Jane

Take-Home SQL Test

James Parkington

In this I would like to begin by expressing my sincere gratitude to the interview team at Jane Technologies for providing me with the opportunity to work on this project. While the prompt initially called for a set of abstract queries based on the given schema, I chose to take this challenge a step further. By dedicating extra time and effort to create something not only useful in the short term but valuable in the long run, my aim was to demonstrate my diverse skill set and my ability to intimately understand and recreate business data.

My overarching approach was to develop an open-source solution that Jane Technologies can use as they see fit. Tied to this project is a GitHub repository that contains well-documented files and modules that could be adapted for various purposes, including potentially for future interviews. I was mindful of using systems, languages, and modules that would be accessible to most analysts, regardless of their preferred operating system.

In the repo, you will find not only the requested SQL queries but also a series of Python classes and utilities designed to generate plausible datasets based on the supplied schemas. Each class has been created with careful consideration given to data relationships and constraints of the data domain they occupy. The various utility functions further support these classes, facilitating tasks such as data cleaning, conversion, and saving.

I've also developed a range of data visualizations using **`matplotlib`**, **`pyplot`**, and **`scikit-learn`**. These analyses vary from straightforward representations of the requested queries to more advanced regression and segmentation analysis. By incorporating these visualizations, I wanted to emphasize my analytical skills and my capability to leverage data for decision-making against common business questions.

In this report, I will walk you through the various aspects of the project, explaining my thought process, the tools and technologies used, and the overall structure of the repository. My hope is that this document will provide a clear and concise overview of the work I have completed, while also offering some insight into my personal approach to problem-solving and data analysis.

Repository Structure 2

Analysis 2

Data 3

Python Packages 3

Repository Structure

Below is a breakdown of the directories in the Jane repo, with short explanations for each of the objects within. The link for each object has been supplied below, as well.

Each of these objects is available for download and redistribution if need be, and I'm happy to supply any of them in a different format, if you'd prefer.

Analysis

Start here if you're looking for the requested SQL queries for the 5 prompts in the test.
Contains Jupyter notebooks with those requested queries, as well as classes and utilities for data generation and analysis

`1: School Attendance Records.ipynb` (*Link*)

- Uses the **`SchoolAttendance`** and **`Students`** classes to generate random datasets as DataFrames, SQL tables, and CSV files.
- Provides queries related to attendance, students, and birthdays.
- Includes visuals for GPA and forecasting attendance based on the sample data.

`2: Customer Orders.ipynb` (*Link*)

- Uses the **`OrderHistory`** and **`CustomerInfo`** classes to generate random datasets as DataFrames, SQL tables, and CSV files.
- Offers queries related to customers, orders, and sales.
- Includes a series of scatterplots, polygon maps, and regressions you'd expect to see in an eCommerce business setting

Classes

A subdirectory of *Analysis* that contains Python files for generating random datasets based on different schemas. The **`Utilities.py`** file provides utility functions for data generation, conversion, file path construction, and more.

- **`CustomerInfo.py`** (*Link*)
  - Generates a randomized dataset of customer information.
  - Provides data with geographic constraints and credit limit considerations.
  - Uses data validation techniques to ensure consistency and accuracy.
- **`OrderHistory.py`** (*Link*)
  - Generates a randomized dataset of order history information.
  - Takes into account shipping dates, customer ordering patterns, expected holidays, and order frequency for typical eCommerce customers
  - Utilizes random sampling to generate plausible order amounts, customer distributions, and statuses.
- **`SchoolAttendance.py`** (*Link*)

    - Generates a randomized dataset of attendance records for students within a specific date range.

- **`Students.py`** ([Link]())
    - Generates a randomized dataset of student profiles.
    - Considers grade level, birth year, and school district associations.

- **`Utilities.py`** ([Link]())
    - Provides utility functions for data generation, DataFrame and CSV creation, file path construction, and more.
    - Contains functions for loading the **`ipython-sql`** extension, SQLite database connection, and customizing **`pyplot`** and **`Basemap`** visualizations.

Data

Holds the SQLite database and CSV exports of the most recent notebook run of each table, for reference.

- **`jane.db`**
    - A SQLite3 database that has tables for each of the classes above, which automatically gets updated whenever a class is called through the **`__call__`** method
- **`customer_info.csv`** (*generated by **`CustomerInfo.py`***)
- **`order_history.csv`** (*generated by **`OrderHistory.py`***)
- **`school_attendance.csv`** (*generated by **`SchoolAttendance.py`***)
- **`students.csv`** (*generated by **`Students.py`***)

Resources contains shape files and metadata needed for the US states analyses, as well as the original prompt.

Python Packages

The followed Python packages were used within the classes, utilities, and Jupyter notebooks in this project. They offer a wide range of capabilities, including advanced numerical computations, data visualization, regression analysis, and more. These packages make it possible to efficiently perform complex data manipulations and generate high-quality visuals to help analyze and communicate the results.

- **`numpy`**: Used for numerical computations and generating random values for the datasets.
- **`pandas`**: A powerful library for handling data manipulation and analysis, used extensively throughout the program for creating and managing DataFrames.
- **`pyplot`**: A part of the **`matplotlib`** library that offers a simplified interface for creating visualizations.
- **`Basemap`**: Provides a toolkit for creating geographic maps and visualizations within the program.
- **`seaborn`**: A statistical data visualization library based on **`matplotlib`**, used for creating more visually appealing plots within the program.
- **`sklearn.linear_model`**: Utilized for Ridge regression, a machine learning technique used for polynomial regression in the program.

- **`sklearn.preprocessing`**: Provides **`PolynomialFeatures`**, which is used for transforming data for polynomial regression and normalization in the program.
- **`scipy.optimize`**: Used for curve fitting in the program to help with data analysis and predictions.
- **`faker`**: A library for generating fake data, such as names, addresses, and phone numbers, which is used to create the customer dataset.
- **`uszipcode`**: Provides functionality to search for US zip codes and their corresponding geographic information, ensuring the generated customer data has accurate zip codes, cities, and states.
- **`holidays`**: A library for handling holidays within the context of order records, allowing the program to consider holidays when generating data.
- **`iPython`**: Used for loading the **`ipython-sql`** extension and running SQL queries within the Jupyter notebooks.
- **`os`**: A standard library for interacting with the operating system, used for file path construction and management.
- **`sqlite3`**: A library for working with SQLite databases, used for storing the generated datasets as SQL tables.

In order to run the program in your environment, please ensure these packages are installed within your Python instance.