZED (

SELECT amount FROM ticket\_flights

)

SELECT SUM(amount) FROM t WHERE amount < 5000;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Нет признаков Finalize Aggregate, Gather, Partial — значит, агрегация не параллельная.

**Aggregate –** итоговая агрегация.

**CTE t -** определение CTE (временной подтаблицы). Структура для планировщика.

**Seq Scan on bookings.ticket\_flights -** последовательное сканирование таблицы ticket\_flights.

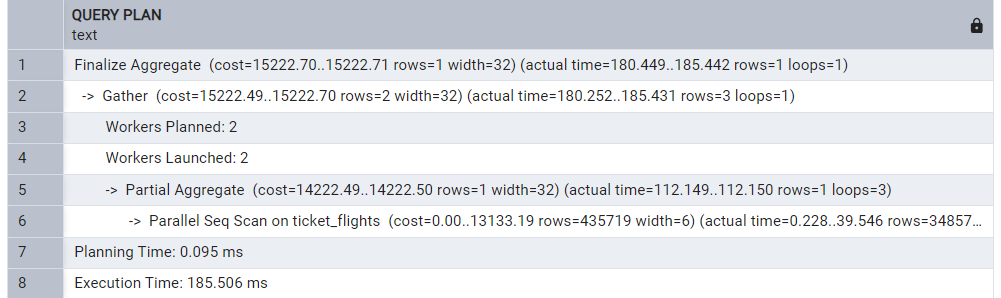
**3.**

**Агрегатная функция:**

EXPLAIN (ANALYZE)

SELECT MIN(amount)

FROM ticket\_flights;



Используется **Seq Scan и** параллельное выполнение.

**Сортировка и LIMIT:**

EXPLAIN (ANALYZE)

SELECT amount

FROM ticket\_flights

ORDER BY amount

LIMIT 1;

Изображение выглядит как текст, Шрифт, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Используется **Sort → Limit** и параллельное выполнение.

**После создания индекса:**

CREATE INDEX idx\_amount ON ticket\_flights(amount);

**Агрегатная функция:**

Изображение выглядит как текст, Шрифт, число, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Используется Index Only Scan using idx\_amount.

**Сортировка и LIMIT:**

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Используется Index Only Scan using idx\_amount.

**Result** - возвращает результат, который ранее был подготовлен.

**InitPlan 1** - инициализационный подзапрос.

**Limit** - ограничивает результат до 1 строки.

**Index Only Scan using idx\_amount on ticket\_flights** - используется Index Only Scan, а не Index Scan или Seq Scan. Это значит, что данные были получены исключительно из индекса (idx\_amount), без обращения к таблице (heap).

При использовании индекса скорость выполнения запроса возрастает в несколько раз.

**4.**

CREATE INDEX idx\_flight\_id\_include ON flights(flight\_id) INCLUDE(scheduled\_departure);

EXPLAIN (ANALYZE)

SELECT flight\_id, scheduled\_departure FROM flights WHERE flight\_id = '12';

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

Используется Index Only Scan using idx\_flight\_id\_include.

**Index Cond** – условие индекса.

**Удалить внешний ключ во ticket\_flights**

ALTER TABLE ticket\_flights

DROP CONSTRAINT ticket\_flights\_flight\_id\_fkey;

**Удалить первичный ключ flights\_pkey**

ALTER TABLE flights

DROP CONSTRAINT flights\_pkey;

**Создать новый UNIQUE индекс с INCLUDE**

CREATE UNIQUE INDEX flights\_pkey ON flights (flight\_id) INCLUDE (aircraft\_code);

**Назначить новый индекс как PRIMARY KEY**

ALTER TABLE flights

ADD CONSTRAINT flights\_pkey PRIMARY KEY USING INDEX flights\_pkey;

**Восстановить внешний ключ в ticket\_flights**

ALTER TABLE ticket\_flights

ADD CONSTRAINT ticket\_flights\_flight\_id\_fkey

FOREIGN KEY (flight\_id) REFERENCES flights(flight\_id);

**Проверка:**

EXPLAIN (ANALYZE)

SELECT flight\_id, aircraft\_code FROM flights WHERE flight\_id = '12';

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

**5.**

EXPLAIN (ANALYZE)

SELECT \* FROM ticket\_flights WHERE amount > 150000;

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Использовался метод Bitmap Index Scan и выполнение заняло 1.573 ms.

SET enable\_bitmapscan = off;

Изображение выглядит как текст, Шрифт, программное обеспечение, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Теперь используется метод Parallel Seq Scan и выполнение заняло 159.710 ms.

Параллельное выполнение является более оптимальным, чем последовательное, поэтому оптимизатор прав.

**6.**

**Задание 3**

EXPLAIN

WITH cte AS (

SELECT \* FROM ticket\_flights WHERE fare\_conditions = 'Economy'

)

SELECT \* FROM cte;

**Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.**

**Задание 4**

EXPLAIN

SELECT total\_amount

FROM bookings

ORDER BY total\_amount DESC

LIMIT 5;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

**Seq Scan on bookings:** последовательно считывает строки таблицы (154581 строк).

**Sort:** выполняет сортировку по total\_amount DESC, используя оперативную память.

**Limit:** берет первые 5 строк из отсортированного списка.

**Задание 6**

EXPLAIN

SELECT flight\_id, amount, RANK() OVER (ORDER BY amount DESC) AS rnk

FROM ticket\_flights;

**WindowAgg** — узел, где применяется оконная функция (например, RANK()).

**Sort** — сортировка по amount DESC для корректной работы оконной функции.

**Seq Scan** — чтение таблицы.