Database Creation, Table Population, and Business Questions

1. Import Challenges

The goal was to import the Sephora dataset from Kaggle into MySQL Workbench for further analysis and database creation. However, an issue occurred during the import process regarding special characters in the raw data file, which MySQL was not able to supply natively. This report summarizes the steps that I took to clean the data so that it was uploaded successfully into MySQL for use in the new database schema.

The first step was to create a new schema in MySQL Workbench for the Sephora dataset. I created a new schema named sephora_data to organize and house the data for subsequent analysis. The Sephora dataset was initially in CSV format when downloaded from Kaggle. My first attempt was to import the file using the Table Data Import Wizard. However, during the import process, some issues appeared, where MySQL could not handle special characters embedded in the dataset (like accented letters, symbols, etc.) leading to errors. MySQL does not natively support some special characters in raw data files, which caused the import process to fail. Such issues have become fairly frequent when importing datasets from sources whose encoding is either inconsistent or has special character formats. That is when I decided to go outside of MySQL and clean the dataset using R before re-uploading it to MySQL.

In R, I checked the dataset contents for special characters that were causing the problem, including, dashes, accents, any symbols, and other characters that appeared to have non-ASCII origin and were incompatible with MySQL. I used stringr and gsub functions in R to replace or remove all non-ASCII characters.The process involved writing a custom function that cleaned each column of the dataset where these special characters were found.

After cleaning the dataset, a new CSV was created with all the special characters removed, ensuring the file was now compatible with MySQL and I was able to upload it using the Table Data Import Wizard without any more issues. The dataset could now be stored in the sephora_data schema as planned.

2. Conversion of the Dataset into Tables

After successfully cleaning the Sephora dataset to remove special characters, the next step was to transform the data into a relational database format by creating individual tables based on the Entity Relationship Diagram (ERD). These are the steps I took in R to create tables, assign unique IDs, and otherwise approach additional challenges to clean the dataset.

Following the ERD design, I created tables in accordance with the structure of various entities specified in the ERD. Since the original dataset lacked unique identifier (ID) columns, these were created for several tables, including category, brand, price, review, size, and marketing. The dplyr package of R was used at the most basic level possible for constructing tables and manipulating datasets.

Once the tables were created, I ran a consistency check to ensure that all required columns and relationships were present in the data and made sure ID columns were assigned wherever necessary to ensure congruity between tables for relational database integrity. After creating the initial tables, further data cleaning was needed to properly structure certain columns.

One of the main issues was the size field, which contained multiple size options in one cell as a text string. In R I was able to separate the size data and split each size option into its own cell to make this data easier to analyze. After splitting the size data by delimiters, all inconsistencies or missing values were properly filled out by clearing empty cells or ensuring that all new columns were populated properly.

The same process of cleaning up the size column was repeated for the ingredients field, which contained long text strings with lists of ingredients in a single cell. However, unlike the size data, the ingredient lists were much messier and did not have clear delimiters or special indicators that could easily separate the ingredients into individual columns. Given these challenges I thought to leave the ingredients listed within the table in order to not over complicate the data architecture and allow for flexibility in data storage.

In addition, I decided to shift the focus of the analysis to the more structured and interpretable data, such as price, review, and marketing data. These fields were better suited for analysis and were prioritized in the business questions. Finally, once all the data tables were created and cleaned I was able to successfully import them through using the Table Data Import Wizard tool in MySQL Workbench.

3. Data Dictionary

Table Name	Column Name	Data Type	Description
Product	product_id	INT	Unique identifier for each product
	brand_id	INT	Foreign key referencing the Brand table
	category_id	INT	Foreign key referencing the Category table
	product_name	VARCHAR	Name of the product
	product_description	TEXT	Description of the product
	product_size	INT	Size of the product (e.g., volume, weight)
	product_url	VARCHAR	URL of the product
	product_instruction	TEXT	Instructions or usage information for the product
Price	price_id	INT	Unique identifier for each price
	product_id	INT	Foreign key referencing the Product table
	price	DECIMAL	Price of the product
	price_value	DECIMAL	Value of the price (e.g., discounted price)
Review	review_id	INT	Unique identifier for each review
	product_id	INT	Foreign key referencing the Product table
	review_rating	INT	Rating of the product by the customer
	review_love	BOOLEAN	Indication of whether the customer loved the product
Brand	brand_id	INT	Unique identifier for each brand
	brand_name	VARCHAR	Name of the brand
Category	category_id	INT	Unique identifier for each category

	category_name	VARCHAR	Name of the category
Size	size_id	INT	Unique identifier for each size
	size_options	VARCHAR	Available size options for the product
Product_Ingredient	product_id	INT	Foreign key referencing the Product table
	ingredient_id	INT	Foreign key referencing the Ingredient table
Ingredient	ingredient_id	INT	Unique identifier for each ingredient
	ingredient_name	VARCHAR	Name of the ingredient
Marketing	marketing_id	INT	Unique identifier for each marketing detail
	product_id	INT	Foreign key referencing the Product table
	marketing_flag	BOOLEAN	Indicates if the product is part of a marketing campaign
	marketing_content	TEXT	Additional marketing information for the product
	marketing_online_only	BOOLEAN	Indicates if the product is available only online
	marketing_exclusive	BOOLEAN	Indicates if the product is exclusive
	marketing_limited_edition	BOOLEAN	Indicates if the product is a limited edition
	marketing_limited_time_offer	BOOLEAN	Indicates if the product is a limited time offer
Product_Size	product_id	INT	Foreign key referencing the Product table
	size_id	INT	Foreign key referencing the Size table

4. Business Questions

1. What is the average rating and total number of reviews for each product?
2. Which brands have the highest number of products priced above $50?
3. List all limited edition products with their prices and ratings.
4. What is the total number of online-only products in each category?
5. Identify brand performance tiers based on average product rating, review count, and price point.
6. Find products that outperform their brand's average in both ratings and price efficiency (rating-to-price ratio).
7. Which products contribute the most to their brand's total revenue in each category?
8. Which brands have the highest number of high-rated products (above 4.5) in at least 3 different categories?

5. MySQL