1. **Data Overview & Initial Assessment**

- **Goal**: Understand the structure, quality, and contents of your data.
- **Actions**:
    - Load each dataset (e.g., customer data, transaction data, product catalog) and examine its structure using `.info()` and `.head()`.
    - Check for missing values, duplicates, and data types.
    - Identify key columns for segmentation, such as customer ID, transaction amount, purchase frequency, and product categories.

2. **Data Cleaning**

- **Goal**: Remove inconsistencies, fill in missing values, and ensure data integrity.
- **Actions**:
    - **Handle Missing Values**:
        - For numeric columns: Fill missing values using median or mean imputation, or remove rows with missing values if appropriate.
        - For categorical columns: Use the mode or create an "Unknown" category.
    - **Remove Duplicates**: Identify and remove duplicate entries, especially in transactional data.
    - **Data Type Conversion**: Ensure that data types are appropriate (e.g., dates as `datetime` objects, numeric data as `float` or `int`).
    - **Outlier Detection**: Identify and handle outliers, particularly in columns like transaction amounts, to prevent skewed model results.

3. **Feature Engineering**

- **Goal**: Create meaningful features that capture customer behavior for better segmentation.
- **Actions**:
    - **Customer-Level Aggregations**:
        - Calculate total spend per customer.
        - Count the number of transactions per customer.
        - Calculate the average transaction amount per customer.
        - Determine the frequency of purchases (e.g., weekly or monthly).
    - **RFM Features** (Recency, Frequency, Monetary):
        - **Recency**: Calculate days since the last transaction for each customer.
        - **Frequency**: Calculate the total number of purchases by each customer.
        - **Monetary**: Calculate the total amount spent by each customer.
    - **Product Category Aggregations**:
        - Calculate the most frequently purchased product categories per customer.
        - Determine the average price of products in each category.
        - Count the diversity of categories purchased by each customer.

4. **Data Transformation**

- **Goal**: Standardize and encode data for model compatibility.
- **Actions**:
    - **Normalization/Standardization**:
        - Standardize numerical features (e.g., `StandardScaler` or `MinMaxScaler`) to ensure consistent scaling.
    - **Encoding Categorical Variables**:
        - Apply one-hot encoding to categorical features, such as product categories or customer regions.
        - Use label encoding if categories are ordinal or have a logical order.
    - **Date Features**:
        - Extract additional features from dates, such as day of the week, month, or year, to capture seasonal or weekly purchasing trends.

5. **Data Merging**

- **Goal**: Combine datasets to form a single dataset with enriched information.
- **Actions**:
    - Merge transaction data with customer data using a common key (e.g., `customer_id`).
    - Join product data to the transaction data on `product_id` to incorporate product details like category and price.
    - Perform an inner join or left join depending on whether you want to include customers without recent transactions.

6. **Data Validation**

- **Goal**: Ensure the dataset is ready for analysis by verifying transformations and completeness.
- **Actions**:
    - Check for any remaining missing values post-transformation.
    - Validate the newly engineered features to ensure they make logical sense (e.g., average transaction amounts should be in line with original data).
    - Perform a quick exploratory analysis to confirm that features align with business expectations, such as distributions of transaction amounts and frequency.

7. **Save Preprocessed Data**

- **Goal**: Create a clean dataset ready for model training.
- **Actions**:
    - Save the preprocessed data to a new file (e.g., `preprocessed_customer_data.csv`) for easier access during modeling.
    - If necessary, store the data in a database such as MySQL, especially if the dataset will be accessed by multiple team members or applications.