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 № 1**

**по теме «Подзапросы. CTE. Расширенное понятие о представлениях.»**

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

**Вариант 10**

Выполнил: студент группы ИД23-1

Маслов Александр Николаевич

Проверил преподаватель: Цветков И.Е.

**Москва – 2025**

**Задание №1.**

1. Напишите запрос на основе использования подзапросов, который выявляет направления, на которые было продано более 100 билетов.

**Запрос:**

SELECT departure\_airport, arrival\_airport, tickets\_sold

FROM (SELECT f.departure\_airport, f.arrival\_airport, COUNT(tf.ticket\_no) AS tickets\_sold

FROM flights f

JOIN ticket\_flights tf ON f.flight\_id = tf.flight\_id

GROUP BY f.departure\_airport, f.arrival\_airport)

WHERE tickets\_sold > 100

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Напишите запрос на основе использования подзапросов, который подсчитывает количество операций бронирования, в которых общая сумма не превышает q от p величины по всей выборке.

q = 0.9, p = максимальной

**Запрос:**

SELECT COUNT(\*) AS booking

FROM bookings

WHERE total\_amount <= (SELECT MAX(total\_amount) \* 0.9

FROM bookings)

Изображение выглядит как текст, снимок экрана, Шрифт, Прямоугольник

Контент, сгенерированный ИИ, может содержать ошибки.

1. Напишите запрос на основе использования подзапросов, который выводит для каждой модели самолета количество мест класса business, количество мест класса comfort и количество мест класса economy, которые занимали более p пассажиров в q-м месяце. p = 4500, q = 7

**Запрос:**

SELECT a.model,

(SELECT COUNT(DISTINCT s.seat\_no)

FROM seats s

JOIN flights f ON s.aircraft\_code = f.aircraft\_code

JOIN ticket\_flights tf ON f.flight\_id = tf.flight\_id

JOIN tickets t ON tf.ticket\_no = t.ticket\_no

WHERE s.fare\_conditions = 'Business' AND s.aircraft\_code = a.aircraft\_code AND EXTRACT(MONTH FROM f.scheduled\_departure) = 7

GROUP BY s.aircraft\_code

HAVING COUNT(t.ticket\_no) > 4500) AS business\_count,

(SELECT COUNT(DISTINCT s.seat\_no)

FROM seats s

JOIN flights f ON s.aircraft\_code = f.aircraft\_code

JOIN ticket\_flights tf ON f.flight\_id = tf.flight\_id

JOIN tickets t ON tf.ticket\_no = t.ticket\_no

WHERE s.fare\_conditions = 'Comfort' AND s.aircraft\_code = a.aircraft\_code AND EXTRACT(MONTH FROM f.scheduled\_departure) = 7

GROUP BY s.aircraft\_code

HAVING COUNT(t.ticket\_no) > 4500) AS comfort\_count,

(SELECT COUNT(DISTINCT s.seat\_no)

FROM seats s

JOIN flights f ON s.aircraft\_code = f.aircraft\_code

JOIN ticket\_flights tf ON f.flight\_id = tf.flight\_id

JOIN tickets t ON tf.ticket\_no = t.ticket\_no

WHERE s.fare\_conditions = 'Economy' AND s.aircraft\_code = a.aircraft\_code AND EXTRACT(MONTH FROM f.scheduled\_departure) = 7

GROUP BY s.aircraft\_code

HAVING COUNT(t.ticket\_no) > 4500) AS economy\_count

FROM aircrafts a

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Напишите запрос на основе использования подзапросов, который позволяет получить перечень аэропортов в тех городах (город, код аэропорта, название аэропорта), в которых больше q аэропортов.

**Запрос:**

SELECT city, airport\_code, airport\_name

FROM airports

WHERE city IN (SELECT city

FROM airports

GROUP BY city

HAVING COUNT(airport\_code) > 0)

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Придумайте и напишите по одному запросу для каждого выражения подзапроса из множества w к базе данных об авиаперевозках. Запросы у разных студентов не должны повторяться.

w = {NOT EXISTS, SOME, ALL}

[NOT EXISTS]

Найти всех пассажиров, которые никогда не летали эконом-классом.

**Запрос:**

SELECT DISTINCT t.passenger\_name

FROM tickets t

WHERE NOT EXISTS (SELECT \*

FROM ticket\_flights tf

JOIN flights f ON tf.flight\_id = f.flight\_id

WHERE tf.ticket\_no = t.ticket\_no AND tf.fare\_conditions = 'Economy')

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

[SOME]

Найти все рейсы, стоимость билетов комфорт класса которых больше, чем стоимость хотя бы одного билета бизнес класса.

**Запрос:**

SELECT DISTINCT tf.flight\_id, tf.fare\_conditions, tf.amount

FROM ticket\_flights tf

WHERE tf.fare\_conditions = 'Comfort' AND tf.amount > SOME (SELECT tf2.amount

FROM ticket\_flights tf2

WHERE tf2.fare\_conditions = 'Business')

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

[ALL]

Найти все аэропорты, у которых количество рейсов больше, чем количество рейсов во всех аэропортах с кодом VKO.

**Запрос:**

SELECT a.airport\_code, a.airport\_name

FROM airports a

JOIN flights f ON a.airport\_code = f.departure\_airport

GROUP BY a.airport\_code, a.airport\_name

HAVING COUNT(f.flight\_id) > ALL (SELECT COUNT(f2.flight\_id)

FROM flights f2

WHERE f2.departure\_airport = 'VKO'

GROUP BY f2.departure\_airport

)

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

**Задание №2.**

1. Напишите запрос на основе CTE, который ищет в базе перелеты, совершенные после даты p (день.месяц), и выводит по ним информацию о пассажирах, которые летели q-классом. p=18.08, q = Business.

**Запрос:**

WITH FlightsAfterDate AS (

SELECT \*

FROM flights

WHERE scheduled\_departure > '2017-08-18')

SELECT t.passenger\_id, t.passenger\_name, fad.scheduled\_departure, fad.departure\_airport, fad.arrival\_airport, tf.fare\_conditions

FROM FlightsAfterDate fad

JOIN ticket\_flights tf ON fad.flight\_id = tf.flight\_id

JOIN tickets t ON tf.ticket\_no = t.ticket\_no

WHERE tf.fare\_conditions = 'Business'

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Напишите запрос на основе CTE, который ищет максимальное количество пассажиров, совершивших минимальное количество перелетов в месяце q, в рейсах, совершенных из городов, название которые начинается с буквы p. q = 8, p = «А»

**Запрос:**

WITH FlightsFromA AS (

SELECT f.flight\_id

FROM flights f

JOIN airports a ON f.departure\_airport = a.airport\_code

WHERE a.city LIKE 'А%' AND EXTRACT(MONTH FROM f.scheduled\_departure) = 8

),

PassengerFlightCounts AS (

SELECT t.passenger\_id, COUNT(tf.flight\_id) AS flight\_count

FROM tickets t

JOIN ticket\_flights tf ON t.ticket\_no = tf.ticket\_no

JOIN flightsFromA f ON tf.flight\_id = f.flight\_id

GROUP BY t.passenger\_id

),

MinFlightCount AS (

SELECT MIN(flight\_count) AS min\_flight\_count

FROM PassengerFlightCounts

)

SELECT COUNT(pfc.passenger\_id) AS max\_passengers

FROM PassengerFlightCounts pfc

JOIN MinFlightCount mfc ON pfc.flight\_count = mfc.min\_flight\_count

Изображение выглядит как текст, снимок экрана, Шрифт, Прямоугольник

Контент, сгенерированный ИИ, может содержать ошибки.

1. Реализуйте рекурсивный запрос для вычисления значения выражения.

Сумма 11 слагаемых n+n\*1.5 + n\*3.5 при начальном n0 = 2, с шагом 2: n1 = 4 и т.д.

**Запрос:**

WITH RECURSIVE Series AS (

SELECT 2 AS n, 1 AS step, (2 + 2 \* 1.5 + 2 \* 3.5) AS total\_sum

UNION ALL

SELECT n + 2, step + 1, total\_sum + ((n + 2) + (n + 2) \* 1.5 + (n + 2) \* 3.5)

FROM Series

WHERE step < 11

)

SELECT SUM(total\_sum)

FROM Series

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

1. Выполните с помощью рекурсивных запросов обходы созданного дерева комментариев ( как создать - см. далее) в глубину и ширину, сделайте скриншоты запросов и результатов. У студентов результаты этой задачи не должны повторяться. Создайте таблицу, в которой будут храниться комментарии чата, имеющие иерархическую структуру: каждый комментарий может иметь до пяти потомков. Заполните таблицу выдуманными данными, в таблице должно быть: не менее двух комментариев с 2 потомками, не менее 3 комментариев с 3 потомками, не менее 4 комментариев с 4 потомками, комментарии с одним потомком, комментарии без потомков (следует хранить не только id комментариев, но и сами комментарии).

**Запрос:**

CREATE TABLE Commentaries (

comment\_id SERIAL PRIMARY KEY,

parent\_id INT REFERENCES Comments(comment\_id),

comment\_t TEXT NOT NULL

);

INSERT INTO Commentaries (parent\_id, comment\_t) VALUES

(NULL, 'Комментарий 1'),

(NULL, 'Комментарий 2'),

(1, 'Подкомментарий 1 Комментария 1'),

(1, 'Подкомментарий 2 Комментария 1'),

(2, 'Подкомментарий 1 Комментария 2'),

(2, 'Подкомментарий 2 Комментария 2'),

(3, 'Подподкомментарий 1 Подкомментария 1'),

(3, 'Подподкомментарий 2 Подкомментария 1'),

(4, 'Подподкомментарий 1 Подкомментария 2'),

(4, 'Подподкомментарий 2 Подкомментария 2'),

(5, 'Подподкомментарий 1 Подкомментария 1 Комментария 2'),

(5, 'Подподкомментарий 2 Подкомментария 1 Комментария 2'),

(6, 'Подподкомментарий 1 Подкомментария 2 Комментария 2'),

(6, 'Подподкомментарий 2 Подкомментария 2 Комментария 2'),

(7, 'Поподдподкомментарий 1 Подподкомментария 1'),

(7, 'Поподдподкомментарий 2 Подподкомментария 1'),

(8, 'Поподдподкомментарий 1 Подподкомментария 2'),

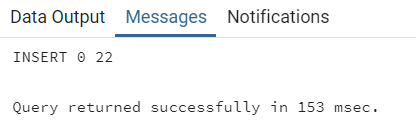
(8, 'Поподдподкомментарий 2 Подподкомментария 2'),

(9, 'Поподдподкомментарий 1 Подподкомментария 1 Комментария 2'),

(9, 'Поподдподкомментарий 2 Подподкомментария 1 Комментария 2'),

(10, 'Поподдподкомментарий 1 Подподкомментария 2 Комментария 2'),

(10, 'Поподдподкомментарий 2 Подподкомментария 2 Комментария 2');



Обход в глубину

**Запрос:**

WITH RECURSIVE DepthIn AS (

SELECT comment\_id, parent\_id, comment\_t, comment\_id::TEXT AS path

FROM commentaries

WHERE parent\_id IS NULL

UNION ALL

SELECT c.comment\_id, c.parent\_id, c.comment\_t, di.path || '->' || c.comment\_id::TEXT

FROM commentaries c

JOIN DepthIn di ON c.parent\_id = di.comment\_id

)

SELECT \* FROM DepthIn ORDER BY path

Изображение выглядит как текст, Шрифт, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Обход в ширину

**Запрос:**

WITH RECURSIVE WideIn AS (

SELECT comment\_id, parent\_id, comment\_t, 0 AS level

FROM commentaries

WHERE parent\_id IS NULL

UNION ALL

SELECT c.comment\_id, c.parent\_id, c.comment\_t, wi.level + 1

FROM commentaries c

JOIN WideIn wi ON c.parent\_id = wi.comment\_id

)

SELECT \* FROM WideIn ORDER BY level, comment\_id

**Изображение выглядит как текст, Шрифт, число, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

**Задание №3.**

1. Создайте обычное представление на основе запроса, включающего внутреннее соединение хотя бы двух таблиц (придумайте запрос). В запросе (где-либо) должен присутствовать номер Вашего варианта.

**Запрос:**

CREATE VIEW TicketFlightInfo AS

SELECT t.ticket\_no, t.passenger\_name, f.flight\_no, f.scheduled\_departure, f.departure\_airport, f.arrival\_airport

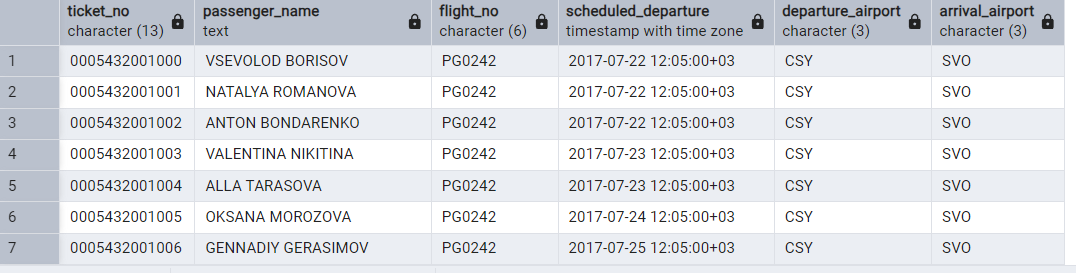
FROM tickets t

JOIN ticket\_flights tf ON t.ticket\_no = tf.ticket\_no

JOIN flights f ON tf.flight\_id = f.flight\_id

WHERE t.ticket\_no LIKE '%10%';

SELECT \* FROM TicketFlightInfo

****

1. Создайте пустое материализованное представление на основе любого запроса к БД. Выполните запрос для заполнения этого представления данными. В запросе (где-либо) должен присутствовать номер Вашего варианта.

**Запрос:**

CREATE MATERIALIZED VIEW FlightTicketCount AS

SELECT f.flight\_id, f.flight\_no, COUNT(tf.ticket\_no) AS ticket\_count

FROM flights f

JOIN ticket\_flights tf ON f.flight\_id = tf.flight\_id

GROUP BY f.flight\_id, f.flight\_no

HAVING COUNT(tf.ticket\_no) > 10

WITH NO DATA;

REFRESH MATERIALIZED VIEW FlightTicketCount;

SELECT \* FROM FlightTicketCount

