|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Задача 1.** | | | | | | |  | | |  |  | | |  | | |  |  | | |  |
| Какая часть дохода по каждому контракту приходится на первый день месяца\*?  **CONRACT\_REVENUE** | | | | | | | | | | | | | | | | | | | | | |
|  | |  | | | | |
|  | |  | | | | |  | | |  |  | | |  | | |  |  | | |  |
| **CONTRACT\_ID** | | **EVENT\_DT** | | | | | **REVENUE** | | |  |  | | |  | | |  |  | | |  |
| 101 | | 01.08.2020 | | | | | 200 | | |  |  | | |  | | |  |  | | |  |
| 101 | | 01.09.2020 | | | | | 200 | | |  |  | | |  | | |  |  | | |  |
| 102 | | 01.09.2020 | | | | | 300 | | |  |  | | |  | | |  |  | | |  |
| 103 | | 01.09.2020 | | | | | 200 | | |  |  | | |  | | |  |  | | |  |
| 103 | | 10.09.2020 | | | | | 150 | | |  |  | | |  | | |  |  | | |  |
|  | |  | | | | |  | | |  |  | | |  | | |  |  | | |  |
| CREATE TABLE CONRACT\_REVENUE(  CONTRACT\_ID int,  EVENT\_DT datetime,  REVENUE int  );  INSERT INTO CONRACT\_REVENUE  values  (101, '01.08.2020', 200),  (101, '01.09.2020', 200),  (102, '01.09.2020', 300),  (103, '01.09.2020', 200),  (103, '10.09.2020', 150);  select \* from CONRACT\_REVENUE  where EVENT\_DT like '01.%'; | | | | | |  |  | | |  |  | | |  | | |  |  | | |  |
|  | |  | | | | |  | | |  |  | | |  | | |  |  | | |  |
| **Задача 2.** | |  | | | | |  | | |  |  | | |  | | |  |  | | |  |
| Вывести количество контрактов по каждому продукту на данный момент, если START\_DT это дата начала действия контракта, END\_DT – дата окончания. | | | | | | | | | | | | | | | | | | | | | |
|  | |  | | | | |  | | |  |  | | |  | | |  |  | | |  |
| **CONTRACTS\_PRODUCTS** | | | |  | | | |  | | | |  | | | **PRODUCT** | | | |  | | | |
|  |  | | |  | | | |  | | | |  | | |  | | | |  | | | |
|  |  | | |  | | | |  | | | |  | | |  | | | |  | | | |
| **CONTRACT\_ID** | **PRODUCT\_ID** | | | **START\_DT** | | | | **END\_DT** | | | |  | | | **PRODUCT\_ID** | | | | **PRODUCT\_NAME** | | | |
| 101 | 1 | | | 01.01.2020 | | | | 01.07.2020 | | | |  | | | 1 | | | | o2 Blue S | | | |
| 101 | 3 | | | 01.07.2020 | | | | 01.09.2020 | | | |  | | | 2 | | | | o2 Blue M | | | |
| 101 | 4 | | | 01.09.2020 | | | | 31.12.2099 | | | |  | | | 3 | | | | o2 Free S | | | |
| 102 | 2 | | | 01.08.2020 | | | | 31.12.2099 | | | |  | | | 4 | | | | o2 Free M | | | |
| 103 | 4 | | | 01.01.2020 | | | | 31.12.2099 | | | |  | | | 5 | | | | o2 Free L | | | |
|  |  | | |  | | | |  | | | |  | | |  | | | |  | | | |
| CREATE TABLE CONTRACTS\_PRODUCTS (  CONTRACT\_ID int,  PRODUCT\_ID int,  START\_DT datetime,  END\_DT datetime  );  INSERT INTO CONTRACTS\_PRODUCTS  values  (101, 1, '01.01.2020', '01.07.2020'),  (101, 3, '01.07.2020', '01.09.2020'),  (101, 4, '01.09.2020', '31.12.2099'),  (102, 2, '01.08.2020', '31.12.2099'),  (103, 4, '01.01.2020', '31.12.2099');  CREATE TABLE PRODUCT (  PRODUCT\_ID int,  PRODUCT\_NAME varchar(255)  );  INSERT INTO PRODUCT  values  (1, 'o2 Blue S'),  (2, 'o2 Blue M'),  (3, 'o2 Free S'),  (4, 'o2 Free M'),  (5, 'o2 Free L');  SELECT PRODUCT\_ID,  COUNT(DISTINCT CONTRACT\_ID) as CONTRACTS  from CONTRACTS\_PRODUCTS  where CONTRACT\_ID in (  --активные контракты  SELECT DISTINCT(CONTRACT\_ID) from CONTRACTS\_PRODUCTS  where END\_DT > DATE('now') and START\_DT < DATE('now'))  group by PRODUCT\_ID; | | |  | |  | | | |  | | | |  | | |  | | | |  | | | |
| **Задача 3.** |  | | |  | | | |  | | | |  | | |  | | | |  | | | |
|  |  | | |  | | | |  | | | |  | | |  | | | |  | | | |
| Имеется таблица Sales, структура и пример содержания:   |  |  |  |  | | --- | --- | --- | --- | | **Manager** | **Product** | **Sale** | **Date** | | 1 | 10 | 500 | 10.01.2020 | | 1 | 20 | 300 | 20.01.2020 | | 1 | 30 | 500 | 30.12.2019 | | 2 | 10 | 400 | 10.02.2020 | | 2 | 20 | 500 | 11.02.2020 | | 2 | 30 | 500 | 11.02.2020 | | 3 | 20 | 500 | 30.12.2019 | | 3 | 30 | 300 | 10.01.2020 |     Необходимо написать запрос, выбирающий из этой таблицы для каждого менеджера (manager) одну строку с максимальной суммой продажи (sale), если строк с максимальной суммой продажи для менеджера несколько, то нужно выбрать ту из них, в которой дата продажи (date) больше, если для строк с максимальными суммами продажи и даты продажи одинаковые, то любую из них.    Результат запроса должен содержать четыре колонки как в исходной таблице:   |  |  |  |  | | --- | --- | --- | --- | | **Manager** | **Product** | **Sale** | **Date** | | | | | | | | | | | | | | | | | | | |  | | | |

create TABLE Sales

(

Manager int,

Product int,

Sale int,

Date datetime

);

INSERT INTO Sales VALUES

(1,10,500,'10.01.2020'),

(1, 20, 300, '20.01.2020'),

(1, 30, 500, '30.12.2019'),

(2, 10, 400, '10.02.2020'),

(2, 20, 500, '11.02.2020'),

(2, 30, 500, '11.02.2020'),

(3, 20, 500, '30.12.2019'),

(3, 30, 300, '10.01.2020');

--drop TABLE Rank\_table\_blank;

-- создаем колонку для ранжирования Sales

create temp table Rank\_table\_blank as

select

null as rank,

Manager, Product, Sale, Date

from Sales

order by Sale DESC ;

--drop TABLE Rank\_table;

-- ранжируем Sales по убыванию и заполняем колонку sale\_rank

create temp table Rank\_table as

select

dense\_rank() over (partition by manager order by sale desc) as sale\_rank,

Manager, Product, Sale, Date

from Rank\_table\_blank

window w as (order by Sale DESC)

order by sale\_rank;

-- создаем колонку для ранжирования Date

--drop TABLE Rank\_table\_blank\_2;

create temp table Rank\_table\_blank\_2 as

select

null as rank,

Manager, Product, Sale, Date, sale\_rank

from Rank\_table

order by Sale DESC ;

-- ранжируем Date по убыванию и заполняем колонку date\_rank

--drop TABLE Rank\_table\_2;

create temp table Rank\_table\_2 as

select

dense\_rank() over (partition by manager order by Date(date) desc) as date\_rank,

Manager, Product, Sale, Date, sale\_rank

from Rank\_table\_blank\_2

window w as (order by Date DESC)

order by date\_rank;

--выбираем из минимальных значений рангов и группируем

SELECT Manager, Product, Sale, Date from Rank\_table\_2

where sale\_rank =1 and date\_rank = 1 group by Manager

Комментарии по задачам:

1. В задачи требуется вывести долю, которая приходится на 1 число, тогда как у вас выводится вся информация по контактам на 1 число

2. Информация должна выводится по каждому продукту - в вашей реализации по тем продуктам, у которых их нет, не будет выводится, они затеряются (необходимая доработка: по этим контрактам должно выводится количество 0).

3. Довольно-таки сложно реализовано, можно проще (создавать новые таблицы необходимости нет), но в целом работать будет.

Просьба реализовать задачи в привычном вам инструменте, необязательно на sql, и прислать решение, главное чтобы было правильно.

1)

SQL:

select

CONTRACT\_ID,

round(sum(case when EVENT\_DT

LIKE "01.%" then round(REVENUE) else 0 end) / sum(round(REVENUE)) , 2)

as CONTRACT\_SHARE

FROM CONTRACT\_REVENUE GROUP BY CONTRACT\_ID

Python:

# Создаем DataFrame

ddt = [datetime.datetime.strptime(date, "%d.%m.%Y")

for date in ['01.08.2020', '01.09.2020', '01.09.2020', '01.09.2020', '10.09.2020']]

df = pd.DataFrame({'CONTRACT\_ID':[101, 101, 102, 103,103],

'EVENT\_DT': ddt,

'REVENUE': [200, 200, 300, 200, 150 ]})

# Выбираем начало месяца

df['F\_day'] = df['EVENT\_DT'].astype("str").str.contains("-01")

# функция условия агрегации данных

def calc\_share(rev):

denominator = np.round(rev.REVENUE.sum(), 2)

res = {"Contract\_sum\_first": rev[rev['F\_day'] == True]['REVENUE'].sum(),

"Contract\_sum": rev['REVENUE'].sum()}

return pd.Series(res, index=['Contract\_sum\_first','Contract\_sum'])

# группируем

df = df.groupby('CONTRACT\_ID').apply(calc\_share)

# округляем и выводим доли контрактов на первое число месяца

df['Share\_Contract\_f\_day'] = np.round(df['Contract\_sum\_first'] / df['Contract\_sum'], 2)

df[['Share\_Contract\_f\_day']]

2)

SQL:

-- создаем объединяющую временную таблицу

drop TABLE ProductContract;

create temp table ProductContract as

select \* from (

select \* from PRODUCT) t1

left join

(select \* from CONTRACTS\_PRODUCTS) t2

on t1.PRODUCT\_ID=t2.PRODUCT\_ID;

-- выбираем только актуальные контракты на текущий момент

select \*,

case when END\_DT > DATE('now') and START\_DT < DATE('now') then 1 else 0 END

as is\_work from ProductContract

Python:

# Создаем DataFrame

START\_DT = [datetime.datetime.strptime(date, "%d.%m.%Y")

for date in ['01.01.2020', '01.07.2020', '01.09.2020', '01.08.2020', '01.01.2020']]

END\_DT = [datetime.datetime.strptime(date, "%d.%m.%Y")

for date in ['01.07.2020','01.09.2020', '31.12.2099', '31.12.2099', '31.12.2099']]

df\_CONTRACTS\_PRODUCTS = pd.DataFrame({'CONTRACT\_ID':['101', '101', '101', '102','103'],

'PRODUCT\_ID': [1, 3, 4, 2, 4],

'START\_DT': START\_DT,

'END\_DT': END\_DT})

df\_PRODUCT = pd.DataFrame({'PRODUCT\_ID': [1, 2,3,4,5],

'PRODUCT\_NAME': ['o2 Blue S','o2 Blue M','o2 Free S','o2 Free M','o2 Free L']})

# объединяем обе таблицы и создаем колонку булевыми значениями под актуальные контракты

today = pd.to\_datetime(datetime.date.today())

filter\_date = (pd.to\_datetime(df\_CONTRACTS\_PRODUCTS['START\_DT']) < today) & \

(pd.to\_datetime(df\_CONTRACTS\_PRODUCTS['END\_DT']) > today)

df\_CONTRACTS\_PRODUCTS['is\_works'] = filter\_date

df\_tmp = pd.merge(df\_CONTRACTS\_PRODUCTS, df\_PRODUCT, on='PRODUCT\_ID', how='outer').fillna(False)

df\_tmp2= df\_tmp[['CONTRACT\_ID','PRODUCT\_ID','is\_works', 'PRODUCT\_NAME']]

# функция условия агрегации данных

def is\_available(table):

res = {'CONTRACTS': table[table['is\_works'] == True]['CONTRACT\_ID'].count(),

'CONTRACT\_ID': table[table['is\_works'] == True]['CONTRACT\_ID'].apply(lambda x: x + ', ').sum(),

'PRODUCT\_NAME': table[table['is\_works'] == True]['PRODUCT\_NAME'].apply(lambda x: x + ', ').sum()}

return pd.Series(res, index=['CONTRACT\_ID','CONTRACTS', 'PRODUCT\_NAME'])

# группируем и применяем функцию

df\_res = df\_tmp2.groupby('PRODUCT\_ID').apply(is\_available)

df\_res['CONTRACT\_ID'] = df\_res['CONTRACT\_ID'].str.rstrip(' ,')

df\_res['PRODUCT\_NAME'] = df\_res['PRODUCT\_NAME'].str.rstrip(' ,')

3)

Python:

# Создаем DataFrame

ddt = [datetime.datetime.strptime(date, "%d.%m.%Y")

for date in

['10.01.2020', '20.01.2020', '30.12.2019', '10.02.2020', '11.02.2020', '11.02.2020', '30.12.2019', '10.01.2020']]

df = pd.DataFrame({'Manager':[1, 1, 1, 2, 2, 2, 3, 3],

'Product': [10, 20, 30, 10, 20, 30, 20, 30],

'Sale': [500, 300, 500, 400, 500, 500, 500, 300],

'Date': ddt})

#Сортируем по 2 полям из условия в порядке убывания и оставляем только уникальных менеджеров (первую встречу в датасете)

df.sort\_values(by=['Sale', 'Date'],ascending=False, inplace=True)

df.drop\_duplicates(subset=['Manager'], keep='first')