Background

Walmart is the biggest retail store in the United States. Just like others, they have been expanding their e-commerce part of the business. By the end of 2022, e-commerce represented a roaring $80 billion in sales, which is 13% of total sales. One of the main factors that affects their sales is public holidays, like the Super Bowl, Labour Day, Thanksgiving, and Christmas.

In this project, you have been tasked with creating a data pipeline for the analysis of supply and demand around the holidays, along with conducting a preliminary analysis of the data. You will be working with two data sources: grocery sales and complementary data. You have been provided with the `grocery_sales` table in `PostgreSQL` database with the following features:

`grocery_sales`

- `"index"` - unique ID of the row
- `"Store_ID"` - the store number
- `"Date"` - the week of sales
- `"Weekly_Sales"` - sales for the given store

Also, you have the `extra_data.parquet` file that contains complementary data:

`extra_data.parquet`

- `"IsHoliday"` - Whether the week contains a public holiday - 1 if yes, 0 if no.
- `"Temperature"` - Temperature on the day of sale
- `"Fuel_Price"` - Cost of fuel in the region
- `"CPI"` – Prevailing consumer price index
- `"Unemployment"` - The prevailing unemployment rate
- `"MarkDown1"`, `"MarkDown2"`, `"MarkDown3"`, `"MarkDown4"` - number of promotional markdowns
- `"Dept"` - Department Number in each store
- `"Size"` - size of the store
- `"Type"` - type of the store (depends on `Size` column)

You will need to merge those files and perform some data manipulations. The transformed DataFrame can then be stored as the `clean_data` variable containing the following columns:

- `"Store_ID"`
- `"Month"`

- "Dept"
- "IsHoliday"
- "Weekly_Sales"
- "CPI"
- ""Unemployment""

After merging and cleaning the data, you will have to analyze monthly sales of Walmart and store the results of your analysis as the agg_data variable that should look like:

Month	**Weekly_Sales**
1.0	33174.178494
2.0	34333.326579
...	...

Finally, you should save the clean_data and agg_data as the csv files.

It is recommended to use pandas for this project.

Instructions

Build a data pipeline using custom functions to extracts, transforms, aggregates, and loads e-commerce data.

1. In the cell `grocery_sales`, use SQL to query all the data from the `grocery_sales` table.

2. Create a function called `extract()` with two arguments, combining the `grocery_sales` table and the `extra_data.parquet` file, returning a variable called `merged_df` containing data from both sources.

3. Implement a function named `transform()` with one argument, taking `merged_df` as input, filling missing numerical values (using any method of your choice), adding a column `"Month"`, keeping the rows where the weekly sales are over $10,000 and drops the unnecessary columns. Ultimately, it should return a DataFrame and be stored as the `clean_data` variable.

4. Define a function called `avg_monthly_sales()` that takes `clean_data` as input and returns an aggregated DataFrame containing two columns - `"Month"` and `"Avg_Sales"` (rounded to 2 decimals). You should call the function and store the results as a variable called `agg_data`.

5. Create a function called `load()` that takes the cleaned and aggregated DataFrames, and their paths, and saves them as `clean_data.csv` and `agg_data.csv` respectively, without an index.

6. Lastly, define a `validation()` function that checks whether the two `csv` files from the `load()` exist in the current working directory.

Note

As it is not possible to rebuild a server on github.
Assume the data from database is imported by the code below.

```
from sqlalchemy import create_engine
import pandas as pd
engine = create_engine('sqlite://Example.sqlite')
con = engine.connect()
rs = con.execute(select * from grocery_sales )

grocery_sales = pd.DataFrame(rs.fetchall())
con.close()
```