1. Title and Project Overview:

Title: Brazilian E-Commerce Public Dataset by Olist

Project Summary:

This project analyzes an e-commerce dataset containing order details, customer demographics, seller performance, and product reviews to identify trends, customer preferences, and areas for improvement.

2. Objectives:

Top-Selling Products: Focus on marketing and stocking high-demand products.

Profitable Periods: Target promotions and campaigns during peak sales periods.

Customer Behavior: Analyze and enhance loyalty strategies by comparing repeat vs. new customer behavior.

Geographic Insights: Direct resources to high-sales regions and explore expansion opportunities.

3. Tools and Technologies Used

Tools: Open Refine, and Excel

Programming Languages: Python (Pandas, NumPy, Matplotlib, Seaborn, ARIMA, autocorrelation plot)

Database / Storage: SQL

Visualization: Tableau dashboards

4. Data Description

Source: Kaggle website: https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce

Structure:

Table Name	Number of Rows	Number of Columns	Description
customers	99441 rows	5 columns	Contains details about customers, including unique IDs and demographic information.
geolocation	738332 rows	5 columns	Includes geographic data such as latitude, longitude, and region mappings.
order items	112650 rows	7 columns	Details of individual items in orders, including product IDs and quantities.
payments	103886 rows	5 columns	Payment-related data, such as payment type and transaction amounts.

reviews	9839 rows	7 columns	Product review data, including review scores and comments.
orders	96461 rows	8 columns	Information about orders, including order IDs, timestamps, and statuses.
products	32951 rows	9 columns	Product data, including category names, descriptions, and prices.
sellers	3095 rows	4 columns	Details about sellers, including their locations and unique IDs.

5. Data Cleaning:

- Handle duplicated id in orders table using Excel to be unique values

- Handle the duplicates and missing values using the Pandas library as an example

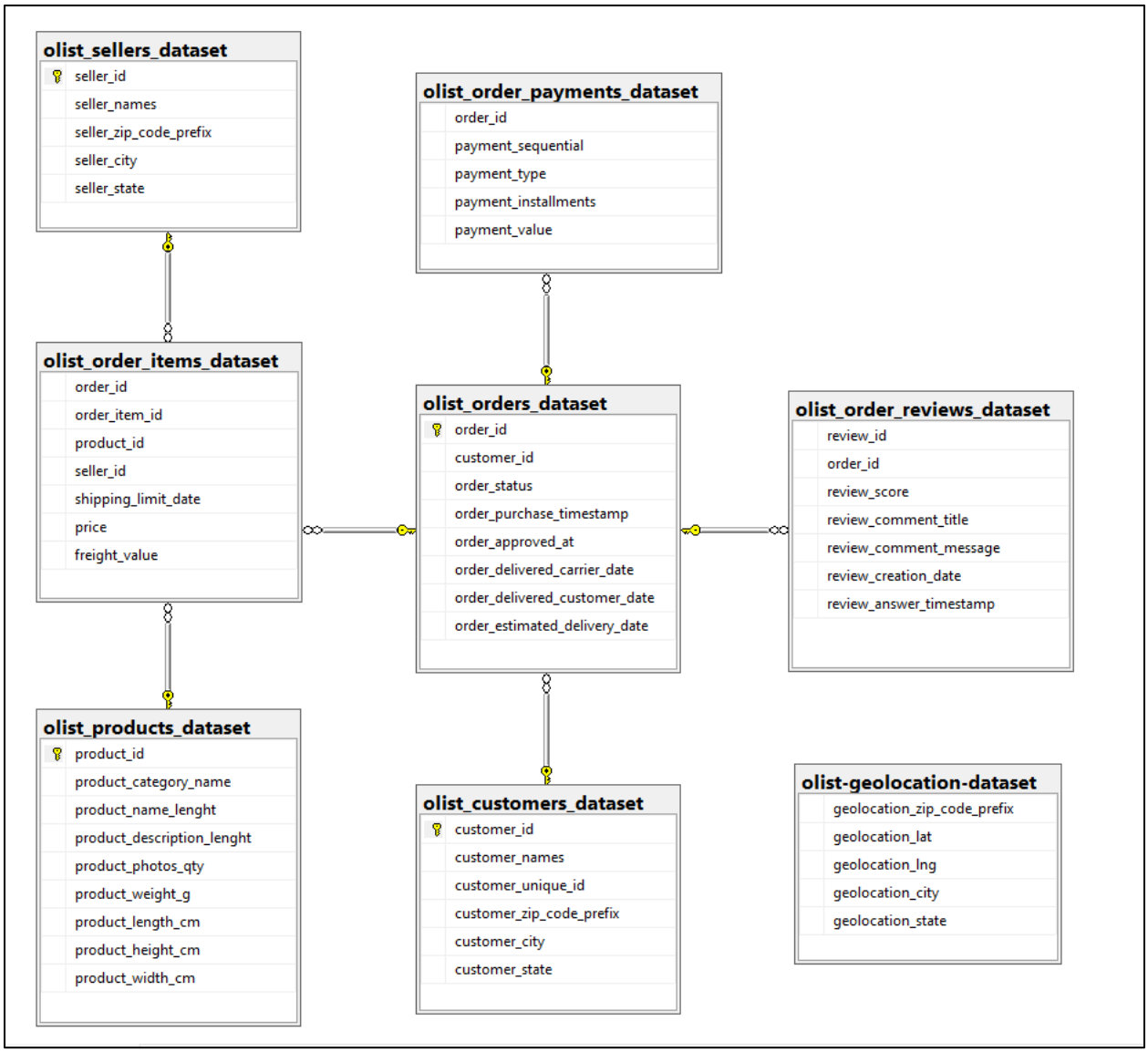
```
customers.drop_duplicates(inplace = True)
geolocation.drop_duplicates(inplace = True)
order_items.drop_duplicates(inplace = True)
payments.drop_duplicates(inplace = True)
reviews.drop_duplicates(inplace = True)
orders.drop_duplicates(inplace = True)
products.drop_duplicates(inplace = True)
sellers.drop_duplicates(inplace = True)
```

```
customers = customers.dropna()
geolocation = geolocation.dropna()
order_items = order_items.dropna()
payments = payments.dropna()
reviews = reviews.dropna()
orders = orders.dropna()
products = products.dropna()
sellers = sellers.dropna()
```

- Standardize column formats: using the Pandas library as an example

```
orders['order_purchase_timestamp'] = pd.to_datetime(orders['order_purchase_timestamp'],errors = 'coerce')
orders['order_approved_at'] = pd.to_datetime(orders['order_approved_at'],errors = 'coerce')
orders['order_delivered_carrier_date'] = pd.to_datetime(orders['order_delivered_carrier_date'],errors = 'coerce')
orders['order_delivered_customer_date'] = pd.to_datetime(orders['order_delivered_customer_date'],errors = 'coerce')
orders['order_estimated_delivery_date'] = pd.to_datetime(orders['order_estimated_delivery_date'],errors = 'coerce')
```

6. Exploratory Data Analysis (EDA)



1. Loading the Tables

- Using **SQL Server Management Studio (SSMS)**, we accessed the database containing our tables.
- We navigated to the **Database Diagrams** section under the database in the Object Explorer.
- A new diagram page was created by right-clicking on **Database Diagrams** and selecting **New Database Diagram**.
- A dialog box prompted us to select the tables to include in the diagram. We loaded the following tables:
  - olist_customers_dataset

- olist_geolocation_dataset
- olist_orders_dataset
- olist_order_items_dataset
- olist_order_payments_dataset
- olist_order_reviews_dataset
- olist_products_dataset
- olist_sellers_dataset

2. Aligning Primary and Foreign Keys

- Once the tables were added to the diagram page, we carefully aligned them to improve readability:
- **Primary Keys (PK):** Highlighted in bold within each table, ensuring their unique role in the dataset.
- **Foreign Keys (FK):** Connections between tables were automatically detected (if defined) and shown as relationship lines.
- For any missing or incorrect relationships, we manually created them:
- Right clicked on the table and selected **Relationships**.
- Added relationships by linking one table's primary key to another's foreign key. For example:
  - customer_id in olist_customers_dataset links to olist_orders_dataset.
  - order_id in olist_orders_dataset links to olist_order_items_dataset.
  - product_id in olist_products_dataset links to olist_order_items_dataset.

3. Ensuring Valid Relationships

- We validated each relationship by:
- Verifying that keys are correctly aligned (e.g., data types match between primary and foreign keys).
- Checking for one-to-many or many-to-one relationships based on the nature of the data:
  - One customer can place multiple orders (one-to-many).
  - One product can appear in multiple order items (many-to-one).

4. Improving Readability

- Tables were arranged logically to reflect their relationships, minimizing line crossings and clutter:
  - Core tables such as olist_orders_dataset were placed centrally, connecting to many other tables.
  - Related tables, such as olist_order_items_dataset and olist_order_payments_dataset, were grouped near olist_orders_dataset.
- We resized table entities and adjusted spacing for a clean, organized view.

5. Saving and Exporting the Diagram

- After completing the alignment and validation, we saved the diagram under a descriptive name for future reference.
- To include the diagram in our project documentation, we exported it as an image:
  - Right clicked on the workspace and selected **Copy to Clipboard**.
  - Pasted the image into our documentation.

Outcome

The final diagram provides a clear and structured overview of our data schema. It highlights all primary and foreign key relationships, making it easier to design SQL queries and explore the dataset during the analysis phase.

7. Analysis and Insights:

First: By MS SQL SERVER

In this section, we delve into the results of our data analysis to uncover actionable insights that can drive strategic decision-making. Using SQL queries, we explored key aspects of the business, including customer distribution, revenue generation, product popularity, and operational efficiency. Each analysis aims to identify trends, understand customer behavior, and highlight areas for improvement. The insights presented here will serve as a foundation for optimizing marketing strategies, enhancing logistics, and improving overall customer satisfaction.

1. Customer Distribution

- **Query:**

```
SELECT COUNT (DISTINCT (customer_city)) AS Distinct_Cities
FROM olist_customers_dataset.
```

- **Detailed Insight:**
  This query calculates the number of unique cities in which customers are located. The results show that the business has a broad geographic reach. By identifying the top cities with the highest concentration of

customers, marketing campaigns can be better targeted.
Follow-Up Actions:

- Analyze customer density per region.
- Align advertising budgets with regions having a higher customer presence.
- Investigate underserved regions to uncover potential growth opportunities.

---

2. Total Revenue

- **Query:**

  SELECT SUM(payment_value) AS Total_Revenue
  FROM olist_order_payments_dataset;

- **Detailed Insight:**
  This query aggregates all payments to calculate the total revenue. This metric is crucial to assess overall business success.
  Additional Analysis:

- Break down revenue by month or year to identify trends over time.
- Compare the revenue generated by different payment types.
- Cross-check total revenue against delivery delays or customer satisfaction to see their impact on business growth.

---

3. Revenue by Location

- **Query:**

  SELECT customer_city, customer_state, SUM (payment_value) AS Total_Revenue
  FROM olist_order_payments_dataset OP
  JOIN olist_customers_dataset C ON OP.customer_id = C.customer_id
  GROUP BY customer_city, customer_state.

- **Detailed Insight:**
  This query combines data from the payments and customers tables to reveal the total revenue generated by city and state.
  Insights from Results:

- States with higher revenue indicate areas of strong customer loyalty and demand.
- Cities with low revenue might need promotional efforts to boost sales. **Follow-Up Actions:**
- Segment revenue further by product category or payment type for granular insights.
- Use location-based insights to enhance logistics planning.

---

4. Sales and Order Trends

- **Query:**

  SELECT order_status, COUNT(order_status) AS Number_Of_Orders

  FROM olist_orders_dataset

  GROUP BY order_status;

- **Detailed Insight:**
  This query provides a count of orders by status (e.g., delivered, canceled, pending).
  Analysis:

  - A higher number of canceled orders could indicate issues in payment, product quality, or customer expectations.
  - Analyzing "pending" orders might reveal bottlenecks in the supply chain.
    Follow-Up Actions:
  - Investigate reasons for cancellations by linking with customer reviews or feedback.
  - Improve communication and support during the order approval phase.

---

5. Popular Products

- **Query:**

  SELECT TOP 10 product_category_name, ROUND(SUM (price), 2) AS Total_Sales

  FROM olist_order_items_dataset

  GROUP BY product_category_name

  ORDER BY Total_Sales DESC.

  - **Detailed Insight:**

  This query identifies the top 10 product categories contributing the most to sales revenue.
  Insights from Results:

  - These categories are high-demand items, crucial for maintaining stock.
  - Trends in popular products could guide decisions on introducing similar products.
    Follow-Up Actions:
  - Compare the sales performance of these categories across different months.
  - Evaluate the effect of discounts or promotions on these products.

---

6. Payment Methods Distribution

- **Query:**

  SELECT payment_type, COUNT (*) AS Num_Payments

  FROM olist_order_payments_dataset

  GROUP BY payment_type;

- **Detailed Insight:**
  This query examines the popularity of different payment methods. Understanding customer preferences here helps streamline payment processes.
  Additional Analysis:

  - Map payment preferences against customer demographics.
  - Consider expanding the range of payment options for methods used less frequently.

---

7. Delivery Time Analysis

- **Query:**

  SELECT product_category_name,

  AVG (DATEDIFF (DAY, order_purchase_timestamp, order_delivered_customer_date)) AS Avg_Delivery_Time

  FROM olist_orders_dataset

  JOIN olist_order_items_dataset ON olist_orders_dataset.order_id = olist_order_items_dataset.order_id

  GROUP BY product_category_name;

- **Detailed Insight:**
  This query calculates the average delivery time for each product category.
  Insights from Results:

  - Categories with longer delivery times may face logistical challenges.
  - Quick delivery categories might indicate efficient supplier chains.
    Follow-Up Actions:
  - Address delays in high-demand categories to improve customer satisfaction.
  - Use this data to inform customers of estimated delivery times.

---

8. Review Sentiments

- **Query:**

```
SELECT product_category_name,
AVG (review_score) AS Avg_Product_Review_Score
FROM olist_order_reviews_dataset
JOIN olist_products_dataset ON olist_order_reviews_dataset.product_id = olist_products_dataset.product_id
GROUP BY product_category_name
ORDER BY Avg_Product_Review_Score DESC.
```

- **Detailed Insight:**
  This query evaluates the average review scores of each product category.
  Insights from Results:
  - Categories with high scores indicate customer satisfaction, likely due to quality or fast delivery.
  - Low scores highlight areas needing improvement, such as quality or delivery issues.
    Follow-Up Actions:
  - Pair review scores with delivery times to see their impact on customer satisfaction.
  - Monitor changes in scores after improving products or services.

Second: By Python

1. Libraries Used

The script utilizes a variety of libraries for:

- **Data Manipulation:** Pandas and NumPy for handling dataframes and numerical computations.
- **Visualization:** Matplotlib and Seaborn for static plots; Plotly for interactive visualizations.
- **Statistical Operations:** Scipy for calculating Z-scores and sklearn.preprocessing for scaling.

---

2. Data Loading

The datasets are loaded using pd.read_csv:

- Tables included:
  - customers, geolocation, order_items, payments, reviews, orders, products, sellers.
- Each dataset is stored in a separate dataframe for further operations.

Key Insight: The script ensures all datasets are accessible, laying the foundation for table merging and deeper analysis.

---

3. Data Cleaning

a. Handling Missing Values

- The .isnull().sum() method is applied to identify null values in each dataframe.
- Rows with missing values are removed using .dropna().

Insight: Removing null values ensures data quality but may reduce the dataset size. A potential improvement could involve imputing missing values where applicable.

---

b. Removing Duplicates

- Duplicate rows are identified using .duplicated().sum() and removed via .drop_duplicates().
- This step ensures the integrity of unique data.

Insight: Cleaning duplicates prevents overestimation in analyses such as total revenue or customer count.

---

4. Data Type Conversion

The script standardizes data types for consistent processing:

- Example: geolocation_zip_code_prefix converted to str.
- Timestamps like order_purchase_timestamp converted to datetime.

Insight: Proper data types improve computational accuracy, particularly for time-based analyses (e.g., delivery time).

---

5. Outlier Detection and Handling

Outliers are identified and treated using the IQR method:

- Example: geolocation_lat and geolocation_lng are checked for extreme values.
- Boxplots visualize data distribution and help detect anomalies.

Steps:

1. Calculate Q1, Q3, and IQR.
2. Define bounds: Lower = Q1 - 1.5 * IQR, Upper = Q3 + 1.5 * IQR.
3. Remove values outside these bounds.

Key Example:
The products['product_weight_g'] column is cleaned to remove unusually heavy or light products, ensuring realistic data.

---

6. Exploratory Data Analysis (EDA)

a. Key Numeric Columns The script identifies numeric columns for analysis using .select_dtypes():

- Examples: price in order_items, payment_value in payments.

Insight: Selecting numeric data focuses EDA on metrics critical to sales, payments, and logistics.

Expanded Analysis and Illustration for Data Analysis Documentation

Here's how you can document the Python script to reflect its key aspects in detail for your **Analysis and Insights** section:

Data Analysis Workflow

1. Libraries Used

The script utilizes a variety of libraries for:

- **Data Manipulation:** Pandas and NumPy for handling dataframes and numerical computations.
- **Visualization:** Matplotlib and Seaborn for static plots; Plotly for interactive visualizations.
- **Statistical Operations:** Scipy for calculating Z-scores and sklearn.preprocessing for scaling.

---

2. Data Loading

The datasets are loaded using pd.read_csv:

- Tables included:
    - customers, geolocation, order_items, payments, reviews, orders, products, sellers.
- Each dataset is stored in a separate dataframe for further operations.

Key Insight: The script ensures all datasets are accessible, laying the foundation for table merging and deeper analysis.

---

3. Data Cleaning

a. Handling Missing Values

- The .isnull().sum() method is applied to identify null values in each dataframe.
- Rows with missing values are removed using .dropna().

Insight: Removing null values ensures data quality but may reduce the dataset size. A potential improvement could involve imputing missing values where applicable.

---

b. Removing Duplicates

- Duplicate rows are identified using .duplicated().sum() and removed via .drop_duplicates().
- This step ensures the integrity of unique data.

Insight: Cleaning duplicates prevents overestimation in analyses such as total revenue or customer count.

---

4. Data Type Conversion

The script standardizes data types for consistent processing:

- Example: geolocation_zip_code_prefix converted to str.
- Timestamps like order_purchase_timestamp converted to datetime.

Insight: Proper data types improve computational accuracy, particularly for time-based analyses (e.g., delivery time).

---

5. Outlier Detection and Handling

Outliers are identified and treated using the IQR method:

- Example: geolocation_lat and geolocation_lng are checked for extreme values.
- Boxplots visualize data distribution and help detect anomalies.

Steps:

1. Calculate Q1, Q3, and IQR.
2. Define bounds: Lower = Q1 - 1.5 * IQR, Upper = Q3 + 1.5 * IQR.
3. Remove values outside these bounds.

Key Example:
The products['product_weight_g'] column is cleaned to remove unusually heavy or light products, ensuring realistic data.

---

6. Exploratory Data Analysis (EDA)

a. Key Numeric Columns The script identifies numeric columns for analysis using .select_dtypes():

- Examples: price in order_items, payment_value in payments.

Insight: Selecting numeric data focuses EDA on metrics critical to sales, payments, and logistics.

---

b. Visualization

- **Boxplots:** Highlight outliers and distributions (e.g., price, freight_value).
- **Histograms/Scatterplots:** (Optional if added) Visualize trends across dimensions like revenue or reviews.

Example Boxplot Analysis:
The boxplot for order_items['freight_value'] shows variability in shipping costs, hinting at factors like region or product weight influencing freight pricing.

8. Visualizations and Dashboards

Visualizations and dashboards play a pivotal role in transforming raw data into actionable insights. Below, we provide an in-depth analysis of the key visualizations created using Tableau and Power BI to demonstrate data patterns and trends, enhancing the comprehensiveness of the analysis. The visualizations effectively illustrate critical insights, making them easier to interpret for stakeholders.

Visualization 1: Number of Orders Per Customer

- **Objective**: This visualization highlights the distribution of orders made by customers.
- **Key Insight**:
    - A majority of customers (96,479) placed only one order, indicating a high level of one-time buyers. This could point to retention challenges.
    - A significantly smaller number of customers made multiple purchases (e.g., 2,382 customers made two orders). This trend diminishes drastically with an increasing number of orders.
- **Actionable Suggestion**:
    - Implement loyalty programs to encourage repeat purchases.
    - Conduct targeted marketing campaigns for customers who have placed two or more orders.

Visualization 2: Olist Overview Dashboard

This comprehensive dashboard provides an aggregated view of Olist's key metrics from 2016 to 2018.

1. **Revenue and Sellers Overview**:
    - **Total Revenue**: $16,009K
    - **Number of Sellers**: 3,000

- Insights into financial performance and seller engagement are presented in a high-level summary.

2. **Top Revenue States**:
    - São Paulo leads significantly, generating $6,466.46K in revenue, followed by Paraná ($1,007.90K).
    - **Actionable Insight**: Focus marketing efforts and inventory optimization in these high-performing states to maximize returns.
3. **Top Product Categories**:
    - **Key Products**: *beleza_saude* (Beauty and Health) and *relogios_presentes* (Watches and Gifts) are the most profitable categories.
    - **Suggestion**: Expand the product range in these categories and ensure a robust supply chain for inventory management.
4. **Customer Feedback and Review Analysis**:
    - **57.4% of reviews** were excellent, showcasing high customer satisfaction.
    - **Actionable Steps**: Investigate the factors contributing to lower ratings (15.4% bad reviews) to improve overall customer experience.

Visualization 3: Revenue and Orders Over Time

- **Objective**: Illustrates the growth trajectory of revenue and orders across quarters from Q3-2016 to Q2-2018.
- **Key Insight**:
    - Both revenue and orders show a consistent upward trend, with a slight plateau towards Q2-2018.
    - **Interpretation**: This indicates a growing customer base and increased sales, though market saturation might be approaching.
- **Actionable Steps**:
    - Diversify offerings or enter new markets to sustain growth.

Visualization 4: Geographic Sales Analysis

- **Objective**: A geographic breakdown of sales performance.
- **Key Insight**:
    - São Paulo dominates with $5,998,227 in revenue.

    - Lesser sales in regions like Roraima (10,065) indicate potential untapped markets.
- **Recommendation**: Develop region-specific strategies to penetrate underperforming areas while consolidating strongholds like São Paulo.