Дитятев Иван

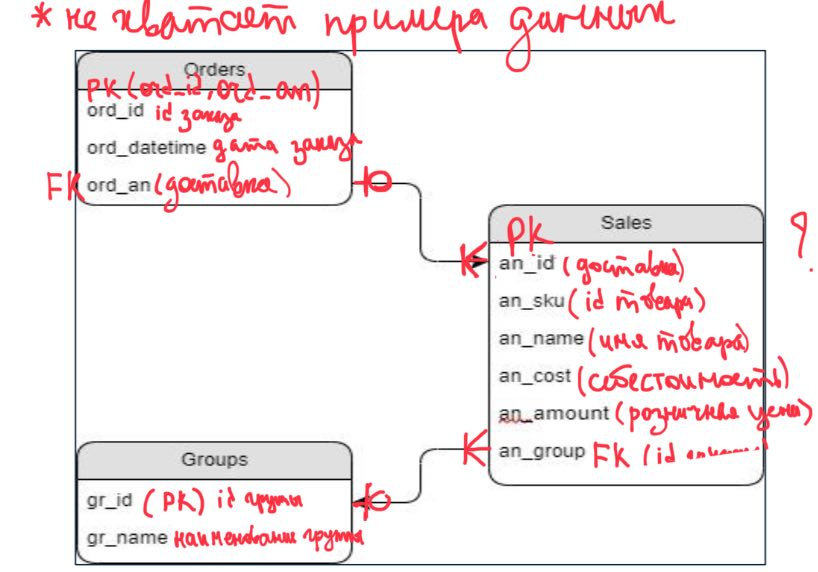
02/06/2024

Тестовое

Задание по SQL

Комментарии:

1. Писал на диалекте MS SQL SERVER
2. Так как не было примера данных, постарался представить, как это могло бы выглядеть, но точно знает только создатель, заранее отмечу возможность различного трактования.



Слегка потерялся в ER-диаграмме, на картинке выше уточнил, как я это вижу.

-- таблица Группы --------------------------------

IF OBJECT\_ID('Groups', 'U') IS NOT NULL

DROP TABLE Groups;

CREATE TABLE Groups (

gr\_id INT PRIMARY KEY,

gr\_name VARCHAR(255) NOT NULL

);

-- таблица Продажи---------------------------------

IF OBJECT\_ID('Sales', 'U') IS NOT NULL

DROP TABLE Sales;

CREATE TABLE Sales (

an\_id INT PRIMARY KEY,

an\_sku INT,

an\_name VARCHAR(255),

an\_cost DECIMAL(10, 2) CHECK(an\_cost >= 0),

an\_amount DECIMAL(10, 2) CHECK(an\_amount >= 0),

an\_group INT FOREIGN KEY REFERENCES Groups(gr\_id)

);

-- таблица Заказы ------------------------------------------

IF OBJECT\_ID('Orders', 'U') IS NOT NULL

DROP TABLE Orders;

CREATE TABLE Orders (

ord\_id INT NOT NULL,

ord\_datetime DATE DEFAULT GETDATE(),

ord\_an INT FOREIGN KEY REFERENCES Sales(an\_id),

PRIMARY KEY (ord\_id, ord\_an)

);

-- insert Groups

**INSERT** **INTO** **Groups** **VALUES** (1, **'Пылесосы'**);

**INSERT** **INTO** **Groups** **VALUES** (2, **'Телевизоры'**);

**INSERT** **INTO** **Groups** **VALUES** (3, **'Компьютеры'**);

**INSERT** **INTO** **Groups** **VALUES** (4, **'Смартфоны'**);

**INSERT** **INTO** **Groups** **VALUES** (5, **'Пылесосы-PRO'**);

**INSERT** **INTO** **Groups** **VALUES** (6, **'пылесосы от tfl'**);

**INSERT** **INTO** **Groups** **VALUES** (7, **'пылесосы0001'**);

**INSERT** **INTO** **Groups** **VALUES** (8, **'12\_пылесосы'**);

-- insert Sales

**INSERT** **INTO** Sales **VALUES** (1, 1001, **'Пылесос A'**, 500.00, 1000.00, 1);

**INSERT** **INTO** Sales **VALUES** (2, 1002, **'Пылесос B'**, 600.00, 1200.00, 1);

**INSERT** **INTO** Sales **VALUES** (3, 1003, **'Телевизор C'**, 700.00, 1400.00, 2);

**INSERT** **INTO** Sales **VALUES** (4, 1004, **'Пылесос C'**, 800.00, 1600.00, 1);

**INSERT** **INTO** Sales **VALUES** (5, 1005, **'Телевизор D'**, 900.00, 1800.00, 2);

**INSERT** **INTO** Sales **VALUES** (6, 1006, **'Телевизор E'**, 1000.00, 2000.00, 2);

**INSERT** **INTO** Sales **VALUES** (7, 1007, **'Компьютер A'**, 1100.00, 2200.00, 3);

**INSERT** **INTO** Sales **VALUES** (8, 1008, **'Компьютер B'**, 1200.00, 2400.00, 3);

**INSERT** **INTO** Sales **VALUES** (9, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (10, 1010, **'Пылесос'**, 1800.00, 2300.00, 1);

**INSERT** **INTO** Sales **VALUES** (11, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (12, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (13, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (14, 1010, **'Пылесос'**, 1800.00, 2300.00, 5);

**INSERT** **INTO** Sales **VALUES** (15, 1010, **'Пылесос'**, 1800.00, 2300.00, 6);

**INSERT** **INTO** Sales **VALUES** (16, 1010, **'Пылесос'**, 1800.00, 2300.00, 7);

**INSERT** **INTO** Sales **VALUES** (17, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (18, 1010, **'Пылесос'**, 1800.00, 2300.00, 8);

**INSERT** **INTO** Sales **VALUES** (19, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (20, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

**INSERT** **INTO** Sales **VALUES** (21, 1009, **'Компьютер C'**, 1300.00, 2600.00, 3);

-- insert Orders

**INSERT** **INTO** Orders **VALUES** (1, **'2022-03-04 10:00:00'**, 1);

**INSERT** **INTO** Orders **VALUES** (1, **'2024-03-21 10:00:00'**, 2);

**INSERT** **INTO** Orders **VALUES** (1, **'2024-03-23 10:00:00'**, 20);

**INSERT** **INTO** Orders **VALUES** (1, **'2024-03-06 10:00:00'**, 14);

**INSERT** **INTO** Orders **VALUES** (1, **'2024-03-07 10:00:00'**, 15);

**INSERT** **INTO** Orders **VALUES** (1, **'2024-03-09 10:00:00'**, 19);

**INSERT** **INTO** Orders **VALUES** (1, **'2024-03-10 10:00:00'**, 21);

**INSERT** **INTO** Orders **VALUES** (2, **'2021-03-05 11:00:00'**, 2);

**INSERT** **INTO** Orders **VALUES** (2, **'2024-03-11 10:00:00'**, 1);

**INSERT** **INTO** Orders **VALUES** (3, **'2024-03-14 10:00:00'**, 1);

**INSERT** **INTO** Orders **VALUES** (3, **'2024-03-24 10:00:00'**, 2);

**INSERT** **INTO** Orders **VALUES** (3, **'2019-03-06 12:00:00'**, 3);

**INSERT** **INTO** Orders **VALUES** (4, **'2024-06-07 13:00:00'**, 4);

**INSERT** **INTO** Orders **VALUES** (4, **'2024-03-16 10:00:00'**, 1);

**INSERT** **INTO** Orders **VALUES** (5, **'2024-06-08 14:00:00'**, 5);

**INSERT** **INTO** Orders **VALUES** (5, **'2024-03-15 10:00:00'**, 6);

**INSERT** **INTO** Orders **VALUES** (6, **'2024-05-09 15:00:00'**, 6);

**INSERT** **INTO** Orders **VALUES** (7, **'2024-05-10 16:00:00'**, 7);

**INSERT** **INTO** Orders **VALUES** (8, **'2024-03-10 17:00:00'**, 8);

**INSERT** **INTO** Orders **VALUES** (9, **'2024-03-12 18:00:00'**, 9);

**INSERT** **INTO** Orders **VALUES** (11, **'2024-03-04 10:00:00'**, 1);

**INSERT** **INTO** Orders **VALUES** (11, **'2024-03-04 11:00:00'**, 2);

**INSERT** **INTO** Orders **VALUES** (11, **'2024-03-04 11:00:00'**, 3);

**INSERT** **INTO** Orders **VALUES** (13, **'2024-03-04 11:00:00'**, 12);

--Задача 1.

--Вывести номера заказов, где в группах товаров встречается слово “пылесос”.

Комментарий: не совсем ясно должен ли учитываться регистр, должно ли быть слово в начале или конце, допустимо ли наличие других символов в слове

SELECT DISTINCT ord\_id

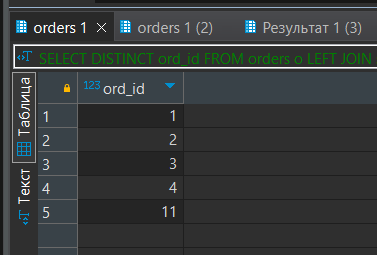
FROM Orders o

JOIN Sales s ON o.ord\_an = s.an\_id

JOIN Groups g ON g.gr\_id = s.an\_group

WHERE LOWER(gr\_name) LIKE '%пылесос%';

Вывод:



Объяснение:

Слово пылесос встречается в группах с gr\_id : 1, 5, 6, 7, 8,

Продажи товаров в этих группах имеют id: 1, 2, 14, 15, 16, 18

Эти продажи с этими позициями есть в заказах (есть заказы, в которых люди решили купить два разных пылесоса): 1, 1, 1, 1, 2, 3, 4, 11

Выбираем уникальные значения и получаем ответ: 1, 2, 3, 4, 11

--Задача 2.

--Вывести даты заказов, название товаров и сумму для всех заказов, которые продавались 04 марта 2024 и всю следующую неделю.

Комментарий: На уточнение, про продавались всю сл. Неделю. Я считаю, что тут можно расценивать двояко: с понедельника до воскресенья или же с понедельника до понедельника? Выбрал, что неделя будет с понедельника по воскресенье, то есть + 6 дней.

Также неясна политика cookies на сайте. То есть можно ли прийти на сайт, заказать пылесос, уйти по делам, на след день вернуться и добавить в заказ компьютер?

Так как это возможно, я считаю допустимой ситуацию, когда человек добавлял в свой заказ товары несколько дней подряд.

В этом случает, дату заказа я считаю последнюю дату, то есть дату последнего добавления продукта в заказ.

WITH tt AS (

SELECT ord\_id, MAX(ord\_datetime) AS final\_date,

STRING\_AGG(an\_name, ', ') WITHIN GROUP (ORDER BY an\_name) AS items,

SUM(an\_amount) AS total\_sum

FROM orders o LEFT JOIN sales s ON o.ord\_an = s.an\_id

GROUP BY ord\_id

HAVING MAX(ord\_datetime) BETWEEN '2024-03-04' AND DATEADD(day, 6, '2024-03-04')

)

SELECT final\_date,

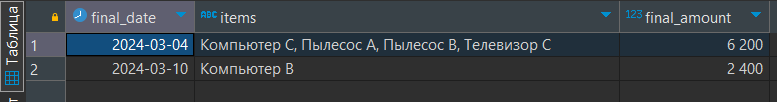
STRING\_AGG(items, ', ') WITHIN GROUP (ORDER BY items) AS items,

SUM(total\_sum) AS final\_amount

FROM tt

GROUP BY final\_date;

Вывод:



Объяснение:

Заказы, которые выполнялись на этой неделе имеют id: 1, 8, 11, 13

В заказ с id = 1 последний товар добавили позже 10-03-2024, соответственно его исключаем

Заказы 11, 13 выполнились 2024-03-04 – агрегируем их товары и общую сумму заказов

Заказ 8 выполнился 2024-03-10, в нем один товар, соответственно и одна цена

--Задача 3.

--Какой год был самый прибыльный по продажам?

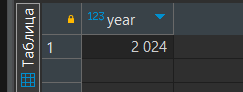
SELECT TOP 1 DATEPART(YEAR, ord\_datetime) AS year

FROM orders o LEFT JOIN sales s ON o.ord\_an = s.an\_id

GROUP BY DATEPART(YEAR, ord\_datetime)

ORDER BY SUM(an\_amount) DESC;

Вывод:



Объяснение:

Про агрегировали сумму по году, отсортировали в порядке убывания, выбрали топ-1

--Задача 4.

--Найдите второй по розничной сумме заказ.

WITH tt AS (

SELECT ord\_id, SUM(an\_amount) AS total\_amount,

DENSE\_RANK() OVER (ORDER BY SUM(an\_amount) DESC) AS rg

FROM Orders o

JOIN Sales s ON o.ord\_an = s.an\_id

GROUP BY ord\_id

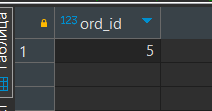
)

SELECT ord\_id

FROM tt

WHERE rg = 2;

Вывод:



Комментарий: не ясно, что делать, если заказы имеют одинаковую сумму?

Так как ничего не сказано, то выведем заказы со второй по величине суммой.

--Задача 5.

/\* Задача 5.

Напишите запрос, который добавляет столбец с позицией каждого заказа на основе общей суммы заказа,

где заказ с самой высокой суммой получает позицию 1. Мы бы ожидали таблицу в таком виде:

Номер заказа | Сумма заказа | Номер позиции \*/

SELECT \*, DENSE\_RANK() OVER (ORDER BY total\_amount DESC) AS rg

FROM (

SELECT ord\_id, SUM(an\_amount) AS total\_amount

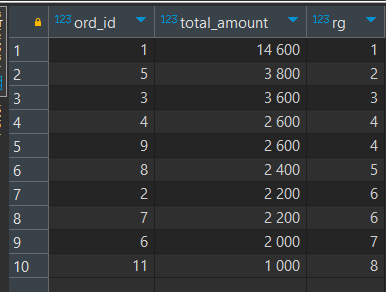
FROM Orders o

JOIN Sales s ON o.ord\_an = s.an\_id

GROUP BY ord\_id

) AS query;

Вывод:



Комментарий тот же, что и к прошлой задаче

Задание на Python

Писал на python

import pandas as pd

from openpyxl import load\_workbook

def test(path\_data: str) -> None:

    """Читает excel-файл,

       обрабатывает данные,

       создает новый файл с обработтаными данными."""

    brend: list[str] = []

    cod: list[int] = []

    item: list[str] = []

    cod\_s: list[str] = []

    cnt: list[int] = []

    price: list[float] = []

    active\_sheet = load\_workbook(path\_data).active

    i = 7

    length = active\_sheet.max\_row

    while i <= length:

        if active\_sheet[f'A{i}'].value == 'Tefal':

            c = 1

            while active\_sheet[f'A{i + c}'].value is None and i + c <= length:

                brend.append('Tefal')

                cod.append(active\_sheet[f'B{i + c}'].value)

                item.append(active\_sheet[f'C{i + c}'].value)

                cod\_s.append(active\_sheet[f'D{i + c}'].value)

                cnt.append(active\_sheet[f'E{i + c}'].value)

                price.append(active\_sheet[f'F{i + c}'].value)

                c += 1

            i += c-1

        i += 1

    df = pd.DataFrame({'Брэнд': brend,

                    'Код': cod,

                    'Товар': item,

                    'КодПроизводителя': cod\_s,

                    'Кол-во': cnt,

                    'Продажа с НДС': price})

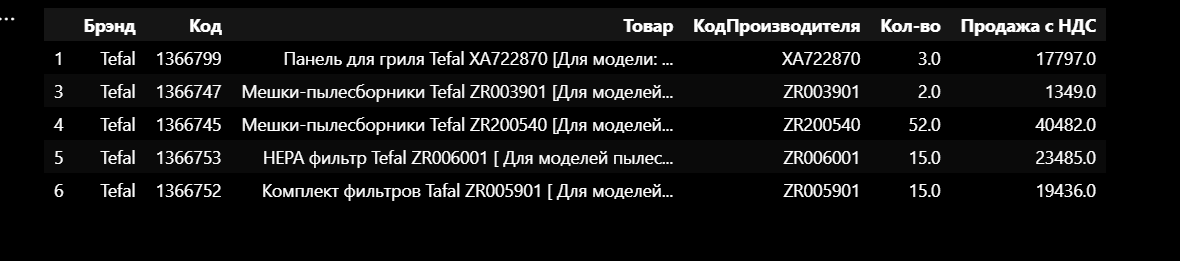
    df.dropna(inplace=True)

    display(df)

    df.to\_csv('Преобразованный файл.csv', index=False)

if \_\_name\_\_ == '\_\_main\_\_':

    test('C:\Dev\stag\_2024\Файл по продажам.xlsx')



Буду благодарен за любую обратную связь,

С уважением, Иван