Case Assignment - Data Engineer

This case consists of 3 separate problems that we would like you to consider.

To successfully complete this assignment please provide us with:

- The SQL queries you developed and the corresponding result set of those queries (csv) for Problem 1 - Balance of Inventory
- The Python program you developed for Problem 2 - String Manipulation
- The program you developed for Problem 3 - Nested JSON analysis

Please feel free to use any medium you feel comfortable with to share the code snippets and your answers to the additional questions (email, Docs, Notebooks, pptx...).

You should not have to spend more than three hours on this assignment.
You can find the provided data in the data folder of this repository.

Problem Statement 1 - Balance of Inventory

You have an inventory movement dataset that shows item transactions for each item, warehouse location and day.

Row	ItemID	LocationID	TransactionDate	TransferQuantity
1	a0W5700000OaJ2cEAF	a0XD0000005JVGVMA4	2020-05-26	-1.0
2	a0W5700000MYVQOEA5	a0XD000000SKq1hMAD	2020-05-26	2.0
3	a0W5700000MYVQOEA5	a0X5700001C7jz6EAB	2020-05-26	-1.0
4	a0W5700000OaJ2cEAF	a0XD0000005JVGQMA4	2020-05-26	-4.0
5	a0W5700000OaJ2cEAF	a0XD000000D3A2WMAV	2020-05-25	1.0
6	a0W5700000OaJ2cEAF	a0XD0000005JVGQMA4	2020-05-25	-11.0
7	a0W5700000OaJ2cEAF	a0XD000000588bmMAA	2020-05-25	-1.0

The friendly business user would like to see the balance of inventory for each item and warehouse location for every single day since the first transaction date and has asked for your help.

1. How would you write the SQL query that would transform the data into a format that would allow the business user to easily answer their question?
    a. Can you think of an alternative way to organize your data pipeline that would allow you to provide suitable data format?
    b. Would your solution be impacted if your source dataset would suddenly be loaded incrementally?
2. Once you have your new dataset how would you write a SQL query on top of it that would return:
    a. The current inventory balance
    b. 30 day moving average of the inventory balance

Problem Statement 2 - String manipulation for ETL pipeline

The business wants to migrate fields and tables from a legacy database to a new data warehouse. However, the legacy database contains many redundant and unnecessary fields.

In order to determine which fields are useful and need to be migrated, you must extract there from .sql files built for the legacy database.

There are many such .sql files but for the scope of this challenge you have only been provided with 5.

Write a Python program with appropriate modularity and tests to open up each of these .sql files, parse their contents and extract from them the field and table names they used in the legacy database. These .sql files only use basic SELECT FROM WHERE statements (i.e. they were not created with GROUP BY, CASE WHEN, HAVING, etc. clauses).

For example, in the following .sql file (base_direct_sales1.sql) the "useful" columns to the business from the "direct_sales" are "id", "sales", "orders" and "purchases". Dbt macros were used throughout these files. They work as functions and can be identified by the double curly brackets.

```
select {{ null_if ('id' }}
  ,sales as  number_of_sales
  ,orders
  ,timestamp(current_date()) as order_date
  ,purchases as number_of_purchases
from {{source('salesforce', direct_sales)}}
```

Store the table names and their associated fields in a format which can later be read programmatically by an ETL pipeline which will migrate this data from the legacy database to the data warehouse.

The .sql files will all be contained within the same folder with the following relative path from your Python script, "../sql_scripts/".

Problem Statement 3 - Nested JSON from External API and Analysis

Carry out the following steps to download, manipulate and analyze the dataset on restaurant food violations found at https://health.data.ny.gov/Health/Food-Service-Establishment-Last-Inspection/cnih-y5dw.

1. Given the following URL, get the nested JSON using the website's REST API https://health.data.ny.gov/api/views/cnih-y5dw/rows.json?accessType=DOWNLOAD
2. Print the nested schema of this dataset
3. Programmatically convert this dataset into a dataframe with correct column names.
4. Output the yearly non-critical violation numbers for the facility which has, for more than one year, had the highest number yearly non-critical violations.
5. Create a visualization of the data from step 4.