Отчет по практическому заданию №3

Команда:

1. Раева Анастасия
2. Богданова Елена
3. Чистяков Иван

Описание задания.

Applepen - это большая торговая сеть, которая занимается продажей всего двух продуктов: яблок и карандашей. Ее магазины расположены в различных уголках Соединенных Штатов и более 10 лет обслуживают покупателей.

Недавно топ-менеджмент компании решил более активно использовать имеющиеся у них данные в принятии решений. Каждый магазин собирает информацию о:

1. закупках (поставки яблок и карандашей два раза в месяц),

2. продажах (лог транзакций, по записи на каждую проданную позицию),

3. инвентарь (месячные данные общего количества яблок и карандашей на складе).Данные доступны в формате CSV. Внутри файла данные отсортированы по дате.

Данные доступны в формате CSV. Внутри файла данные отсортированы по дате. К сожалению, данные никогда не консолидировались и не проверялись.

Нам необходимо получить следующие данные в CSV-файлах

1 Состояние склада на каждый день

Данные о состоянии склада в конце каждого дня после того как все поставки и продажи были совершены. Подобная информация будет очень ценна менеджерам магазинов. Состояние склада должно строится на основе месячных данных об инвентаре. Известно, что люди воруют из магазинов, но сейчас нет никакой возможности узнать объем сворованного товара

2. Месячные данные о количестве сворованоого товара.

3. Агрегированные данные об объемах продаж и количестве сворованной продукции по штату и городу.

Подход к решению.

Структура названия входных файлов:

‘MS-<letter><number>-<type>.csv’

MS – штат, в которм расположен магазин

letter может принимать значения: b, m, s

number – 1,2,3,4,5

type – sell, supply, inventory

1 Состояние склада на каждый день:

Данные в файле о продажах по конкретному магазину записаны в формате: дата, лог транзакции. Если лог содержит ‘ap’, то он соответствует продаже яблока, если ‘pen’, то продаже ручки. Формируем таблицу с информацией о количестве проданных яблок и ручек в каждый день месяца. Для этой цели используется функция:

“sell_dataframe_change(df_sell)”

Чтобы найти состояние склада на каждый день, необходимо сложить количество товара на склада в начале дня с поставленным и вычесть проданный в этот день.

Поставки происходят два раза в месяц (01 и 15 числа).

Построение таблицы с данными о состоянии склада на каждый день реализуется функцией:

“daily_inv(df_sell, df_supply)”.

Пример части таблицы, полученной на выходе:

date	apple	pen
2006-01-01	33271	2574
2006-01-02	31409	2431
2006-01-03	29529	2260
2006-01-04	27732	2107
2006-01-05	25790	1974
2006-01-06	23892	1825
2006-01-07	21997	1687

(данные представленны для магазина MS-b1)

2. Месячные данные о количестве сворованого товара.

Данные о состоянии склада на каждый день, полученные в предыдущем пункте, составлены без учета возможных краж продукции.

В конце каждого месяца в магазинах проходит инвентаризация.

Чтобы найти количество сворованного, из данных о складе в конце месяца нужно вычесть данные инвентаризации. В результате получится таблица с количеством краж, в которой последней дате месяца соответсвует объем украденного за этот месяц и все предыдущие. Поэтому далее вычитаем из данных о кражах за текущей месяц кражи за предыдущие. Алгоритм реализован в функции:

“month_steal(df_inv, df_daily_inv)”

Пример части таблицы, полученной на выходе:

date	apple	pen
2006-01-31	10.0	11.0
2006-02-28	6.0	6.0
2006-03-31	7.0	6.0
2006-04-30	6.0	14.0
2006-05-31	8.0	1.0
2006-06-30	7.0	9.0
2006-07-31	10.0	12.0

(данные представленны для магазина MS-b1)

3. Агрегированные данные об объемах продаж и количестве сворованной продукции по штату и городу:

Для заполнения агрегированной таблицы используются таблицы о продажах и об украденном товаре, полученные в предыдущих пунктах.

Полученная таблица:

year	state	apple_sold	apple_stollen	pen_sold	pen_stollen
2006-12-31	MS	2152006	418.0	155633	461.0
2007-12-31	MS	2150384	377.0	154730	346.0
2008-12-31	MS	2163559	383.0	154597	382.0
2009-12-31	MS	2152502	433.0	155409	454.0
2010-12-31	MS	2149787	418.0	155523	441.0
2011-12-31	MS	2154860	436.0	154158	452.0
2012-12-31	MS	2160040	381.0	155798	421.0
2013-12-31	MS	2157901	361.0	154496	444.0
2014-12-31	MS	2153434	433.0	154687	441.0
2015-12-31	MS	2152497	370.0	153562	395.0

Пинцип работы программы и описание функций.

В df_sell считываем данные файла о продажах, в df_supply – данные о поставках, в df_inv – данные инвентаризации.

Индекс df_sell – дата в формате(2006-01-01)

sell_dataframe_change(df_sell): лог транзакции(столбец sku_num) заменяем на 1, если соответствует продаже яблока, иначе – 0

в df_sell столбец ‘sku_num’ переименовываем в ‘apple’,

добавляем столбец ‘pen’, заносим в строку 1, если в ‘sku_num’ этому индексу соответствовал 0.

Группируем по дате и суммируем по яблокам и

ручкам:

```
df_sell = df_sell.groupby(['date'])['apple', 'pen'].sum()
```

daily_inv(df_sell, df_supply): создается DataFrame df_daiily_inv. Изначально содержит количество яблок и ручек с обратным знаком, информация на каждый день месяца. Далее соединяем с таблицей о поставках(они происходят 01 и 15 числа каждого месяца):

```
df_daily_inv = pd.concat([df_daily_inv,df_supply])
```

группируем содержимое дата фрейма по дате и суммирум:

```
df_daily_inv = df_daily_inv.groupby(['date']).sum()
```

так как поставки происходят не каждый день, используем сумму с накоплением:

```
df_daily_inv['apple'] = df_daily_inv['apple'].cumsum()
```

month_steal(df_inv, df_daily_inv): создается дата фрейм steal

```
steal = pd.DataFrame(columns = ['ap','pe'])
```

Изначально в столбце ‘ap’ количество яблок на складе после инвентаризации со знаком минус.

Индекс – даты, последний день месяца. Индекс df_daily_inv – дата, каждый день месяца.

Поэтому соединяем таблицы с пересечением по индексу:

```
steal = pd.concat([steal, df_daily_inv], axis = 1, join = 'inner')
```

В полученном дата фрейме суммируем значения столбцов ‘ap ‘apple’, ‘pen’ ‘pe’.

Сдвигаем на одно значение, первое заполняем нулем и полученное вычетаем из исходного:

```
steal['apple'] = steal['apple']-steal['apple'].shift(1).fillna(0)
steal['pen'] = steal['pen']-steal['pen'].shift(1).fillna(0)
```

Чтобы сгруппировать данные по году:

индекс – дата в формате “2006-01-01”, с помощью pandas.to_datetime преобразуем индекс из строкового формата в datatime формат. Далее используем resample('Y') чтобы сгрупировать данные по году, одновременно суммируем значения столбцов ‘apple’ ‘pen’

piv_t – дата фрейм, соответствующий сводной таблице.

create_pivot_table(piv_t,df_sell, steal): заполняет piv_t данными по первому обработанному магазину

upd_piv_tab(piv_t,df_sell, steal): обносляет данные piv_t по мере обрабатывания новых магазинов.

Вклад участников команды.

Все этапы выполнения задания были проделаны совместно.

Набор входных и выходных данных для тестирования по ссылке:

https://console.cloud.google.com/storage/browser/artem-pyanykh-cmc-prac-task3-seed17