**SQL**

1. Требуется составить расписание случайных проверок. Создайте оператор выбора, который выдаст 100 дат, начиная с текущей, при этом каждая дата отличается от предыдущей на 2-7 дней.

Пример:

|  |
| --- |
| 25.02.2023 |
| 28.02.2023 |
| 04.03.2023 |
| 06.03.2023 |
| 13.03.2023 |
| 16.03.2023 |
| …. |

**WITH RECURSIVE** dates AS

(

**SELECT** today() AS next\_date

**UNION ALL**

**SELECT** toDate(addDays(next\_date, rand() % 6 + 2)) AS next\_date

**FROM** dates

)

**SELECT** \*

**FROM** dates

**LIMIT** 100;

1. Требуется оценить эффективность продавцов. Создайте запрос, который вернёт количество и сумму продаж для каждого продавца, а также ранжирует продавцов по количеству продаж и по сумме продаж.

|  |  |  |
| --- | --- | --- |
| Таблица **employee** | | |
| ключ | **id** | **int** |
|  | **name** | **varchar** |

|  |  |  |
| --- | --- | --- |
| Таблица **sales** | | |
| ключ | **id** | **int** |
| внешний ключ | **employee\_id** | **int** |
|  | **price** | **int** |

Результат запроса должен содержать столбцы **id**, **name** из таблицы employee, а также столбцы:

**sales\_c** - количество продаж,

**sales\_rank\_c** - ранг по количеству продаж,

**sales\_s** - сумма продаж,

**sales\_rank\_s** - ранг по сумме продаж.

**WITH** sales\_summary AS (

**SELECT**

e.id,

e.name,

COUNT(s.id) AS sales\_c,

SUM(s.price) AS sales\_s,

ROW\_NUMBER() OVER (ORDER BY COUNT(s.id) DESC) AS sales\_rank\_c,

ROW\_NUMBER() OVER (ORDER BY SUM(s.price) DESC) AS sales\_rank\_s

**FROM** employee e

LEFT JOIN sales s ON e.id = s.employee\_id

**GROUP BY** e.id, e.name

)

**SELECT**

id,

name,

sales\_c,

sales\_rank\_c,

sales\_s,

sales\_rank\_s

**FROM** sales\_summary;

1. Имеется таблица денежных переводов transfers.

|  |  |
| --- | --- |
| Таблица **transfers** | |
| **from** | **int** |
| **to** | **int** |
| **amount** | **int** |
| **tdate** | **date** |

**from** – номер аккаунта, с которого сделан перевод,

**to** – номер аккаунта, на который сделан перевод,

**amount** – сумма перевода,

**tdate** – дата перевода.

Требуется создать оператор выбора, который для каждого аккаунта выведет периоды постоянства остатков. Результат запроса должен содержать столбцы **acc** – номер аккаунта, **dt\_from** - начало периода,

**dt\_to** - конец периода, **balance** – остаток на счёте в данном периоде.

Дата конца последнего периода – 01.01.3000.

Пример

Таблица **transfers**

|  |  |  |  |
| --- | --- | --- | --- |
| **from** | **to** | **amount** | **tdate** |
| 1 | 2 | 500 | 23.02.2023 |
| 2 | 3 | 300 | 01.03.2023 |
| 3 | 1 | 200 | 05.03.2023 |
| 1 | 3 | 400 | 05.04.2023 |

Результат запроса

|  |  |  |  |
| --- | --- | --- | --- |
| **acc** | **dt\_from** | **dt\_to** | **balance** |
| 1 | 23.02.2023 | 05.03.2023 | -500 |
| 1 | 05.03.2023 | 05.04.2023 | -300 |
| 1 | 05.04.2023 | 01.01.3000 | -700 |
| 2 | 23.02.2023 | 01.03.2023 | 500 |
| 2 | 01.03.2023 | 01.01.3000 | 200 |
| 3 | 01.03.2023 | 05.03.2023 | 300 |
| 3 | 05.03.2023 | 05.04.2023 | 100 |
| 3 | 05.04.2023 | 01.01.3000 | 500 |

**WITH RECURSIVE** cte AS (

**SELECT** "from" AS acc, tdate AS dt\_from, tdate AS dt\_to, -amount AS balance

**FROM** transfers

**UNION ALL**

**SELECT** t.to, t.tdate, c.dt\_to, c.balance + **CASE WHEN** t."from" = c.acc **THEN** -t.amount ELSE t.amount END

**FROM** transfers t

**JOIN** cte c ON (t."from" = c.acc OR t.to = c.acc) AND t.tdate >= c.dt\_from AND t.tdate <= c.dt\_to

)

**SELECT** acc, dt\_from, dt\_to, balance

**FROM** cte

**ORDER BY** acc, dt\_from;