

|  |
| --- |
| ETL Project |
| Two Sunnys |
| Report |
|  |
|  |
|  |

Sunmin Lee

|  |  |
| --- | --- |
| Thidar Swe Tin | 21 April 2019 |

Contents

[Project Description 3](#_Toc6736092)

[Extract 4](#_Toc6736093)

[Data Source 1 – Sephora (Sunmin Lee) 4](#_Toc6736094)

[Data Source 2 – Strawberrynet (Thidar Swe Tin) 5](#_Toc6736095)

[Transform 6](#_Toc6736096)

[Data Source 1 – Sephora (Sunmin Lee) 6](#_Toc6736097)

[Data Source 2 – Strawberrynet (Thidar Swe Tin) 8](#_Toc6736098)

[Load 11](#_Toc6736099)

[Appendices 13](#_Toc6736100)

[MongoDB Database 14](#_Toc6736101)

[MySQL Database 16](#_Toc6736102)

# Project Description

Web scraping of products and reviews from 2 sources (Sephora and Strawberrynet) for the following categories:

#### Skin care categories:

● Eye & lip

● Cleansers

● Masks

● Moisturizers / Treatments

● Sun Care

#### Steps:

1. Scrape all the product urls from the category page
2. Scrape all the product details from the product urls
3. Store raw data from scraped URLs into MongoDB
4. Retrieve data from MongoDB and transform the data using panda dataframes
5. Load the transformations back into MySQL
6. Create views in MySQL for both sites to look at
   * Top 100 products by ratings
   * Top 10 brands by ratings

# Extract

## Data Source 1 – Sephora (Sunmin Lee)

#### Step 1: Scrape all the product urls from the category page

##### URL of webpage to be scraped

* + - eye\_url = 'https://www.sephora.com/shop/eye-treatment-dark-circle-treatment?pageSize=300'
    - lip\_url = 'https://www.sephora.com/shop/lip-treatments?pageSize=300'
    - cleanser1\_url = 'https://www.sephora.com/shop/cleanser?pageSize=300&currentPage=1'
    - cleanser2\_url = 'https://www.sephora.com/shop/cleanser?pageSize=300&currentPage=2'
    - mask1\_url = 'https://www.sephora.com/shop/face-mask?pageSize=300&currentPage=1'
    - mask2\_url = 'https://www.sephora.com/shop/face-mask?pageSize=300&currentPage=2'
    - moist1\_url = 'https://www.sephora.com/shop/moisturizing-cream-oils-mists?pageSize=300&currentPage=1'
    - moist2\_url = 'https://www.sephora.com/shop/moisturizing-cream-oils-mists?pageSize=300&currentPage=2'
    - moist3\_url = 'https://www.sephora.com/shop/moisturizing-cream-oils-mists?pageSize=300&currentPage=3'
    - treat1\_url = 'https://www.sephora.com/shop/facial-treatments?pageSize=300&currentPage=1'
    - treat2\_url = 'https://www.sephora.com/shop/facial-treatments?pageSize=300&currentPage=2'
    - suncare\_url = 'https://www.sephora.com/shop/sunscreen-sun-protection?pageSize=300'

##### Products found [Total: **2852**]

● Eye & lip [**381**]

● Cleansers [**517**]

● Masks [**338**]

● Moisturizers / Treatments [**1398**]

● Sun Care [**218**]

#### Step 2: Scrape all the product details from the product urls

Took a total of **7 hours** to scrape all the **2852** product details.

#### Step 3: Store raw data from scraped URLs into MongoDB

Refer to [Appendices: MongoDB Database Structure (Sephora)](#_Database_Structure_(Sephora)) for more details.

## Data Source 2 – Strawberrynet (Thidar Swe Tin)

#### Step 1: Scrape all the product urls from the category page

##### URL of webpage to be scraped

urls = [{"eyes\_lips":"/en-us/skincare/?groupid=3&sort=popularity"},

{"cleansers":"/en-us/skincare/cleansers/t/?groupid=4&sort=popularity"},

{"masks":"/en-us/skincare/masks/t/?groupid=4&sort=popularity"},

{"moisturisers":"/en-us/skincare/moisturizers-treatments/t/?groupid=4&sort=popularity"},

{"sun\_care":"/en-us/skincare/?groupid=7&sort=popularity"}]

##### Products found [Total: **3247**]

● Eye & lip [**641**]

● Cleansers [**444**]

● Masks [**521**]

● Moisturizers / Treatments [**1228**]

● Sun Care [**413**]

#### Step 2: Scrape all the product details from the product urls

Took a total of **8.5 hours** to scrape all the **3247** products details and reviews.

#### Step 3: Store raw data from scraped URLs into MongoDB

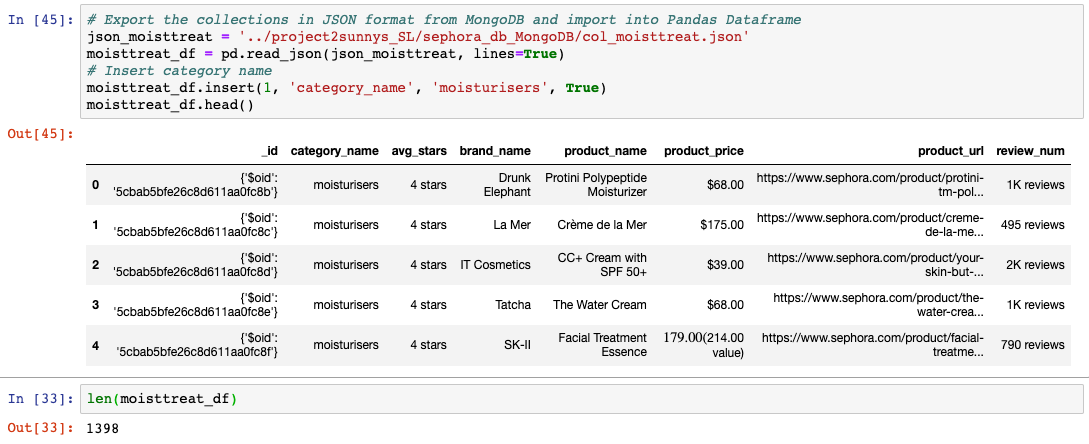
Refer to [Appendices: MongoDB Database Structure (Strawberrynet)](#_Database_Structure_(Strawberrynet)) for more details.

# Transform

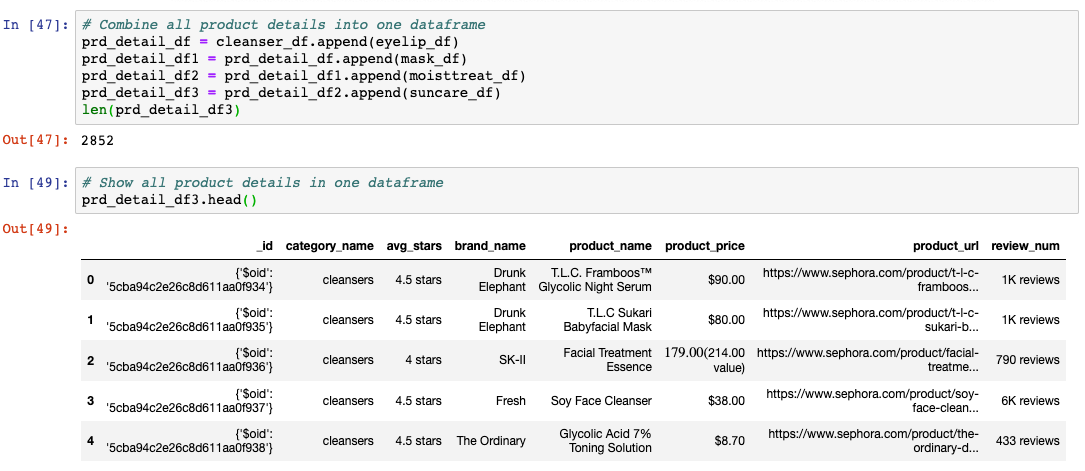
## Data Source 1 – Sephora (Sunmin Lee)

#### Step 4: Retrieve data from MongoDB and transform the data using pandas dataframes

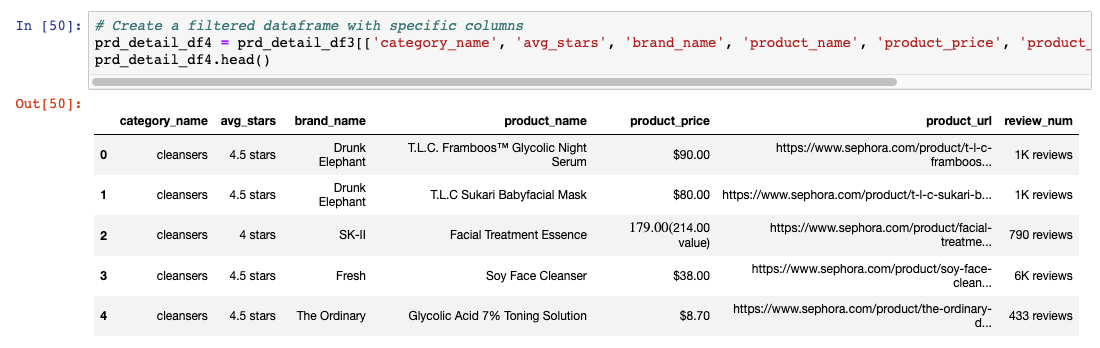
1. Export the collections in JSON format from MongoDB and import into Pandas dataframe



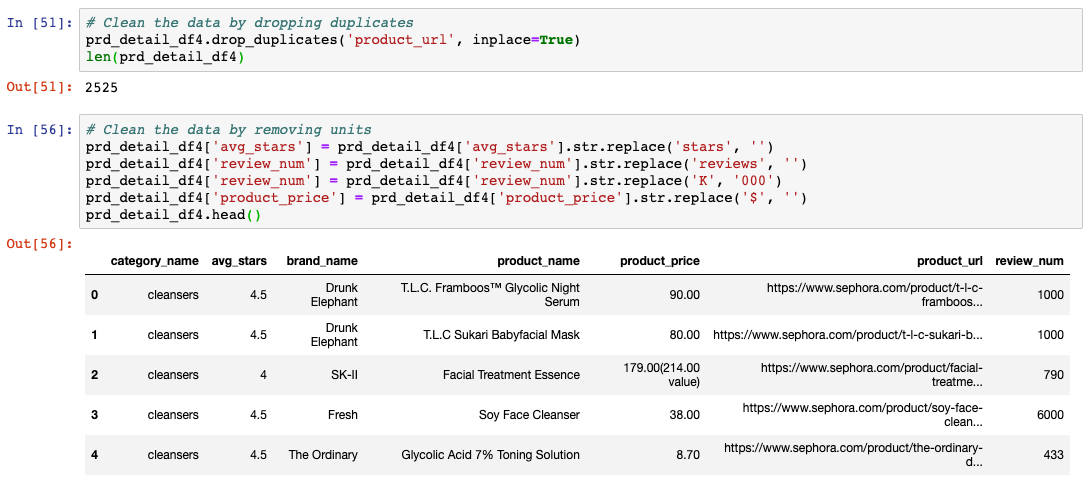
1. Combine all product details into one dataframe



1. Create a filtered dataframe with specific columns



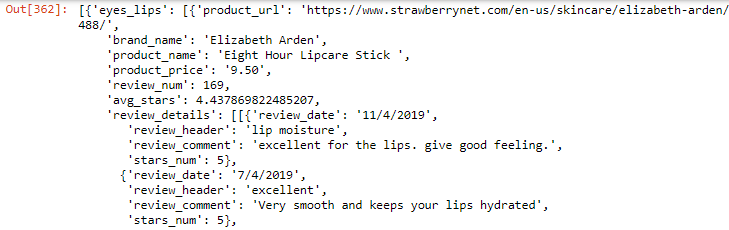
1. Clean the data by dropping duplicates / Clean the data by removing units



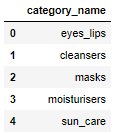
## Data Source 2 – Strawberrynet (Thidar Swe Tin)

#### Step 4: Retrieve data from MongoDB and transform the data using pandas dataframes

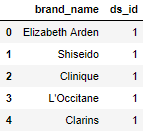
1. Get all items from product details collection in MongoDB



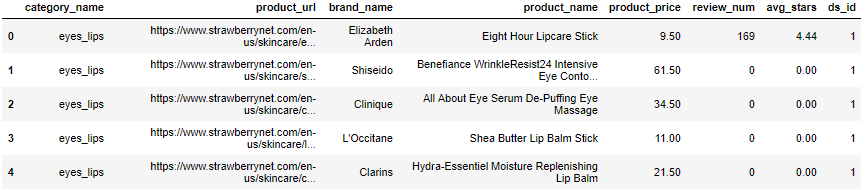
1. Store categories in DataFrame (category\_df)



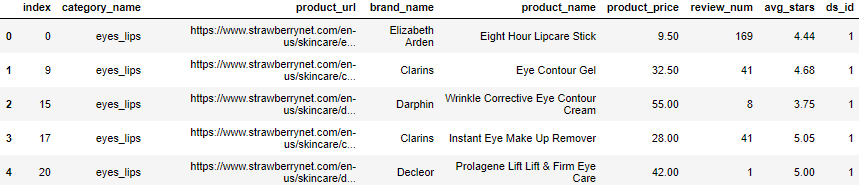
1. Store unique brand names in DataFrame (brand\_df)



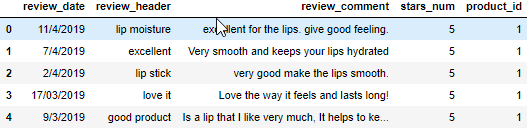
1. Store products in DataFrame (product\_df)



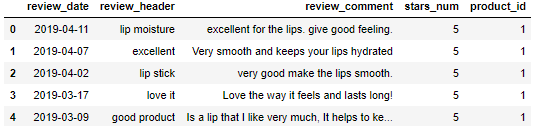
1. Get all the products that has reviews (review\_df)



1. Retrieve reviews and store reviews in dataframe (review\_df)



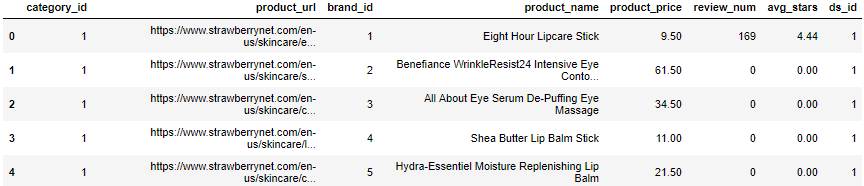
1. Drop duplicates, invalid dates and convert to date time (review\_df)



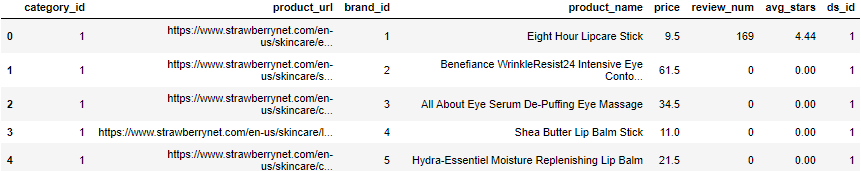
1. Replace brand names with brand id (product\_df)



1. Replace category names with category id (product\_df)



1. Clean product price (product\_df)



# Load

#### Step 5: Load the transformations back into MySQL

Load the 4 dataframes into MySQL (.to\_sql)

* category\_df
* brand\_df
* product\_df
* review\_df

Refer to [Appendices: MySQL Database Structure](#_Database_Structure)  for more details.

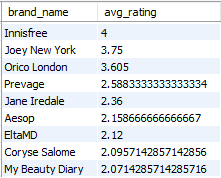


#### Step 6: Create views in MySQL for both sites to look at

* Top 100 products by ratings



* Top 10 brands by ratings

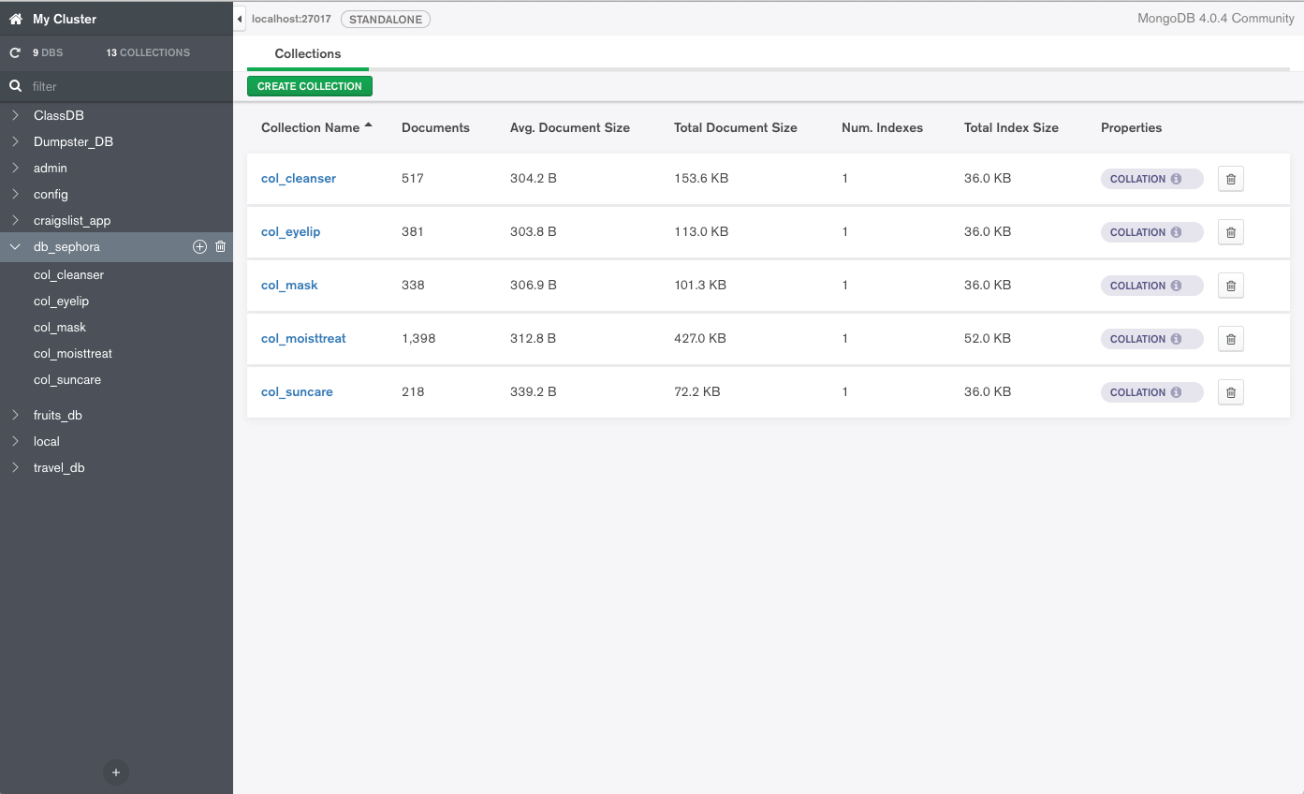


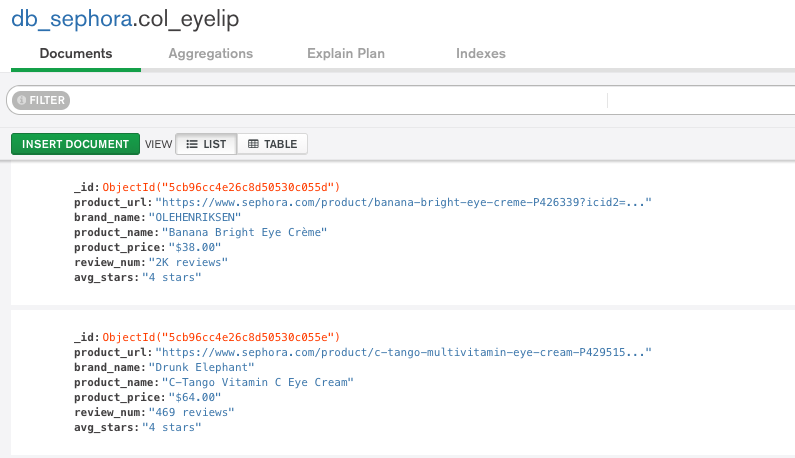
* Refer to [Appendices: MySQL Database Script](#_Database_Script) for more details.

Appendices

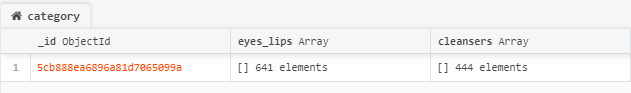
MongoDB Database

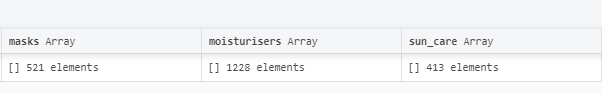
#### Database Structure (Sephora)

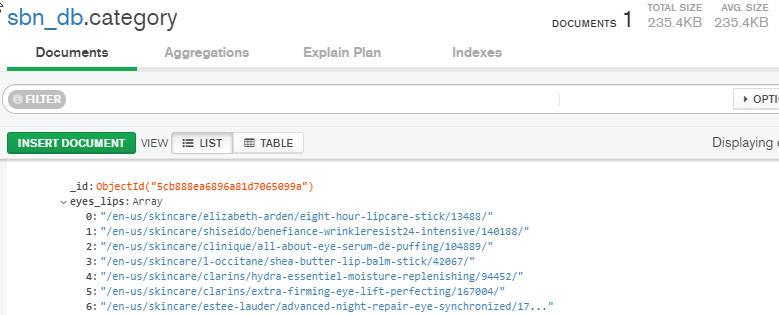


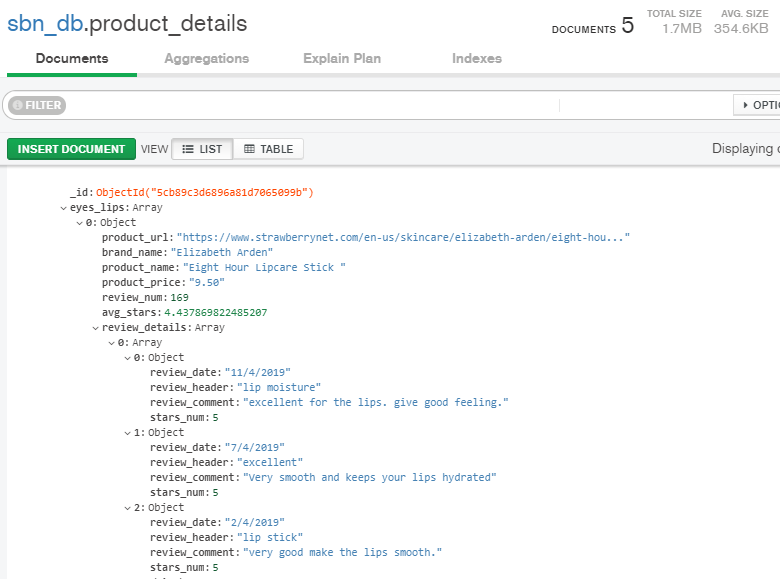


#### Database Structure (Strawberrynet)





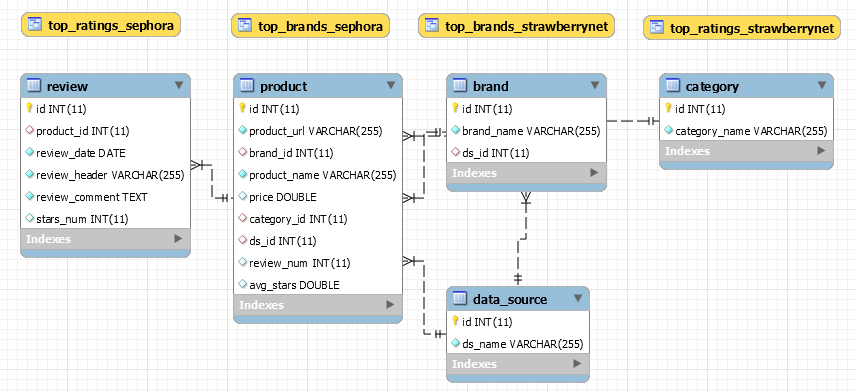




MySQL Database

#### Database Structure

* **5 tables and 4 views**



#### Database Script

CREATE DATABASE skincare\_db;

USE skincare\_db;

CREATE TABLE IF NOT EXISTS category (

id INT AUTO\_INCREMENT,

category\_name VARCHAR(255) NOT NULL,

PRIMARY KEY (id)

);

CREATE TABLE IF NOT EXISTS data\_source (

id INT AUTO\_INCREMENT,

ds\_name VARCHAR(255) NOT NULL,

PRIMARY KEY (id)

);

CREATE TABLE IF NOT EXISTS brand (

id INT AUTO\_INCREMENT,

brand\_name VARCHAR(255) NOT NULL,

ds\_id INT,

PRIMARY KEY (id),

FOREIGN KEY (ds\_id) REFERENCES data\_source(id)

);

CREATE TABLE IF NOT EXISTS product (

id INT AUTO\_INCREMENT,

product\_url VARCHAR(255) NOT NULL,

brand\_id INT,

product\_name VARCHAR(255) NOT NULL,

price DOUBLE,

category\_id INT,

ds\_id INT,

review\_num INT,

avg\_stars DOUBLE,

PRIMARY KEY (id),

FOREIGN KEY (brand\_id) REFERENCES brand(id),

FOREIGN KEY (category\_id) REFERENCES category(id),

FOREIGN KEY (ds\_id) REFERENCES data\_source(id)

);

CREATE TABLE IF NOT EXISTS review (

id INT AUTO\_INCREMENT,

product\_id INT,

review\_date DATE NOT NULL,

review\_header VARCHAR(255) NOT NULL,

review\_comment TEXT NOT NULL,

stars\_num INT,

PRIMARY KEY (id),

FOREIGN KEY (product\_id) REFERENCES product(id)

);

INSERT INTO data\_source (ds\_name) VALUES ('Strawberrynet');

INSERT INTO data\_source (ds\_name) VALUES ('Sephora');

INSERT INTO data\_source (ds\_name) VALUES ('Both');

CREATE VIEW top\_ratings\_strawberrynet AS SELECT c.category\_name, b.brand\_name, p.product\_name, p.price, p.avg\_stars

FROM product p

JOIN brand b

ON (p.brand\_id = b.id)

JOIN category c

ON (p.category\_id = c.id)

JOIN data\_source ds

ON (p.ds\_id = ds.id)

WHERE ds.id = 1 ORDER BY p.avg\_stars DESC, c.category\_name ASC LIMIT 100;

CREATE VIEW top\_ratings\_sephora AS SELECT c.category\_name, b.brand\_name, p.product\_name, p.price, p.avg\_stars

FROM product p

JOIN brand b

ON (p.brand\_id = b.id)

JOIN category c

ON (p.category\_id = c.id)

JOIN data\_source ds

ON (p.ds\_id = ds.id)

WHERE ds.id = 2 ORDER BY p.avg\_stars DESC, c.category\_name ASC LIMIT 100;

CREATE VIEW top\_brands\_strawberrynet AS SELECT b.brand\_name, avg(p.avg\_stars) as avg\_rating

FROM product p

JOIN brand b

ON (p.brand\_id = b.id)

WHERE p.ds\_id = 1 GROUP BY b.brand\_name ORDER BY avg\_rating DESC LIMIT 10;

CREATE VIEW top\_brands\_sephora AS SELECT b.brand\_name, avg(p.avg\_stars) as avg\_rating

FROM product p

JOIN brand b

ON (p.brand\_id = b.id)

WHERE p.ds\_id = 2 GROUP BY b.brand\_name ORDER BY avg\_rating DESC LIMIT 10;

