**Tutorial: Skincare Product Analysis using Hadoop Cluster**

Christopher Lee

[Clee219@calstatela.edu](mailto:Clee219@calstatela.edu)

**Objective**

The beauty and skincare industry is rapidly changing, driven by new consumer demands, digital influence, and the desire for ethical, personalized products. This tutorial uses two Kaggle datasets, Most Used Beauty & Cosmetics Products in the World and Sephora Products and Skincare Reviews, to analyze these trends. The datasets provide product details, brand attributes, user ratings, ingredients, pricing, and demographics. Hive QL will be used to clean, merge, and analyze data, addressing product popularity, ingredient satisfaction correlation, and the impact of brand and price on purchases.

**Introduction**

In this tutorial you will use HiveQL to analyze skincare products, and what is popular and the possible reasons for popularity.

* Download multiple csv files
* Assemble data using Python
* Upload csv files
* Wrangle the data
* Rename Columns
* Drop Columns
* Normalize Values
* Export Data
* Create Visualizations in Python

**Prerequisites**

You can utilize a Python IDE such as JupyterLab to run the code for assembling the data with the latest Python 3 kernel installed. You can run JupyterLab from a GUI such as Anaconda Navigator.

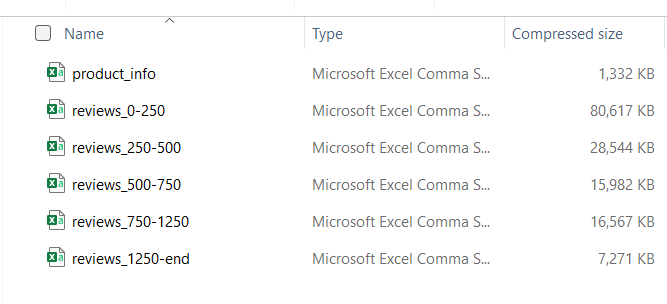
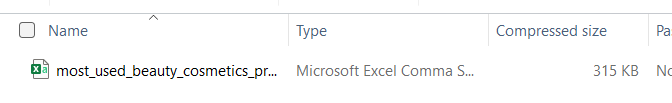
* Anaconda Navigator Download: <https://www.anaconda.com/download>
* Python 3 Kernel: <https://www.python.org/downloads/>

**Download data**

Download the data from the following links to your local machine.

* Sephora Products and Skincare Reviews: <https://www.kaggle.com/datasets/nadyinky/sephora-products-and-skincare-reviews>
* Top Beauty & Cosmetics Products Worldwide 2024: <https://www.kaggle.com/datasets/waqi786/most-used-beauty-cosmetics-products-in-the-world>

**Assemble data using Python**

* Note: You can also download the ipnyb file directly from JupyterLab.
  + <https://github.com/yochris723/CIS5200_Project/blob/affa331555cf8a772e4066360d22bc803fe7b5e5/Code/Table%20Assembly.py>
* Sephora Products and Skincare Reviews
  + A zip file will be provided upon download: Sephora Products and Skincare Reviews.zip
  + Extract the following files
  + 
* Top Beauty & Cosmetics Products Worldwide 2024
  + A zip file will be provided upon download: Top Beauty & Cosmetics Products Worldwide 2024.zip
  + Extract the following file
  + 
* Assemble
  + Open Jupyterlab and create a new notebook by clicking the “Plus” button on the top left. It will take you to a new screen and you can select a new python notebook. This entire section on the initial assembly is entirely performed in Python.
  + Import the libraries that you will need in JupyterLab in python language.
    - import pandas as pd

import os

import glob

* + Locate the directory of the extracted csv files.
    - directory = '/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/Reviews'
    - (Identify your directory, locally on your machine)
  + Initialize an empty list to store DataFrames
    - dataframes = []
  + Reach each CSV file and append to the list
    - for file in csv\_files:

# Read the CSV file into a DataFrame

df = pd.read\_csv(file, low\_memory=False)

# Append the DataFrame to the list

dataframes.append(df)

* + Concatenate all DataFrames into a single DataFrame
    - combined\_df = pd.concat(dataframes, ignore\_index=True)
  + Save the combined DataFrame to a new CSV file
    - combined\_df.to\_csv(os.path.join(directory, 'combined\_sephora\_reviews.csv'), index=False)

print(f"Combined {len(csv\_files)} CSV files into combined\_sephora\_reviews.csv'")

* + Load the CSV files
    - product\_info = pd.read\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/product\_info.csv")
  + Merge the tables on the key 'product\_id'
    - sephora\_reviews = pd.merge(product\_info, combined\_df, on="product\_id", how="inner")
  + Display the first five rows of the merged table
    - sephora\_reviews.head(5)
  + Save the merged table to a new CSV file
    - sephora\_reviews.to\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/sephora\_reviews.csv", index=False)
    - (Identify your directory, locally on your machine)
  + Print the schema of the merged table
    - print(sephora\_reviews.dtypes)
  + Load the csv file " Top \_BeautyCosmetics\_Products\_Worldwide2024.csv" and print the schema of it
    - top\_world\_products = pd.read\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/top\_world\_products.csv")
    - (Identify your directory, locally on your machine)
  + Display the first five rows of the top beauty products table
    - top\_world\_products.head(5)
  + Print the schema of the top beauty products table
    - print(top\_world\_products.dtypes)
  + Drop the column "size""variation\_type"variation\_value""variation\_desc""value\_price\_usd""sale\_price\_usd""limited\_edition""new""online\_only""out\_of\_stock""sephora\_exclusive""highlights""secondary\_category""tertiary\_category""child\_count""child\_max\_price""child\_min\_price""Unnamed: 0""author\_id""rating\_y""helpfulness""submission\_time""review\_text""review\_title""skin\_tone""eye\_color""hair\_color""product\_name\_y""brand\_name\_y""price\_usd\_y"from the sephora\_reviews table
    - sephora\_reviews = sephora\_reviews.drop(columns=["size", "variation\_type", "variation\_value", "variation\_desc", "value\_price\_usd", "sale\_price\_usd", "limited\_edition", "new", "online\_only", "out\_of\_stock", "sephora\_exclusive", "highlights", "secondary\_category", "tertiary\_category", "child\_count", "child\_max\_price", "child\_min\_price", "Unnamed: 0", "author\_id", "rating\_y", "helpfulness", "submission\_time", "review\_text", "review\_title", "skin\_tone", "eye\_color", "hair\_color","product\_name\_y","brand\_name\_y","price\_usd\_y"])
  + Display the first five rows of the sephora\_reviews table
    - sephora\_reviews.head(5)
  + Save the cleaned sephora\_reviews table to a new CSV file
    - sephora\_reviews.to\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/sephora\_reviews\_cleaned.csv", index=False)
    - (Identify your directory, locally on your machine)
  + Print the schema of the cleaned sephora\_reviews table
    - print(sephora\_reviews.dtypes)
  + Drop the column "Usage\_Frequency" "Product\_Size" "Gender\_Target" "Packaging\_Type" "Cruelty\_Free" from the top\_world\_products table
    - top\_world\_products = top\_world\_products.drop(columns=["Usage\_Frequency", "Product\_Size", "Gender\_Target", "Packaging\_Type", "Cruelty\_Free"])
  + Display the first five rows of the top\_world\_products table
    - top\_world\_products.head(5)
  + Save the cleaned top\_world\_products table to a new CSV file
    - top\_world\_products.to\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/top\_world\_products\_cleaned.csv", index=False)
    - (Identify your directory, locally on your machine)
  + Print the schema of the cleaned top\_world\_products table
    - print(top\_world\_products.dtypes)
  + You should now have two files produced from your python code.
    - top\_world\_products\_cleaned.csv
    - sephora\_reviews\_cleaned.csv

**Upload Assembled Data**

Below is the process of uploading data into the linux file system and then into the HDFS (Hadoop Distributed File System)

* --load files into linux files system
  + scp sephora\_reviews.csv clee219@144.24.13.0:~
  + scp top\_world\_products.csv clee219@144.24.13.0:~
* --connect to hadoop cluster
  + ssh clee219@144.24.13.0
  + Enter Password
* --check files (linux)
  + ls
* --check folder (hdfs)
  + hdfs dfs -ls
* --check folder (hdfs)
  + hdfs dfs -ls project
* --create a folder in hdfs
  + hdfs dfs -mkdir project
  + hdfs dfs -mkdir project/reviews
  + hdfs dfs -mkdir project/products
* --load the file into hdfs
  + hdfs dfs -put sephora\_reviews.csv project/reviews
  + hdfs dfs -put top\_world\_products.csv project/products
* --check folder (hdfs)
  + hdfs dfs -ls project/reviews
  + hdfs dfs -ls project/products
* --check file specifications (hdfs)
  + hdfs dfs -du -h project/reviews/sephora\_reviews.csv
  + hdfs dfs -du -h project/products/top\_world\_products.csv
* --check file contents (hdfs)
  + hdfs dfs -cat project/reviews/sephora\_reviews.csv | tail -3
  + hdfs dfs -cat project/products/top\_world\_products.csv | tail -3

**Utilize Hive to create tables and wrangle data**

* --connect to hive
  + beeline
* --Create database (Beeline)
  + create database IF NOT EXISTS clee219;
* --Show database (Beeline)
  + show databases;
* --Use database (Beeline)
  + use clee219;
* --drop table (Beeline)
  + DROP TABLE IF EXISTS sephora\_reviews\_source;
* --create a table from the csv file sephora\_reviews.csv (Beeline)
* CREATE EXTERNAL TABLE IF NOT EXISTS sephora\_reviews\_source (

product\_id string,

product\_name\_x string,

brand\_id int,

brand\_name\_x string,

loves\_count int,

rating\_x float,

reviews float,

size string,

variation\_type string,

variation\_value string,

variation\_desc string,

ingredients string,

price\_usd\_x float,

value\_price\_usd float,

sale\_price\_usd float,

limited\_edition int,

new int,

online\_only int,

out\_of\_stock int,

sephora\_exclusive int,

highlights string,

primary\_category string,

secondary\_category string

tertiary\_category string,

child\_count int,

child\_max\_price float,

child\_min\_price float,

Unnamed string,

author\_id string,

rating\_y int,

is\_recommended float,

helpfulness float,

total\_feedback\_count int,

total\_neg\_feedback\_count int,

total\_pos\_feedback\_count int,

submission\_time string,

review\_text string,

review\_title string,

skin\_tone string,

eye\_color string,

skin\_type string,

hair\_color string,

product\_name\_y string,

brand\_name\_y string,

price\_usd\_y float)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ","

LOCATION "/user/clee219/project/reviews"

TBLPROPERTIES ('skip.header.line.count' = '1');

* --Display the schema of the merged table

DESCRIBE FORMATTED sephora\_reviews\_source;

* --check table (Beeline)

select \* from sephora\_reviews\_source LIMIT 3;

**Normalize Values**

* --drop table (Beeline)
  + DROP TABLE IF EXISTS sephora\_reviews\_normalize;
* -- Normalize strings for all the string columns in the sephora\_reviews table
  + CREATE TABLE sephora\_reviews\_normalize AS

SELECT

LOWER(TRIM(product\_id)) AS product\_id,

LOWER(TRIM(product\_name\_x)) AS product\_name\_x,

LOWER(TRIM(brand\_name\_x)) AS brand\_name\_x,

LOWER(TRIM(size)) AS size,

LOWER(TRIM(variation\_type)) AS variation\_type,

LOWER(TRIM(variation\_value)) AS variation\_value,

LOWER(TRIM(variation\_desc)) AS variation\_desc,

LOWER(TRIM(ingredients)) AS ingredients,

LOWER(TRIM(highlights)) AS highlights,

LOWER(TRIM(primary\_category)) AS primary\_category,

LOWER(TRIM(secondary\_category)) AS secondary\_category,

LOWER(TRIM(tertiary\_category)) AS tertiary\_category,

LOWER(TRIM(submission\_time)) AS submission\_time,

LOWER(TRIM(review\_text)) AS review\_text,

LOWER(TRIM(review\_title)) AS review\_title,

LOWER(TRIM(skin\_tone)) AS skin\_tone,

LOWER(TRIM(eye\_color)) AS eye\_color,

LOWER(TRIM(skin\_type)) AS skin\_type,

LOWER(TRIM(hair\_color)) AS hair\_color,

LOWER(TRIM(product\_name\_y)) AS product\_name\_y,

LOWER(TRIM(brand\_name\_y)) AS brand\_name\_y,

brand\_id,

loves\_count,

rating\_x,

price\_usd\_x,

is\_recommended,

total\_feedback\_count,

total\_neg\_feedback\_count,

total\_pos\_feedback\_count,

reviews

FROM sephora\_reviews\_source;

* --Display the schema of the merged table
  + DESCRIBE FORMATTED sephora\_reviews\_normalize;
* --Display the first five rows of the normalized sephora\_reviews table
  + SELECT \* FROM sephora\_reviews\_normalize LIMIT 5;
* --drop table (Beeline)
  + DROP TABLE IF EXISTS top\_world\_products\_source;
* --create a table from the csv file top\_world\_products.csv (Beeline)
  + CREATE EXTERNAL TABLE IF NOT EXISTS top\_world\_products\_source (

Product\_Name string,

Brand string,

Category string,

Usage\_Frequency string,

Price\_USD float,

Rating float,

Number\_of\_Reviews int,

Product\_Size string,

Skin\_Type string,

Gender\_Target string,

Packaging\_Type string,

Main\_Ingredient string,

Cruelty\_Free boolean,

Country\_of\_Origin string)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ","

LOCATION "/user/clee219/project/products"

TBLPROPERTIES ('skip.header.line.count' = '1');

* --Display the schema of the merged table
  + DESCRIBE FORMATTED top\_world\_products\_source;
* --check table (Beeline)
  + select \* from top\_world\_products\_source LIMIT 3;
* --drop table (Beeline)
  + DROP TABLE IF EXISTS top\_world\_products\_normalize;
* --Normalize strings for all the string columns in the top\_world\_products table
  + CREATE TABLE top\_world\_products\_normalize AS

SELECT

LOWER(TRIM(Product\_Name)) AS Product\_Name,

LOWER(TRIM(Brand)) AS Brand,

LOWER(TRIM(Category)) AS Category,

LOWER(TRIM(Usage\_Frequency)) AS Usage\_Frequency,

LOWER(TRIM(Product\_Size)) AS Product\_Size,

LOWER(TRIM(Skin\_Type)) AS Skin\_Type,

LOWER(TRIM(Gender\_Target)) AS Gender\_Target,

LOWER(TRIM(Packaging\_Type)) AS Packaging\_Type,

LOWER(TRIM(Main\_Ingredient)) AS Main\_Ingredient,

LOWER(TRIM(Country\_of\_Origin)) AS Country\_of\_Origin,

Price\_USD,

Rating,

Number\_of\_Reviews,

Cruelty\_Free

FROM top\_world\_products\_source;

* --Display the schema of the merged table
  + DESCRIBE FORMATTED top\_world\_products\_normalize;
* --Display the first five rows of the normalized top\_world\_products table
  + SELECT \* FROM top\_world\_products\_normalize LIMIT 5; **Drop Columns**
* --drop table (Beeline)
  + DROP TABLE IF EXISTS sephora\_reviews\_clean;

**Drop Columns**

* -- Create a new table with selected columns from sephora\_reviews
  + CREATE TABLE sephora\_reviews\_clean AS

SELECT

product\_id,

product\_name\_x,

brand\_id,

brand\_name\_x,

loves\_count,

rating\_x,

reviews,

ingredients,

price\_usd\_x,

primary\_category,

is\_recommended,

total\_feedback\_count,

total\_neg\_feedback\_count,

total\_pos\_feedback\_count,

skin\_type

FROM sephora\_reviews\_normalize;

* -- Display the first five rows of the cleaned sephora\_reviews table
  + SELECT \* FROM sephora\_reviews\_clean LIMIT 5;
* -- Print the schema of the cleaned sephora\_reviews table
  + DESCRIBE FORMATTED sephora\_reviews\_clean;
* --drop table (Beeline)
  + DROP TABLE IF EXISTS top\_world\_products\_clean;
* -- Create a new table with selected columns from top\_world\_products
  + CREATE TABLE top\_world\_products\_clean AS

SELECT

Product\_Name,

Brand,

Category,

Usage\_Frequency,

Product\_Size,

Skin\_Type,

Gender\_Target,

Packaging\_Type,

Country\_of\_Origin,

Price\_USD,

Rating,

Number\_of\_Reviews,

Main\_Ingredient,

Cruelty\_Free

FROM top\_world\_products\_normalize;

* -- Display the first five rows of the cleaned top\_world\_products table
  + SELECT \* FROM top\_world\_products\_clean LIMIT 5;
* -- Print the schema of the cleaned top\_world\_products table
  + DESCRIBE FORMATTED top\_world\_products\_clean;

Data Cleaning

* --Detect the missing values in the sephora\_reviews table
  + SELECT

COUNT(\*) AS total\_rows,

COUNT(product\_id) AS missing\_product\_id,

COUNT(product\_name\_x) AS missing\_product\_name\_x,

COUNT(brand\_id) AS missing\_brand\_id,

COUNT(brand\_name\_x) AS missing\_brand\_name\_x,

COUNT(loves\_count) AS missing\_loves\_count,

COUNT(rating\_x) AS missing\_rating\_x,

COUNT(reviews) AS missing\_reviews,

COUNT(price\_usd\_x) AS missing\_price\_usd\_x,

COUNT(primary\_category) AS missing\_primary\_category,

COUNT(is\_recommended) AS missing\_is\_recommended,

COUNT(total\_feedback\_count) AS missing\_total\_feedback\_count,

COUNT(total\_neg\_feedback\_count) AS missing\_total\_neg\_feedback\_count,

COUNT(total\_pos\_feedback\_count) AS missing\_total\_pos\_feedback\_count,

COUNT(skin\_type) AS missing\_skin\_type

FROM sephora\_reviews\_clean;

* --drop table (Beeline)
  + DROP TABLE IF EXISTS sephora\_reviews\_updated;
* --Fill missing string values with a placeholder
  + CREATE TABLE sephora\_reviews\_updated AS

SELECT

product\_id,

product\_name\_x,

brand\_id,

brand\_name\_x,

loves\_count,

rating\_x,

reviews,

price\_usd\_x,

primary\_category,

COALESCE(ingredients, "Not specified") AS ingredients, -- Replace NULL with "Not specified"

COALESCE(is\_recommended, 0) AS is\_recommended, -- Replace NULL with 0

total\_feedback\_count,

total\_neg\_feedback\_count,

total\_pos\_feedback\_count,

COALESCE(skin\_type, "unknown") AS skin\_type -- Replace NULL with "unknown"

FROM sephora\_reviews\_clean;

* --Calculate the mode for skin\_type
  + SELECT skin\_type

FROM sephora\_reviews\_updated

GROUP BY skin\_type

ORDER BY COUNT(\*) DESC

LIMIT 1;

* --Drop table
  + DROP TABLE sephora\_reviews\_filled;
* --Fill missing skin\_type values with the mode
  + CREATE TABLE sephora\_reviews\_filled AS

SELECT

product\_id,

product\_name\_x,

brand\_id,

brand\_name\_x,

loves\_count,

rating\_x,

reviews,

price\_usd\_x,

primary\_category,

ingredients,

COALESCE(skin\_type, "normal") AS skin\_type, -- Replace NULL with the mode

is\_recommended,

total\_feedback\_count,

total\_neg\_feedback\_count,

total\_pos\_feedback\_count

FROM sephora\_reviews\_updated;

* --Drop table
  + DROP TABLE sephora\_reviews\_filled;
* --Fill missing skin\_type values with the mode
  + CREATE TABLE sephora\_reviews\_filled AS

SELECT

product\_id,

product\_name\_x,

brand\_id,

brand\_name\_x,

loves\_count,

rating\_x,

reviews,

price\_usd\_x,

primary\_category,

ingredients,

COALESCE(skin\_type,

(SELECT skin\_type

FROM sephora\_reviews\_updated

GROUP BY skin\_type

ORDER BY COUNT(\*) DESC

LIMIT 1)) AS skin\_type, -- Dynamically replace NULL with the mode

is\_recommended,

total\_feedback\_count,

total\_neg\_feedback\_count,

total\_pos\_feedback\_count

FROM sephora\_reviews\_updated;

* --Detect the missing values in the top\_world\_products table
  + SELECT

COUNT(\*) AS total\_rows,

SUM(CASE WHEN Product\_Name IS NULL THEN 1 ELSE 0 END) AS missing\_Product\_Name,

SUM(CASE WHEN Brand IS NULL THEN 1 ELSE 0 END) AS missing\_Brand,

SUM(CASE WHEN Category IS NULL THEN 1 ELSE 0 END) AS missing\_Category,

SUM(CASE WHEN Usage\_Frequency IS NULL THEN 1 ELSE 0 END) AS missing\_Usage\_Frequency,

SUM(CASE WHEN Price\_USD IS NULL THEN 1 ELSE 0 END) AS missing\_Price\_USD,

SUM(CASE WHEN Rating IS NULL THEN 1 ELSE 0 END) AS missing\_Rating,

SUM(CASE WHEN Number\_of\_Reviews IS NULL THEN 1 ELSE 0 END) AS missing\_Number\_of\_Reviews,

SUM(CASE WHEN Product\_Size IS NULL THEN 1 ELSE 0 END) AS missing\_Product\_Size,

SUM(CASE WHEN Skin\_Type IS NULL THEN 1 ELSE 0 END) AS missing\_Skin\_Type,

SUM(CASE WHEN Gender\_Target IS NULL THEN 1 ELSE 0 END) AS missing\_Gender\_Target,

SUM(CASE WHEN Packaging\_Type IS NULL THEN 1 ELSE 0 END) AS missing\_Packaging\_Type,

SUM(CASE WHEN Main\_Ingredient IS NULL THEN 1 ELSE 0 END) AS missing\_Main\_Ingredient,

SUM(CASE WHEN Cruelty\_Free IS NULL THEN 1 ELSE 0 END) AS missing\_Cruelty\_Free,

SUM(CASE WHEN Country\_of\_Origin IS NULL THEN 1 ELSE 0 END) AS missing\_Country\_of\_Origin

FROM top\_world\_products\_clean;

**Rename Columns**

* --drop table (Beeline)
  + DROP TABLE IF EXISTS sephora\_reviews\_renamed
* --Rename the column names in the sephora\_reviews table
  + CREATE TABLE sephora\_reviews\_renamed AS

SELECT

product\_id AS Product\_ID,

product\_name\_x AS Product\_Name,

brand\_id AS Brand\_ID,

brand\_name\_x AS Brand\_Name,

loves\_count AS Loves,

rating\_x AS Rating,

ingredients AS Ingredients,

price\_usd\_x AS Price\_USD,

primary\_category AS Category,

is\_recommended AS Is\_Recommended,

total\_feedback\_count AS Feedback,

total\_neg\_feedback\_count AS Neg\_Feedback,

total\_pos\_feedback\_count AS Pos\_Feedback,

skin\_type AS Skin\_Type,

reviews AS Reviews

FROM sephora\_reviews\_filled;

* --Display the first five rows of the renamed sephora\_reviews table
  + SELECT \* FROM sephora\_reviews\_renamed LIMIT 5;
* --Print the schema of the renamed sephora\_reviews table
  + DESCRIBE FORMATTED sephora\_reviews\_renamed;
* --drop table (Beeline)
  + DROP TABLE IF EXISTS top\_world\_products\_renamed;
* Rename the column names in the top\_world\_products table
  + CREATE TABLE top\_world\_products\_renamed AS

SELECT

Product\_Name,

Brand AS Brand\_Name,

Category,

Price\_USD,

Rating,

Number\_of\_Reviews AS Reviews,

Skin\_Type,

Main\_Ingredient AS Ingredients,

Country\_of\_Origin

FROM top\_world\_products\_clean;

* --Display the first five rows of the renamed top\_world\_products table
  + SELECT \* FROM top\_world\_products\_renamed LIMIT 5;
* --Print the schema of the renamed top\_world\_products table
  + DESCRIBE FORMATTED top\_world\_products\_renamed;
* Save and rename the sephora\_reviews table to a new CSV file
  + INSERT OVERWRITE DIRECTORY '/user/clee219/project/sephora\_reviews\_final'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT \* FROM sephora\_reviews\_renamed;

* Save the top\_world\_products table to an HDFS directory
  + INSERT OVERWRITE DIRECTORY '/user/clee219/project/top\_world\_products\_final'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT \* FROM top\_world\_products\_renamed;

**Statistical Analysis**

* --Summary statistics in sephora\_reviews table
  + SELECT

MIN(Price\_USD) AS min\_price,

MAX(Price\_USD) AS max\_price,

AVG(Price\_USD) AS avg\_price,

MIN(Rating) AS min\_rating,

MAX(Rating) AS max\_rating,

AVG(Rating) AS avg\_rating,

MIN(Reviews) AS min\_reviews,

MAX(Reviews) AS max\_reviews,

AVG(Reviews) AS avg\_reviews

FROM sephora\_reviews\_renamed;

* --Summary statistics in top\_world\_products table
  + SELECT

MIN(Price\_USD) AS min\_price,

MAX(Price\_USD) AS max\_price,

AVG(Price\_USD) AS avg\_price,

MIN(Rating) AS min\_rating,

MAX(Rating) AS max\_rating,

AVG(Rating) AS avg\_rating,

MIN(Reviews) AS min\_reviews,

MAX(Reviews) AS max\_reviews,

AVG(Reviews) AS avg\_reviews

FROM top\_world\_products\_renamed;

* --Individual stats in sephora\_reviews table
  + SELECT

AVG(Price\_USD) AS price\_mean,

PERCENTILE\_APPROX(CAST(Price\_USD AS DOUBLE), 0.5) AS price\_median, -- Use PERCENTILE\_APPROX

STDDEV(Price\_USD) AS price\_std,

MIN(Rating) AS rating\_min,

MAX(Rating) AS rating\_max,

SUM(Reviews) AS review\_sum,

AVG(Reviews) AS review\_mean

FROM sephora\_reviews\_renamed;

* --Individual stats in top\_world\_products table
  + SELECT

AVG(Price\_USD) AS price\_mean,

PERCENTILE\_APPROX(CAST(Price\_USD AS DOUBLE), 0.5) AS price\_median, -- Use the correct column name

STDDEV(Price\_USD) AS price\_std,

MIN(Rating) AS rating\_min,

MAX(Rating) AS rating\_max,

SUM(Reviews) AS review\_sum,

AVG(Reviews) AS review\_mean

FROM top\_world\_products\_renamed;

**Download Files**

* --check for files and directories (hadoop)
  + hdfs dfs -ls /user/clee219/project
* --copy files to local (hadoop)
  + hdfs dfs -get /user/clee219/project/sephora\_reviews\_final/00000\*\_0 sephora\_reviews\_final.csv

hdfs dfs -get /user/clee219/project/top\_world\_products\_final/00000\*\_0 top\_world\_products\_final.csv

* --check local files (linux)
  + ls
* --check file (linux)
  + cat sephora\_reviews\_final.csv | tail -n 2

cat top\_world\_products\_final.csv | tail -n 2

* --if it exists remove the file (linux)
  + rm sephora\_reviews\_final.csv

rm top\_world\_products\_final.csv

* --download file (linux)
  + scp clee219@144.24.13.0:~/sephora\_reviews\_final.csv .

scp clee219@144.24.13.0:~/top\_world\_products\_final.csv .

**Create Visualizations in Python**

* # load the csv file "sephora\_reviews\_final.csv" as generated by Hive QL and print the schema of it
  + sephora\_reviews = pd.read\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/sephora\_reviews\_final.csv")
* # Display the first five rows of the sephora\_reviews table
  + sephora\_reviews.head(5)
* # print the schema of the sephora\_reviews table
  + print(sephora\_reviews.dtypes)
* # load the csv file " Top \_BeautyCosmetics\_Products\_Worldwide2024.csv" and print the schema of it
  + top\_world\_products = pd.read\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/top\_world\_products\_final.csv")
* # Display the first five rows of the top beauty products table
  + top\_world\_products.head(5)
* # print the schema of the top beauty products table
  + print(top\_world\_products.dtypes)
* #Import libraries for visualizations
  + import matplotlib.pyplot as plt

import seaborn as sns

* # 1. Top 10 most popular products from each dataset, categorized the same products, ranking by reviews
* fig, axes = plt.subplots(1, 2, figsize=(15, 6))

sns.barplot(data=sephora\_reviews.groupby("Product\_Name", as\_index=False)["reviews"]

.max()

.sort\_values(by="reviews", ascending=False)

.head(10), x="reviews", y="Product\_Name", ax=axes[0])

axes[0].set\_title("Top 10 Sephora Products by Reviews")

axes[0].set\_xlabel("Reviews")

axes[0].set\_ylabel("Product")

sns.barplot(data=top\_world\_products.sort\_values(by="Reviews", ascending=False).head(10), x="Reviews", y="Product\_Name", ax=axes[1])

axes[1].set\_title("Top 10 Global Products by Reviews")

axes[1].set\_xlabel("Reviews")

axes[1].set\_ylabel("Product")

plt.tight\_layout()

plt.show()

* # 2. Top 10 most popular products from each dataset, categorized the same products, ranking by rating
* fig, axes = plt.subplots(1, 2, figsize=(20, 10))

sns.barplot(data=sephora\_reviews.groupby("Product\_Name", as\_index=False)["Rating"]

.max()

.sort\_values(by="Rating", ascending=False).head(10), x="Rating", y="Product\_Name", ax=axes[0])

axes[0].set\_title("Top 10 Sephora Products by Rating")

axes[0].set\_xlabel("Rating")

axes[0].set\_ylabel("Product")

sns.barplot(data=top\_world\_products.sort\_values(by="Rating", ascending=False).head(10), x="Rating", y="Product\_Name", ax=axes[1])

axes[1].set\_title("Top 10 Global Products by Rating")

axes[1].set\_xlabel("Rating")

axes[1].set\_ylabel("Product")

plt.tight\_layout()

plt.show()

* # 3. Popularity vs Price
* plt.figure(figsize=(10, 6))

sns.scatterplot(data=sephora\_reviews, x="Price\_USD", y="reviews", hue="Rating", palette="viridis", alpha=0.7)

plt.title("Popularity vs Price (Sephora)")

plt.xlabel("Price (USD)")

plt.ylabel("Popularity (Reviews)")

plt.legend(title="Rating", bbox\_to\_anchor=(1.05, 1), loc='upper left')

plt.tight\_layout()

plt.show()

* # 4. Average Rating by Country (top\_world\_products only)
* avg\_rating\_world = top\_world\_products.groupby(["Country\_of\_Origin"])["Rating"].mean().reset\_index()

plt.figure(figsize=(10, 6))

sns.lineplot(data=avg\_rating\_world, x="Country\_of\_Origin", y="Rating", marker="o")

plt.title("Global: Average Rating by Country")

plt.xlabel("Country")

plt.ylabel("Average Rating")

plt.tight\_layout()

plt.show()

* # 5. Average Product Price Across Skin Types (Sephora vs World)
* fig, axes = plt.subplots(1, 2, figsize=(14, 6))

sns.barplot(data=sephora\_reviews, x="Skin\_Type", y="Price\_USD", ax=axes[0])

axes[0].set\_title("Sephora: Average Price by Skin Type")

axes[0].set\_xlabel("Skin Type")

axes[0].set\_ylabel("Average Price (USD)")

sns.barplot(data=top\_world\_products, x="Skin\_Type", y="Price\_USD", ax=axes[1])

axes[1].set\_title("World: Average Price by Skin Type")

axes[1].set\_xlabel("Skin Type")

axes[1].set\_ylabel("Average Price (USD)")

plt.tight\_layout()

plt.show()

* #6 World Products Analysis
* top\_world\_products = pd.read\_csv("/Users/CLee/OneDrive - Cal State LA/CIS 5200/Project/Sephora Products and Skincare Reviews/top\_world\_products\_final.csv")

top\_world\_products.head(5)

top\_ingredients\_world = top\_world\_products.groupby("Ingredients")["Reviews"].sum().sort\_values(ascending=False).head(10)

plt.figure(figsize=(10, 6))

top\_ingredients\_world.plot(kind="bar")

plt.title("Top Global Ingredients by Reviews")

plt.xlabel("Ingredient")

plt.ylabel("Total Reviews")

plt.tight\_layout()

plt.show()