Data Analysis Final Project

Software Requirement Specifications

by Balázs Boldogh

In this project I am creating calculations and visualizations based on a dataset found on Kaggle. This dataset contains comprehensive business data from the automotive sector in 2023. In this project my main goal is to create interactable dashboards using PowerBI, as well as demonstrate the data processing capabilities of Python and the connectivity between SQL and PowerBI to create visualizations directly from live databases.

Software requirements:

- The Python script must to be able to upload the contents of the CSV files into an SQL database.
- The Python script also must be able to create calculated fields for further analysis.
- The SQL Database must contain all the original data, their keys and relations.
- The PowerBI project must take the data from the SQL database and create interactable visualizations which are easy to navigate and visually appealing to the user.
- An extra document must be created detailing the findings and interpretation of the visualizations created using PowerBI

Activities by day (2023.11.13 – 2023.11.17):

- **Day1** – Find datasets, Create ER Model, Create this SRS Document
- **Day2** – Use Python on the datasets to create calculated fields to add all the data to an SQL database
- **Day 3** – Connect the SQL Database to PowerBI and create visualizations
- **Day 4** – Document the visualizations and the findings in the data, send the project for feedback
- **Day 5** – Make adjustments to the project based on the feedback and submit for final approval

Dataset:

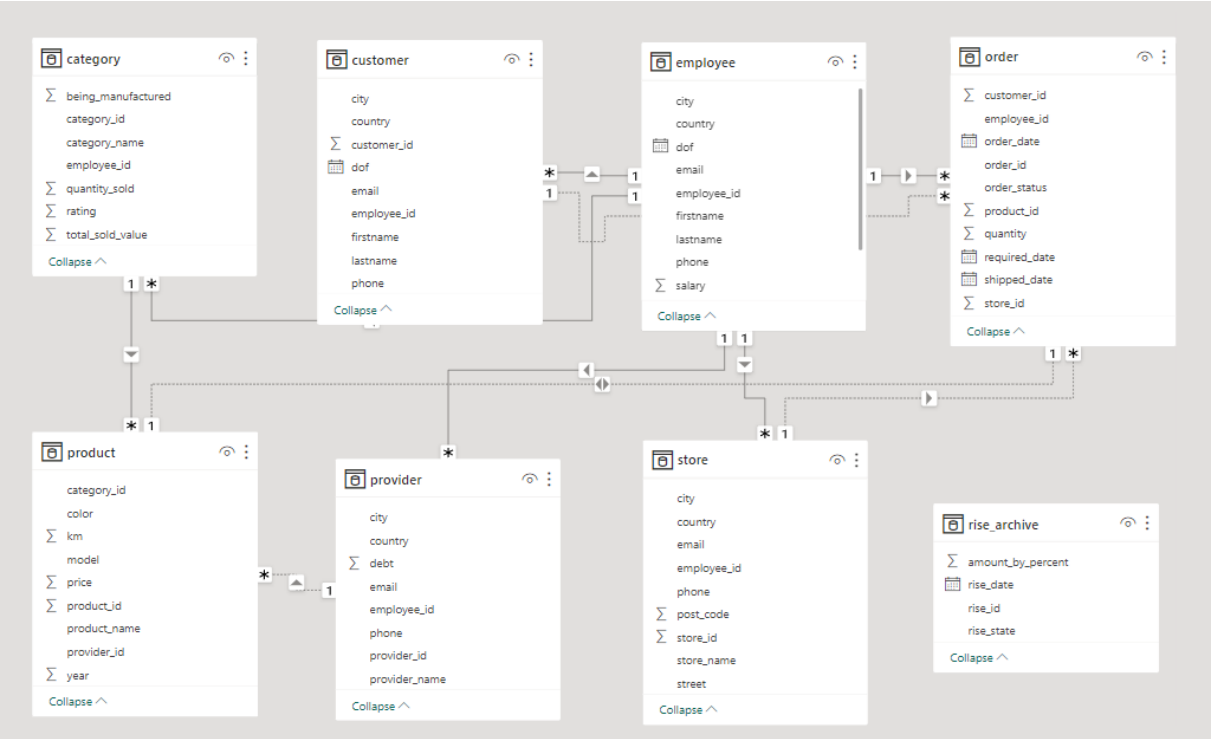
The dataset titled **"Automotive Sector Comprehensive Business Data-2023"** is publicly available on this link: https://www.kaggle.com/datasets/grncode/automotive-sector-compherensive-business-data-2023?select=category.csv

The dataset contains 8 CSV files which are the following:

- category
- customer
- employee
- order
- product
- provider
- rise_archive
- store

These CSV files will each correspond to a table in the SQL Database. They all contain unique identifiers as well as foreign keys, so they can be used to create an Entity-Relationship model.

ER Model of the Automotive Dataset:



Connections in the model:

- category is connected to employee on employee_id (Many to One)
- category is connected to product on category_id (One to Many)
- customer is connected to employee on employee_id (Many to One)
- customer is connected to order on customer_id (One to Many)
- employee is connected to order on employee_id (One to Many)
- product is connected to provider on product_id (Many to One)
- provider is connected to employee on employee_id (Many to One)
- store is connected to order on store_id (One to Many)
- store is connected to employee on employee_id (Many to One)
- rise_archive is the only table without a relationship to any other table

Visualization Plans*:

1. Overview Dashboard:

- Create a dashboard that provides an overview of key metrics for the automotive sector.
- Include summary cards for total customers, employees, orders, products, providers, and stores.
- Use KPI visuals to show important indicators such as revenue, average order value, etc.

2. Category Distribution:

- Visualize the distribution of products across different categories using a treemap or a bar chart.
- Use a donut chart to represent the proportion of products in each category.

3. Customer Demographics:

- Create a customer demographics dashboard using bar charts or pie charts to show the distribution of customers based on factors like age, gender, and location.

4. Employee Performance:

- Visualize employee performance metrics, such as sales per employee, using a bar or column chart.
- Implement a line chart to track employee performance over time.

5. Order Analysis:

- Use a line chart or area chart to show the trend of orders over time.
- Create a heatmap to analyze order frequency based on different time periods.

6. Product Sales and Revenue:

- Visualize product sales using a bar or column chart.
- Implement a stacked bar chart to show the contribution of each product to total revenue.

7. Provider Analysis:

- Display a map visual to show the geographic distribution of providers.
- Use a scatter plot to visualize the relationship between provider ratings and the number of products supplied.

8. Store Performance:

- Create a dashboard that shows key metrics for each store, such as sales, customer satisfaction, and product variety.
- Use a table or matrix to display store-wise performance.

9. Rise Archive Insights:

- Explore historical data using time-series visualizations to identify trends or patterns.
- Use a line chart to visualize the rise in various aspects over time.

10. Cross-Category Analysis:

- Implement a stacked column chart to show the distribution of product categories within each order.
- Use a matrix or table to analyze which categories are frequently bought together.

11. Customer Segmentation:

- Utilize clustering algorithms to segment customers based on their behavior and preferences.
- Create visuals that represent each customer segment's characteristics.

12. Employee and Customer Interaction:

- Build a visual that shows the interaction between employees and customers, such as the number of interactions per employee.

***Please note, that these are only notes for potential visualizations. Actual visualizations may differ based on experimentation during work on the project.**

Development software used:

- MySQL Workbench (MySQL)
- Visual Studio Code (Python)
- Microsoft PowerBI
- Tableau

Python libraries and dependencies:

1. **pandas:** This library is excellent for data manipulation and analysis. It can handle reading CSV files into DataFrames, which can be easily manipulated and then inserted into a MySQL database.
2. **sqlalchemy:** This is a powerful and SQL-agnostic database toolkit. It allows you to interact with your MySQL database in a Python.
3. **mysql-connector-python:** A MySQL driver for Python, which is required for connecting to the MySQL database using SQLAlchemy.
4. **numpy:** If you need to perform numerical operations on your data before inserting it into the database, you might find numpy useful.
5. **csv:** This is a built-in Python module, so you don't need to install it separately. It will help you with basic CSV file handling.
6. **os:** Another built-in module, useful for interacting with the operating system, such as listing files in a directory.
7. **glob:** Also a built-in module, useful for matching file paths using patterns.