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 № 2**

**по теме «Оконные функции. Продвинутые методы работы с SQL»**

**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. С помощью запроса, использующего концепцию оконных функций, выведите накопленные суммы продаж билетов по q в каждом месяце. q = «неделям»

**Запрос:**

SELECT DISTINCT

EXTRACT(month FROM f.scheduled\_departure) AS month,

EXTRACT(week FROM f.scheduled\_departure) AS week,

SUM(tf.amount) OVER (

PARTITION BY EXTRACT(month FROM f.scheduled\_departure)

ORDER BY EXTRACT(week FROM f.scheduled\_departure)

) AS sum\_sales

FROM flights f

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

ORDER BY month, week

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. С помощью запроса, использующего концепцию оконных функций, по каждой модели самолета, выполняющего рейсы, выведите ежедневное количество рейсов, накопленное количество рейсов, скользящее среднее количества рейсов для фрейма p (учитываем, что фрейм находится внутри раздела), относительное количество рейсов, приходящееся на весь рассматриваемый период полетов. p - текущая строка и 4 следующие

**Запрос:**

WITH DailyFlightCounts AS (

SELECT a.model, DATE(f.scheduled\_departure) AS flight\_date, COUNT(f.flight\_id) AS daily\_flights

FROM aircrafts a

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

GROUP BY a.model, DATE(f.scheduled\_departure)

)

SELECT model, flight\_date, daily\_flights, SUM(daily\_flights) OVER(

PARTITION BY model

ORDER BY flight\_date

ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

) AS cumulative\_flights,

AVG(daily\_flights) OVER (

PARTITION BY model

ORDER BY flight\_date

ROWS BETWEEN CURRENT ROW AND 4 FOLLOWING

) AS moving\_avg\_flights,

daily\_flights \* 1.0 / SUM(daily\_flights) OVER(

PARTITION BY model

) AS relative\_flights

FROM DailyFlightCounts

ORDER BY model, flight\_date

Изображение выглядит как текст, снимок экрана, число, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.

1. С помощью запроса, использующего концепцию оконных функций, вычислите ранги моделей самолетов по количеству пассажиров в месяц.

**Запрос:**

WITH MonthlyPassengers AS (

SELECT a.model, EXTRACT(month FROM f.scheduled\_departure) AS month, COUNT(t.ticket\_no) AS passengers\_count

FROM aircrafts a

JOIN flights f ON a.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

GROUP BY a.model, EXTRACT(month FROM f.scheduled\_departure)

)

SELECT model, month, passengers\_count,

RANK() OVER(

PARTITION BY month

ORDER BY passengers\_count DESC

) AS rank\_by\_passengers

FROM MonthlyPassengers

ORDER BY month, rank\_by\_passengers

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

**Задание №2.**

1. Увеличьте количество записей в таблице t в p раз, аналогично увеличению количества записей в таблице advertisement. Выполните подсчет строк в таблице t, используя выборки из нее. Вероятность включения строки в выборку в процентах = q. t = ticket\_flights, p=4, q=1

**Запрос:**

WITH ticketflights\_expansion AS (

SELECT \*

FROM ticket\_flights

CROSS JOIN generate\_series(1,3))

SELECT COUNT(\*) AS estimated\_row\_count

FROM ticketflights\_expansion

WHERE random() < 0.01

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

1. Придумайте и реализуйте запрос с латеральным подзапросом для ранее созданной таблицы комментариев чата (хранящей дерево комментариев), в соответствии со своим вариантом.

**Запрос:**

SELECT parent.comment\_id AS parent\_id, parent.comment\_t AS parent\_text, child.comment\_id AS child\_id, child.comment\_t AS child\_text

FROM commentaries parent

LEFT JOIN LATERAL (

SELECT comment\_id, comment\_t

FROM commentaries child

WHERE child.parent\_id = parent.comment\_id

ORDER BY child.comment\_id

LIMIT 3

) AS child ON true

WHERE parent.parent\_id IS NULL

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

1. Напишите запрос, который находит p номер билета для каждого бронирования (используйте DISTINCT ON, а также альтернативные варианты). p = «минимальный»

**Запрос:**

SELECT DISTINCT ON (book\_ref) book\_ref, ticket\_no

FROM tickets

ORDER BY book\_ref, ticket\_no

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

1. Напишите запрос, который возвращает ранг числа x в множестве, которое начинается числом a, ограничено числом b, шаг между числами – h. Выведите все числа этого множества. x = 15, a = 1, b = 50, h = 2

**Запрос:**

WITH NumberSeries AS (SELECT generate\_series(1, 50, 2) AS num)

SELECT num, RANK() OVER (ORDER BY num) AS rank

FROM numberSeries

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.



1. Напишите запрос о распределении количества билетов по рейсам для p и q процентилей. p = 3, q = 99

**Запрос:**

WITH TicketCounts AS (

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

FROM ticket\_flights

GROUP BY flight\_id

),

Percentiles AS (

SELECT

PERCENTILE\_CONT(0.03) WITHIN GROUP (ORDER BY ticket\_count) AS p3,

PERCENTILE\_CONT(0.99) WITHIN GROUP (ORDER BY ticket\_count) AS p99

FROM TicketCounts

)

SELECT tc.flight\_id, tc.ticket\_count, p.p3, p.p99

FROM TicketCounts tc

CROSS JOIN Percentiles p

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Используя ROLLUP, GROUPING SETS, CUBE (напишите одни и те же запросы разными способами) вычислите суммы продаж билетов на авиарейсы по моделям самолетов, дням, месяцам и сумму продаж за весь период.

[ROLLUP]

**Запрос:**

SELECT a.model, EXTRACT(day FROM f.scheduled\_departure) AS day, EXTRACT(month FROM f.scheduled\_departure) AS month,

SUM (tf.amount) AS total\_sales

FROM ticket\_flights tf

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

JOIN aircrafts a ON f.aircraft\_code = a.aircraft\_code

GROUP BY ROLLUP (a.model, EXTRACT(month FROM f.scheduled\_departure), EXTRACT(day FROM f.scheduled\_departure))

ORDER BY a.model, month, day

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

Контент, сгенерированный ИИ, может содержать ошибки.**

[GROUPING SETS]

**Запрос:**

SELECT a.model, EXTRACT(day FROM f.scheduled\_departure) AS day, EXTRACT(month FROM f.scheduled\_departure) AS month,

SUM (tf.amount) AS total\_sales

FROM ticket\_flights tf

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

JOIN aircrafts a ON f.aircraft\_code = a.aircraft\_code

GROUP BY GROUPING SETS (

(a.model, EXTRACT(month FROM f.scheduled\_departure)),

(a.model, EXTRACT(day FROM f.scheduled\_departure)),

(a.model),

()

)

ORDER BY a.model, month, day

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

Контент, сгенерированный ИИ, может содержать ошибки.

[CUBE]

**Запрос:**

SELECT a.model, EXTRACT(day FROM f.scheduled\_departure) AS day, EXTRACT(month FROM f.scheduled\_departure) AS month,

SUM (tf.amount) AS total\_sales

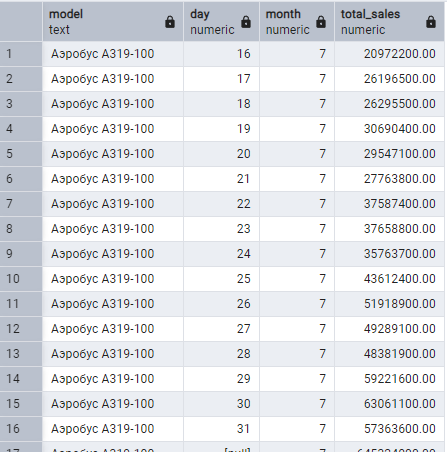
FROM ticket\_flights tf

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

JOIN aircrafts a ON f.aircraft\_code = a.aircraft\_code

GROUP BY CUBE (a.model, EXTRACT(month FROM f.scheduled\_departure), EXTRACT(day FROM f.scheduled\_departure))

ORDER BY a.model, month, day

****