Analysis On Beauty Industry

Team 5: Ahjeong Yeom, Kenji Laurens, Steve Shi And Vanessa Li

Agenda

01 **Team Introduction**

02 **Executive Summary**

03 **Data**

- Data Infrastructure
- Data Profile
- Data Quality Dimension Assessment
- ETL Process
- Implementation Tools
- Design Considerations
- Data Collection
- Data Preparation

03 **Cloud Storage: GCP**

04 **EER & Dimensional Model**

05 **NoSQL: MongoDB & Neo4j**

06 **Five Business Use Cases**

07 **Lessons & Recommendations**

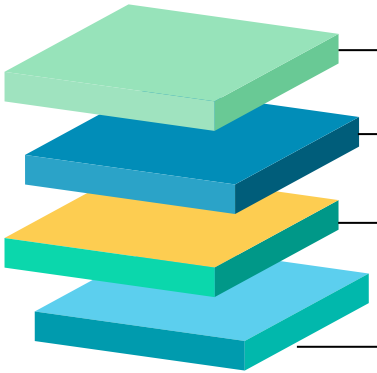
Team: Millenium Consulting

Ahjeong Yeom	Kenji Laurens	Steve Shi	Vanessa Li
			
Founder Data Scientist	Founder Data Scientist	Founder Data Scientist	Founder Data Scientist

Executive Summary

- **Our goal is to bring more meaningful and multi-dimensional data to users on cosmetic brands and products so that both brands and end-consumers can make informed decisions**
- **Data Collection:** utilize different tools such as Tweepy API, XHR requests to extract product, twitter, brands, reviews and ingredient data from various sources
- **Data Cleaning:** utilize Python and OpenRefine to clean data and make them ready to import into MySQL
- **Data Storage:** database is stored in cloud (GCP) to allow multiple remote collaborators and MySQL Workbench was chosen as GUI tool
- **Data Model:** EER and dimensional model created
- **Data Analysis:** utilize SQL, Pandas, NLP libraries to perform analysis
- **Data Visualization On 5 Business Use Cases :**
    1. Brand Perception Based On Sentiment Of Tweets
    2. Key Performance
    3. User Journey On Demo App Built On Tkinter
    4. Competitor Lookup
    5. Customer Demographic and Attribute Distribution

Data Infrastructure

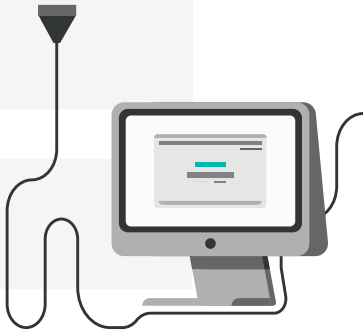


Portal
Visualization and analytic tools: Tableau, Plotly, Neo4j, Pandas, Tkinter

Strategic Data Marts
Brands, Product Reviews, Ingredients, Performance

Data Warehouse
MySQL, GCP, MongoDB, Neo4j

Data Lake
Google Drive



Data Profile

	SOURCE	DESCRIPTION	DATA SIZE	FORMAT
Product	Kaggle	Products listed on Sephora	12.4MB	csv
Brands	Phantom Buster	Information on cosmetic brands and company information	226KB	json/csv
Reviews	BazaarVoice API	User product reviews on Sephora	500MB	json
Twitter	Tweepy API	Users' tweets with hashtag of brand name	7.3MB	csv
Ingredient	California Safe Cosmetics Program Product DB	Chemicals used in cosmetics that are known to be harmful	2.2MB	csv

Data Quality Dimensions Assessment

Completeness

Original data did not contain all products up to this year and tweets data was limited to recent data

Accuracy

Tweet data contained inaccurate feedback on brands as search query was simply limited to "#brand" and not advanced keywords such as product names

Consistency

The amount of data for product, reviews and tweets per brand was not consistent

Validity

For ingredient harmfulness, the data is perfectly valid as it has scientific backing. Using tweet sentiment and review data however, may not be perfectly representative of reality

Uniqueness

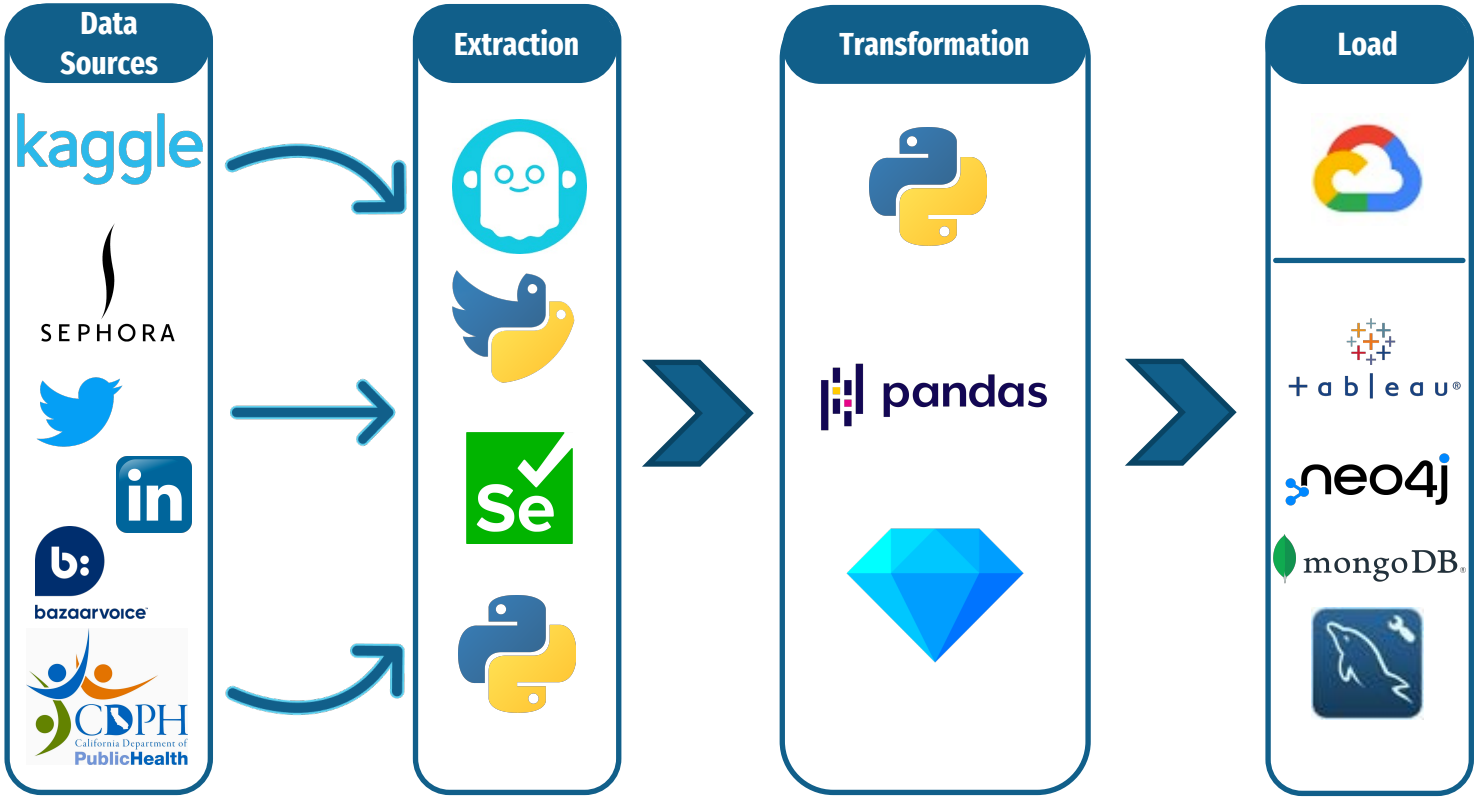
A customer may have repeated review based on multiple experience

Sentiment analysis may be based on multiple tweets that could have been one

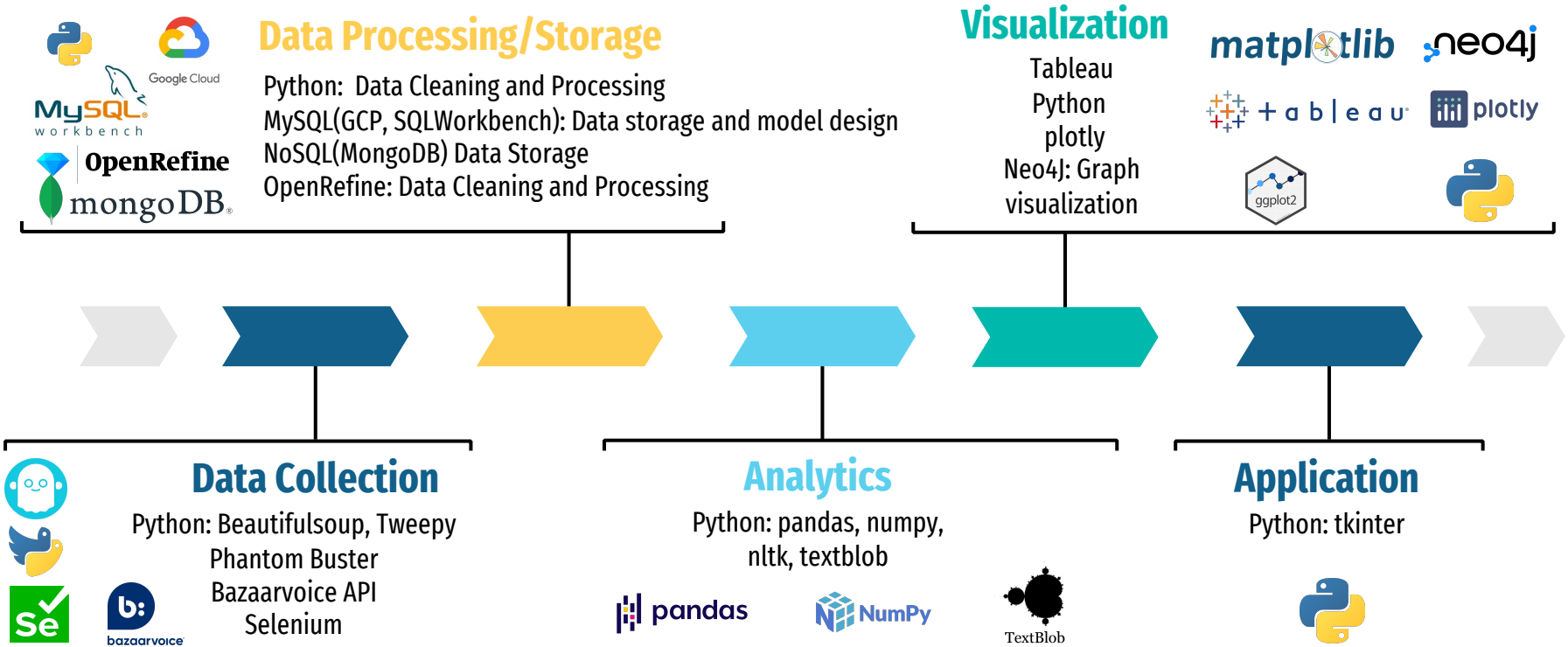
Integrity

Entity, domain and referential integrities were considered with constraints

ETL Process: Extraction, Transformation & Load



Implementation Tools



Design Considerations

Branding
Original product contains products across all categories, but we limit our data to facial brands under Sephora's makeup and skincare categories

Outliers & Anomalies
Tweet location data contains non-location data
Intensive clustering and replacing with NULL if no-location can be derived from

Ingredients
Intermediary lookup table linking harmful ingredients to the product' ingredient

Dealing With NAs
MySQL Workbench recognizes NULL instead of empty string so removed unused columns and filled NaNs as NULL

Product Reviews
The reviews have too many attributes that are sparsely populated due to the difference between product categories, making it hard to create table columns in SQL for it

Thus, using NoSQL and store as key value pairs is what we decided to do

Data Naming Convention
Standardize the format for names across entities and attributes (snake case)

Normalization
Using brand id across the tables, referring to the brand table to normalize

Data Collection: Selenium Attempt

Selenium 4

```
import requests, selenium, time
from selenium import webdriver
from bs4 import BeautifulSoup
```

Initial scrape attempt did not work, Sephora website keeps on timing out

```
soup = BeautifulSoup(requests.get('https://www.sephora.com/shop/makeup-cosmetics', timeout = 3))
```

```
---------------------------------------------------------------------------
timeout                                   Traceback (most recent call last)
~/opt/anaconda3/lib/python3.8/site-packages/urllib3/connectionpool.py in _make_request(self, conn, method, url, timeout, chunked, **httplib_request_kw)
    444                     # Otherwise it looks like a bug in the code.
--> 445                     six.raise_from(e, None)
    446         except (SocketTimeout, BaseSSLError, SocketError) as e:

~/opt/anaconda3/lib/python3.8/site-packages/urllib3/packages/six.py in raise_from(value, from_value)

~/opt/anaconda3/lib/python3.8/site-packages/urllib3/connectionpool.py in _make_request(self, conn, method, url, timeout, chunked, **httplib_request_kw)
    439                 try:
--> 440                     httplib_response = conn.getresponse()
```

Tried to use Selenium to emulate a browser

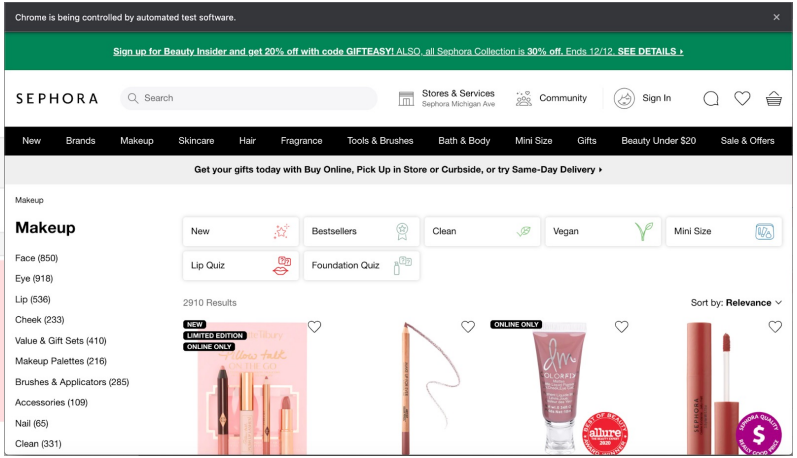
```
driver = webdriver.Chrome("/Users/admin/Downloads/chromedriver")
```

```
<ipython-input-75-ae252c651443>:1: DeprecationWarning: executable_path has been deprecated, please pass in a Service object
  driver = webdriver.Chrome("/Users/admin/Downloads/chromedriver")
```

Website is accessible using Selenium, but since page does not load everything upfront, browser manipulation is needed to scroll down the page and load more products.

```
driver.implicitly_wait(10)
driver.get('https://www.sephora.com/shop/makeup-cosmetics')
while True:
    soup = BeautifulSoup(driver.page_source)
    div = soup.select('div[style^="order:"]')
    if '''LazyLoad"''' not in str(div):
        break
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
```

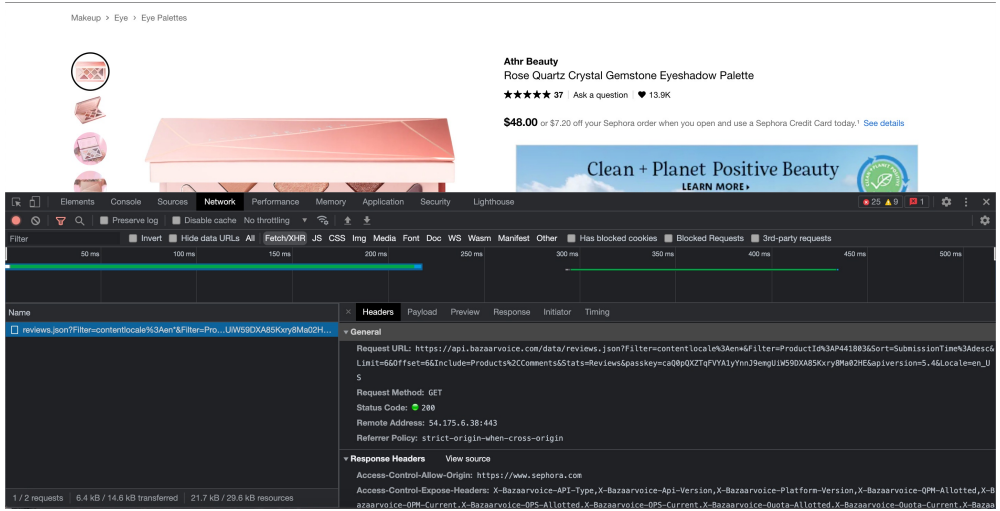
This was quite tricky and complicated, so we decided to go a different approach and used XHR requests to get product reviews ¶



Selenium was considered initially as a substitute to BS4 to scrape Sephora

- Due to JavaScript components on Sephora, BS4 was not able to properly parse the page
- Used Selenium to emulate a browser and execute JavaScript
- Problems arose due to delayed page execution

Data Collection: XHR Requests To Scrape Sephora Data



XHR provided a better solution to scrape Sephora product reviews over BS4 & Selenium

- Since scraping with both BS4 and Selenium did not work, we turned to XHR requests
- This turned to our advantage since the XHR API was generally cleaner, more complete product listing (website removed old products) and contains reviewer metadata

```
[7]: file = '/Users/admin/Desktop/Uchicago/Data Platform Engineering/Project/reviews2.csv'
     products = '/Users/admin/Desktop/Uchicago/Data Platform Engineering/Project/products2.csv'
     calls = 0
     prod_cols=['Description', 'Id', 'ImageUrl', 'Name', 'ReviewStatistics','TotalReviewCount']
     review_cols=['UserNickname','Rating','ReviewText','ContextDataValues','ProductId','AuthorId','SubmissionTime','IsRecommended','ContentLocale','Id']
     downloaded = pd.read_csv(products)['Id'].to_list()
     for k, v in tqdm(df.iterrows(), total = len(df)):
         continue
         offset = 0
         limit = 50
         temp = pd.DataFrame()
         pid = v['URL'].split('grid:')[1].upper()
         if pid in downloaded:
             continue
         while True:
             url = f'https://api.bazaarvoice.com/data/reviews.json?Filter=contentlocale%3Aen*&Filter=ProductId%3A{pid}&Sort=SubmissionTime%3Adesc&Limit={limit}&Of
             r = requests.get(url,timeout=10)
             calls += 1
             response = json.loads(r.text)
             total = response['TotalResults']
             if total == 0:
                 break

             temp = pd.concat([temp, pd.DataFrame(response['Results'])[review_cols]])
             offset += limit

             c = len(temp)
             if c >= total or (c >= 100 and c > int(total * 0.1)):
                 break
         if total == 0:
             continue
         if len(temp) > 0:
             review = pd.concat([review,temp])

         products_df = pd.concat([products_df, pd.DataFrame(response['Includes']['Products']).T.reset_index()])
         products_df.reset_index(inplace=True, drop=True)

         if k % 1000 == 0:
             products_df[prod_cols].to_csv(products, index=False)
             review.to_csv(file, index=False)

     products_df[prod_cols].to_csv(products, index=False)
     review.to_csv(file, index=False)
```

```
100%|██████████| 4419/4419 [00:00<00:00, 14171.98it/s]
{
  "Includes": {
    "Products": {
      "P486520": {
        "CategoryId": "cat60045",
        "EANs": [

        ],
        "Description": "What it is: A rose-quartz infused, sustainable eyeshadow palette that offers a variety of 10 romantic shades in shimmer, metallic and matte finishes.Highlighted Ingredients: - Organic* and Fair Trade** Coconut Oil: Creates a buttery-smooth texture.- Rose Quartz Gemstone Powder: Helps diminish the look of wrinkles, redness, and fine lines.- Organic* Rose Hip Oil: Supports radiant-looking, glowing skin. Ingredient Callouts: This product is vegan, cruelty-free, and gluten-free.What Else You Need to Know: Athr Beauty's bestselling palette is redesigned to illuminate the skin with high-payoff, pigment-packed eyeshadows. Made with real rose quartz that fights visible signs aging and helps to reduce the appearance of fine lines, wrinkles, and dark spots. Organic coconut oil calms, protects, and hydrates the skin; organic shea butter offers a smooth, buttery texture for a one-swipe application; and organic rosehip oil boosts collagen, soothes irritation, and brightens the skin. Seven additional, universally flattering shades cool tones come in a more travel-friendly, compact size. This product is formulated without talc, parabens, nanoparticles, and fragrance-ensuring these ingredients are fair-trade, child-labor free, and ethically sourced. This Palette Contains:- 8 x 0.05 oz/ 1.5 g Eyeshadows in Stardust (light pink shimmer), Sandstone (neutral beige matte), Heartstone (rose pink shimmer), Rose Quartz (antique rose metallic), Scorpio (rose brown matte), Crystalline (antique champagne metallic), Aphrodite (violet gray matte), Grounding (charcoal matte)- 4 x 0.04 oz/ 1.25 g Eyeshadows in Reawakening (pink to gold duo-chrome shimmer), Divine (pink to peach duo-chrome shimmer), Astarte ( lavender to silver blue duo-chrome shimmer), Aura (maroon to blue purple duo-chrome shimmer)*Certified organic by Quality Assurance International. **Certified Fair Trade by Fair Trade USA\n Clean + Planet Positive at Sephora\nWhen you see our Clean + Planet Positive seal, this brand's products are formulating without certain ingredients that are potentially harmful to human health and environment. They also address ingredients clients are concerned about, including phthalates, formaldehyde or formaldehyde releasers, oxybenzone and octinoxate, hydroquinone, triclosan, coal tar, methylisothiazolinone and methylchloroisothiazolinone, insoluble plastic microbeads, and more: For the full list of Ingredients tab.On top of meeting the Clean at Sephora formulated without list, these brands are on a mission to do right by your routine and the planet by m itments and focusing on sustainable sourcing, responsible packaging, and environmental giving.",
        "ModelNumbers": [

        ],
        "Attributes": {
          "BV_FE_EXPAND": {
            "Values": [
              {
                "Locale": null,
                "Value": "BV_FE_FAMILY:P441803"
              }
            ],
```

bazaarvoice

Data Collection: Tweepy Api

```
from config import *
import tweepy as tw
import pandas as pd
import numpy as np
import re
import csv
```


```
#Authentification process for twitter (tweepy api)
auth = tw.OAuthHandler(TWITTER_CONSUMER_KEY, TWITTER_CONSUMER_SECRET)
auth.set_access_token(TWITTER_ACCESS_TOKEN, TWITTER_ACCESS_TOKEN_SECRET)
api = tw.API(auth, wait_on_rate_limit=True)
```

```
add_tweets = pd.DataFrame(columns=['brand','tweet_id', 'tweet_date', 'follower_count', 'retweets','location','tweet_text'])
```

```
for x in brands: #goes through each keyword
    for tweet in tw.Cursor(api.search_tweets, tweet_mode='extended', q=x, lang="en").items(200):
        appending_dataframe = pd.DataFrame([[x, tweet.id,
                                             tweet.created_at,
                                             tweet.user.followers_count,
                                             tweet.retweet_count,
                                             tweet.user.location,
                                             tweet.full_text.encode('utf-8')]],
                                             columns=['brand','tweet_id', 'tweet_date', 'follower_count', 'retweets','location','tweet_text'])
        add_tweets = add_tweets.append(appending_dataframe)
```

	Brand	Tweet Id	Tweet Date	Follower Count	Retweets	Location	Tweet Text	brand_index
0	AERIN	1460799974095015939	2021-11-17 02:40:28+00:00	81	0	San Diego	b"@RTCath95 he's awffully strange aint he?"	1
1	AERIN	1460799337655590919	2021-11-17 02:37:57+00:00	595	0	テキサス	b'@Just_Like_Aerin Lyse is a solid not entirel...	1
2	AERIN	1460798057801392131	2021-11-17 02:32:51+00:00	81	0	San Diego	b"@SNovatore true let's homosexually terrorize...	1
3	AERIN	1460797365879590916	2021-11-17 02:30:06+00:00	1042	0	NaN	b'@spaceshinji ngl if I were to go with someth...	1
4	AERIN	1460796093441232900	2021-11-17 02:25:03+00:00	595	0	テキサス	b'@Just_Like_Aerin Alphinaud is your new middl...	1
...	...	...	...	...	...	...	...	...
20569	Yves Saint Laurent	1460180104194129923	2021-11-15 09:37:20+00:00	1491	0	she/her — 17	b'besties, how do i pronounce yves saint laure...	1

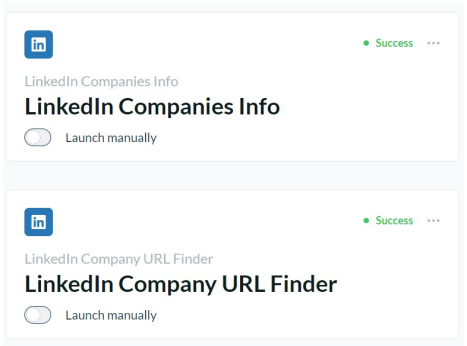
Tweepy Api helped collect tweets with brand name hashtag (ie. #AERIN)

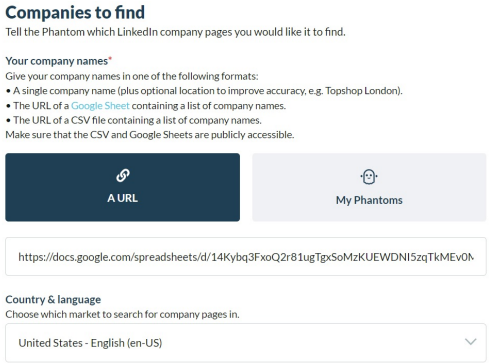
- Tweeter developer account created to use the free-tier api calls to scrape tweet data per brand (brand, tweet_id, tweet_date, follower_count, retweets, location, tweet_text)
- Iterate over brand names in brands table and append collected tweets to dataframe using Pandas
- For 186 brands, avg 110 tweets per brand were collected but number of tweets varied per brand, which could result in diminishing the value of analysis

Data Collection: Phantom Buster

Phantom Buster Was A Great Substitute to BS4 To Scrape Brand Data

- Attempted to scrape data from Wikipedia, but not all brands are listed on Wikipedia
- Discovered most brands have their company page on LinkedIn providing valuable information
- Chose **Phantom Buster** to extract brand information automatically from LinkedIn
- The extraction result is in JSON/CSV format





[{"linkedinUrl":"https://www.linkedin.com/company/philosophy","description":"philosophy | 6,543 followers on LinkedIn. philosophy's promise is to bring its customers products that inspire them to live a better life by being better to themselves. our products care for your skin and your overall well-being. we take a mind, body, spirit approach to personal care and believe only when you feel good can you truly look your best.","title":"philosophy | LinkedIn","query":"philosophy","timestamp":"2021-11-23T19:45:19.621Z"},{"linkedinUrl":"https://www.linkedin.com/company/pmd","description":"PMD - Profesionales en Manejo de Datos S.A. de C.V. | 918 followers on LinkedIn. En PMD trabajamos en conjunto con las organizaciones para fomentar un óptimo aprovechamiento de sus datos ...","title":"PMD - Profesionales en Manejo de Datos S.A. de C.V. | LinkedIn","query":"PMD","timestamp":"2021-11-23T19:45:25.783Z"},{"linkedinUrl":"https://www.linkedin.com/company/pretty-vulga-llc","description":"Pretty Vulgar | 56 followers on LinkedIn. A Beautiful Contradiction | A South Florida based brand that offers full eye, face, and lip collections, Pretty Vulgar cosmetics embodies the fact that ...","title":"Pretty Vulgar | LinkedIn","query":"Pretty Vulgar","timestamp":"2021-11-23T19:45:30.998Z"},{"linkedinUrl":"https://www.linkedin.com/company/prima-co","description":"Prima. 3,795 followers. 4d. Report this post. We're honored and excited to have won the Luxury Meets Cannabis Conference Gold Leaf 2021 Best Wellness Brand Award alongside other incredible ...","title":"Prima | LinkedIn","query":"Prima","timestamp":"2021-11-23T19:45:36.088Z"},{"linkedinUrl":"https://www.linkedin.com/company/primera","description":"Primera | 1,457 followers on LinkedIn. Primera--Bringing Home Interiors to Life Primera, a privately owned and operated company, employs over 400 staff across 5 States. It is a dynamic and growing ...","title":"Primera | LinkedIn","query":"Primera","timestamp":"2021-11-23T19:45:40.963Z"},{"linkedinUrl":"https://www.linkedin.com/company/proactiv-s.l.","description":"Proactiv Entertainment is Spain's leading producer and promoter of large-scale shows, international exhibitions and music concerts. Based in Barcelona, with offices in Madrid and Abu Dhabi, we ...","title":"Proactiv Entertainment | LinkedIn","query":"Proactiv","timestamp":"2021-11-23T19:45:46.342Z"},{"linkedinUrl":"https://www.linkedin.com/company/ren-skincare","description":"REN Clean Skincare | 24,449 followers on LinkedIn. High-performance skincare with clinically proven results | Formulated for sensitive skin | Zero Waste by end of 2021 | 🧴 high-performance skincare with clinically proven results 🌿 with nature's most powerful + kindest ingredients 🌱 formulated for sensitive skin","title":"REN Clean Skincare | LinkedIn","query":"REN Clean Skincare","timestamp":"2021-11-23T19:45:51.180Z"},{"linkedinUrl":"https://www.linkedin.com/company/rms-beauty","description":"rms beauty. 6,412 followers. 2w. Report this post. Straight Up™ Mascara was named Best Mascara over $20! We are so honored The ZOE Report mentioned us in their \"20 Best New Makeup of 2021 Awards ...","title":"rms beauty | LinkedIn","query":"rms beauty","timestamp":"2021-11-23T19:45:55.994Z"},{"linkedinUrl":"https://www.linkedin.com/company/the-estee-lauder-companies-inc","description":"The Estée Lauder Companies Inc. | 1.244.286 seguidores en LinkedIn. The Estée Lauder Companies Inc. is one of the world's leading manufacturers and marketers of quality skin care, makeup, fragrance and hair care products. As the global leader in prestige beauty, we touch over half a billion consumers a year. Our Company's products are sold in over 150 countries and territories under the ...","title":"The Estée Lauder Companies Inc. | LinkedIn","query":"RODIN olio lusso","timestamp":"2021-11-23T19:46:01.361Z"},{"linkedinUrl":"https://www.linkedin.com/company/roen-beauty","description":"RÓEN Beauty was Co-founded by beauty devotee and clean beauty advocate Tiffany Thurston Scott in the heart of Los Angeles. After collaborating with world renowned fashion photographer David ...","title":"RÓEN Beauty | LinkedIn","query":"ROEN BEAUTY","timestamp":"2021-11-23T19:46:06.192Z"},{"linkedinUrl":"https://www.linkedin.com/company/rosebud-mining-co-","description":"Rosebud Mining Co. | 73 abonnés sur LinkedIn. Rosebud Mining Co. is a company based out of United States.","title":"Rosebud Mining Co. | LinkedIn","query":"Rosebud Perfume Co.","timestamp":"2021-11-23T19:46:11.249Z"},{"linkedinUrl":"https://www.linkedin.com/company/saint-jane-beauty","description":"Saint Jane Beauty | 924 followers on LinkedIn. Thoughtful, Mindful, Luxury Beauty Made By Women, For Women | We are Saint Jane. Luxury CBD beauty, small-batch-crafted in California with thoughtful ...","title":"Saint Jane Beauty | LinkedIn","query":"Saint Jane Beauty","timestamp":"2021-11-23T19:46:16.577Z"},{"linkedinUrl":"https://www.linkedin.com/company/saturday-skin","description":"SATURDAY SKIN Cosmetics Los Angeles, California 88 followers Like a weekend in bottle, Saturday Skin leaves your skin refreshed and renewed with a glow that radiates from within.","title":"SATURDAY SKIN | LinkedIn","query":"Saturday Skin","timestamp":"2021-11-23T19:46:21.392Z"},

Data Preparation: Data Cleanup

Removing NaN

```
def remove_nan_negatives_fast(df):
    non_numeric_columns = df.select_dtypes(include=[object]).columns
    df[non_numeric_columns] = df[non_numeric_columns].apply(lambda x: x.str.upper().str.strip())

    replacements = {np.inf: "",
                    -np.inf: "",
                    "-1": "",
                    "-1.0": "",
                    -1: "",
                    -1.0: "",
                    "NaN": "",
                    "NaT": "",
                    "nan": "",
                    "NAN": "",
                    "NAT": "",
                    "NONE": "",
                    "Nan": "",
                    ",": "", }

    df = df.replace(replacements).fillna("")
    return df
```

Removing Emojis From Tweets

```
import re
def remove_emojis(data):
    emoj = re.compile("["
        u"\U0001F600-\U0001F64F"  # emoticons
        u"\U0001F300-\U0001F5FF"  # symbols & pictographs
        u"\U0001F680-\U0001F6FF"  # transport & map symbols
        u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
        u"\U00002500-\U00002BEF"  # chinese char
        u"\U00002702-\U000027B0"
        u"\U00002702-\U000027B0"
        u"\U000024C2-\U0001F251"
        u"\U0001f926-\U0001f937"
        u"\U00010000-\U0010ffff"
        u"\u2640-\u2642"
        u"\u2600-\u2B55"
        u"\u200d"
        u"\u23cf"
        u"\u23e9"
        u"\u231a"
        u"\ufe0f"  # dingbats
        u"\u3030"
                      "]+", re.UNICODE)
    return re.sub(emoj, '', data)
```

```
remove_emojis(df['location'])
```

REGEX To Clean Tweet Text

```
#Converting bytes data to string data
df8["tweet_text"] = df8["tweet_text"].str.decode("utf-8")
```

```
#Data cleaning and creating a new column with cleaned textual data
clean_tweets = []
for key, tweet in df8["tweet_text"].iteritems():
    tweet = tweet.lower()
    tweet = re.sub(r"b","", tweet)
    tweet = re.sub("@[A-Za-z0-9_]+","", tweet)
    tweet = re.sub("#[A-Za-z0-9_]+","", tweet)
    tweet = re.sub(r"http\S+", "", tweet)
    tweet = re.sub(r"www.\S+", "", tweet)
    tweet = re.sub('[()!?]', ' ', tweet)
    tweet = re.sub('\[.*?\]',' ', tweet)
    tweet = re.sub("[^a-z0-9]"," ", tweet)
    tweet = ''.join(c for c in tweet if c not in emoji.UNICODE_EMOJI) #Remove Emojis
    tweet = tweet.replace("#", "").replace("_", " ") #Remove hashtag sign but keep the text
    tweet = " ".join(w for w in nltk.wordpunct_tokenize(tweet) \
         if w.lower() in words or not w.isalpha())
    clean_tweets.append(tweet)

df8["clean_tweet"] = clean_tweets
```

```
df8
```

	tweet_id	brand_id	tweet_date	follower_count	retweets	location	tweet_text	clean_tweet
0	1460799974095015939	1	2021-11-17 02:40:28+00:00	81	0	SAN DIEGO, CA	B"@RTCATH95 HE'S AWFFULLY STRANGE AINT HE?"	he s strange aint he

1. **REGEX:** Tweet text data includes *emoji, hashtag, numbers, links and etc* so REGEX is used to clean up each text
2. **NLTK:** NLTK tokenizing by word was used to extract more meaningful and logical words from cleaned tweet texts

Data Preparation: Data Cleanup Continued

```
prod_review = pd.read_csv('reviews2.csv')
prod_merge = pd.read_csv('products2.csv')
review_add = pd.read_csv('review_add.csv')
addition = pd.read_csv('addition.csv')
```

```
/Users/admin/opt/anaconda3/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3165: DtypeWarning: Columns (7) have mixed types.Specify dtype option
on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

```
prod = prod[['brand','category','name','size','rating','love','price','value_price','URL','details','how_to_use','ingredients']]
prod.columns = [x if x != 'URL' else 'url' for x in prod.columns.to_list()]
```

```
prod['product_id'] = prod['url'].str.split('grid:').str[1].str.upper()
```

```
prod_review.columns = ['user_nickname', 'rating', 'review_text', 'context_data_values','product_id','author_id','submission_time','is_recommended','content_l
prod_review = prod_review[['submission_id','product_id','rating','review_text','submission_time','content_locale','is_recommended','context_data_values']]

review_add.columns = ['user_nickname', 'rating', 'review_text', 'context_data_values','product_id','author_id','submission_time','is_recommended','content_lo
review_add = review_add[['submission_id','product_id','rating','review_text','submission_time','content_locale','is_recommended','context_data_values']]

prod_review = pd.concat([prod_review, review_add])
```

```
prod_merge.columns = ['description','product_id','image_url','product_name','review_statistics','total_reviews']

addition = addition[['Description', 'Id', 'ImageUrl', 'Name', 'ReviewStatistics','TotalReviewCount']]
addition.columns = ['description','product_id','image_url','product_name','review_statistics','total_reviews']

prod_merge = pd.concat([prod_merge, addition])
```

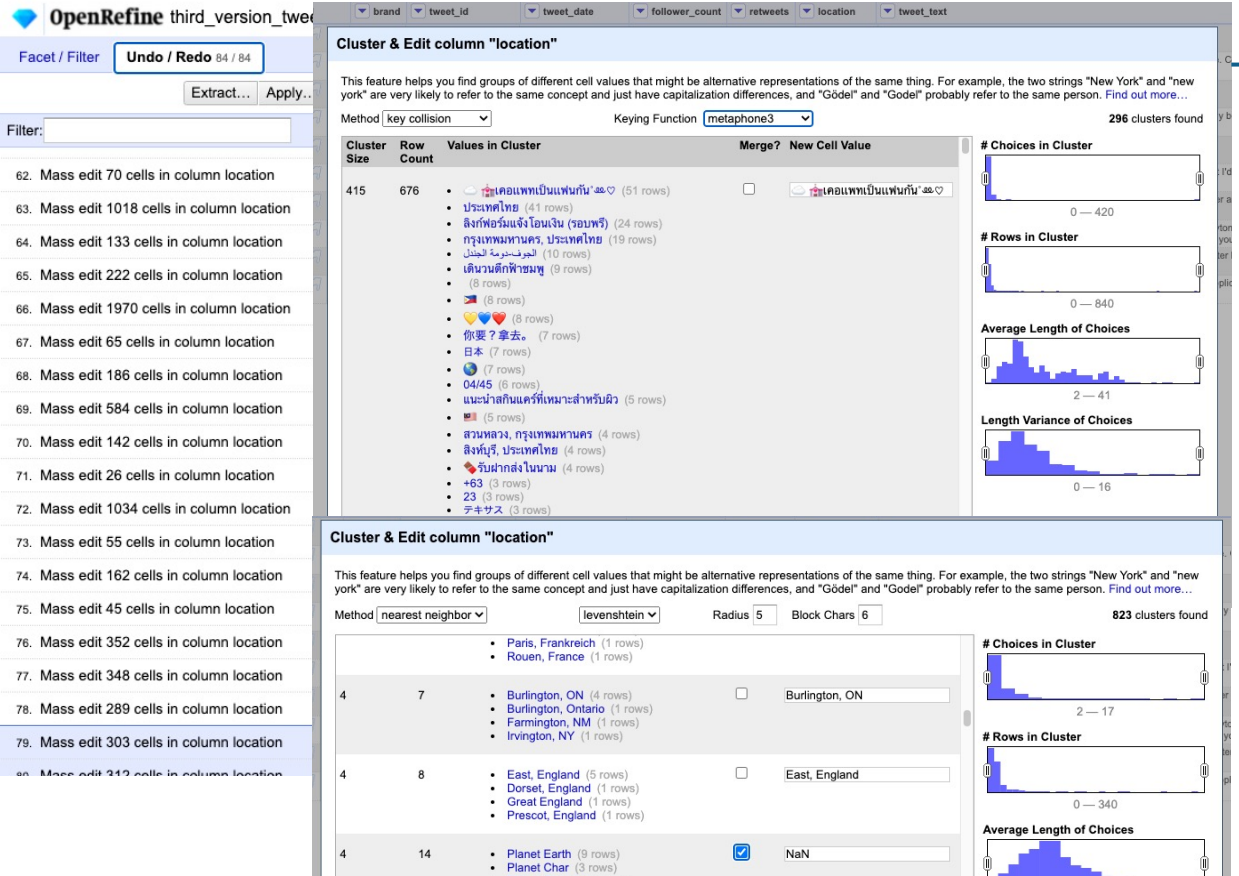
```
altmerged = prod.merge(prod_merge, how = 'left', left_on = 'product_id', right_on = 'product_id')
altmerged2 = prod.merge(prod_merge, how = 'left', left_on = 'name', right_on = 'product_name')

altmerged.dropna(thresh=14)
merged = altmerged.fillna(altmerged2)
```

```
altmerged.drop_duplicates('product_id').fillna('NULL').to_csv('review_product.csv', index = False)
prod_review.drop_duplicates().to_csv('review_review.csv', index = False)
```

Data Preparation For MySQL Workbench Import

Removing unused columns and filling NaNs as NULL – MySQL Workbench recognizes NULL instead of empty string

Data Preparation: OpenRefine



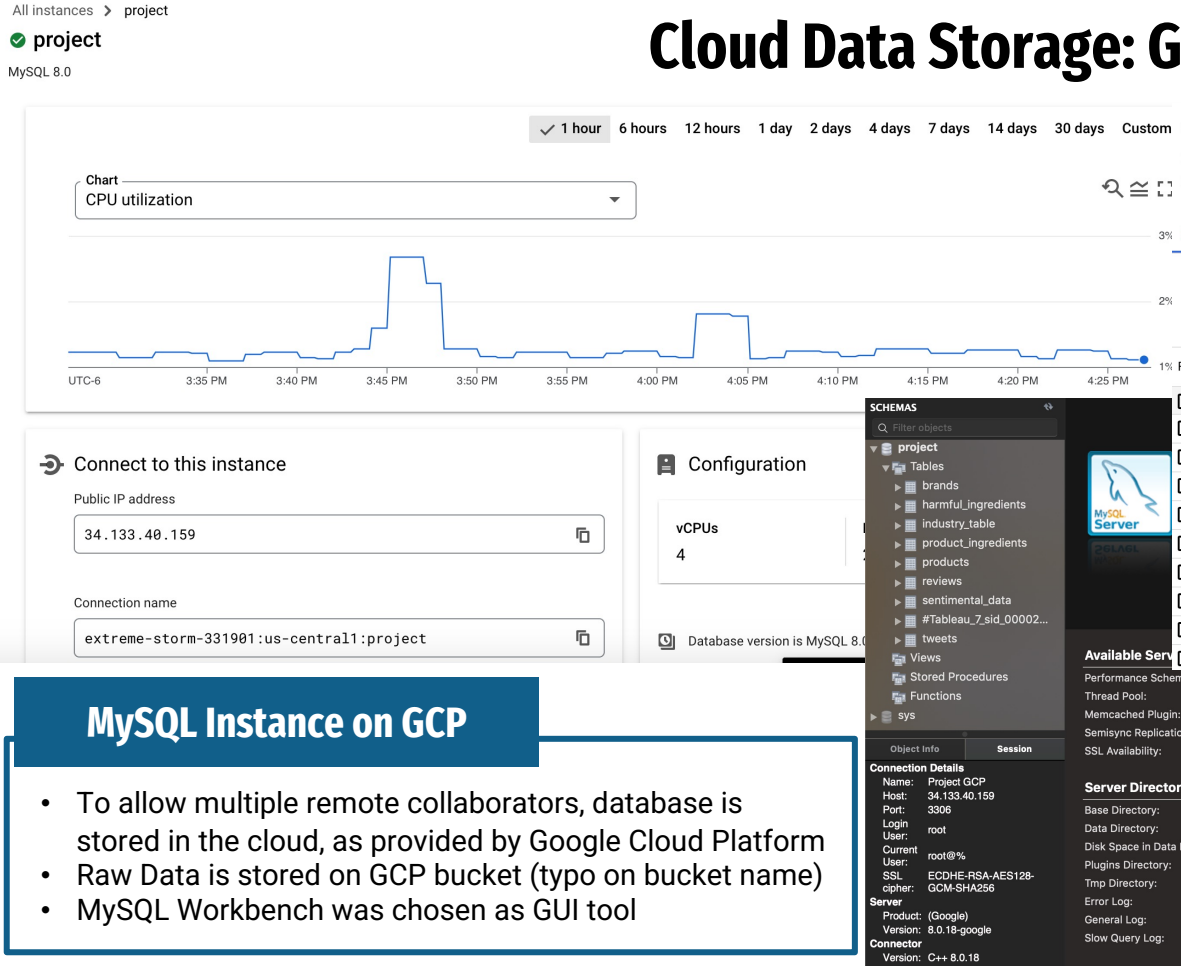
Clustering On Location Data From Tweets

Tweets' location is not structured so cleaning up the users' various input on location required heavy clustering that involved:

- Key Collision Methods
- Nearest Neighbor Methods
- Fingerprinting
- N-Gram Fingerprint
- Phonetic Fingerprint (**metaphone3** was the most effective in finding and clustering data that was not appropriate as location)

- For data that was not classified as location, " " was assigned.

Cloud Data Storage: GCP



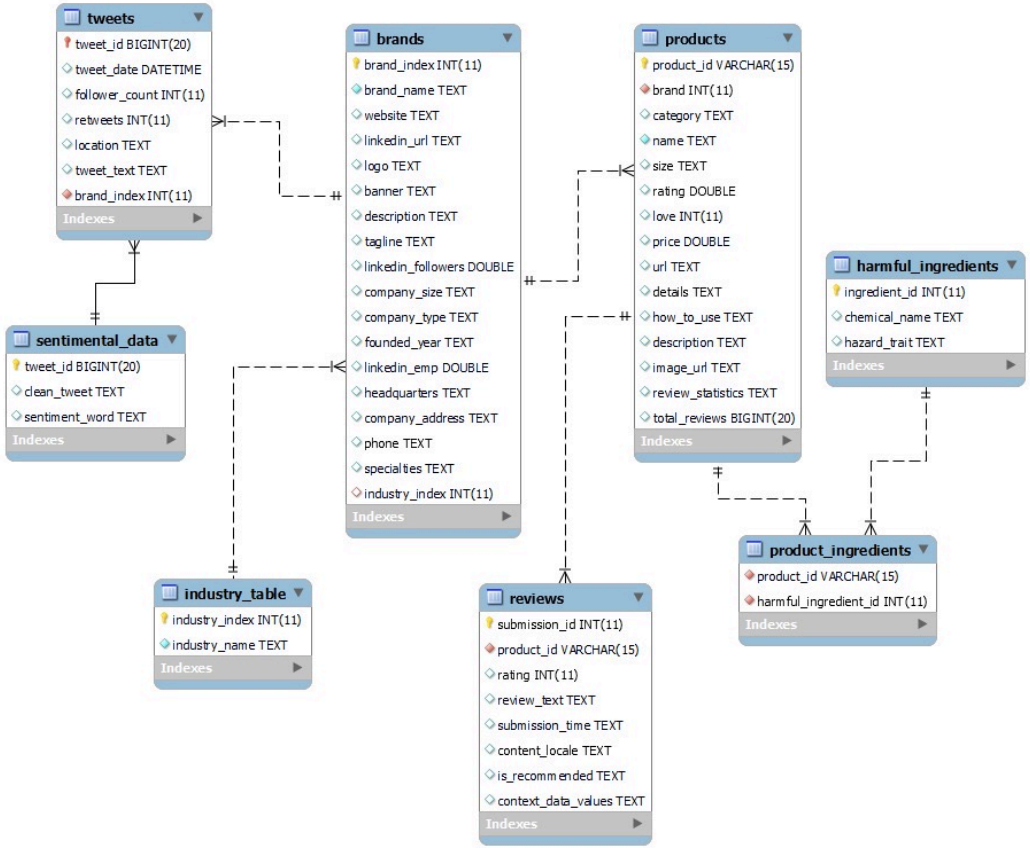
MySQL Instance on GCP

- To allow multiple remote collaborators, database is stored in the cloud, as provided by Google Cloud Platform
- Raw Data is stored on GCP bucket (typo on bucket name)
- MySQL Workbench was chosen as GUI tool

SQL: EER Model

OLTP

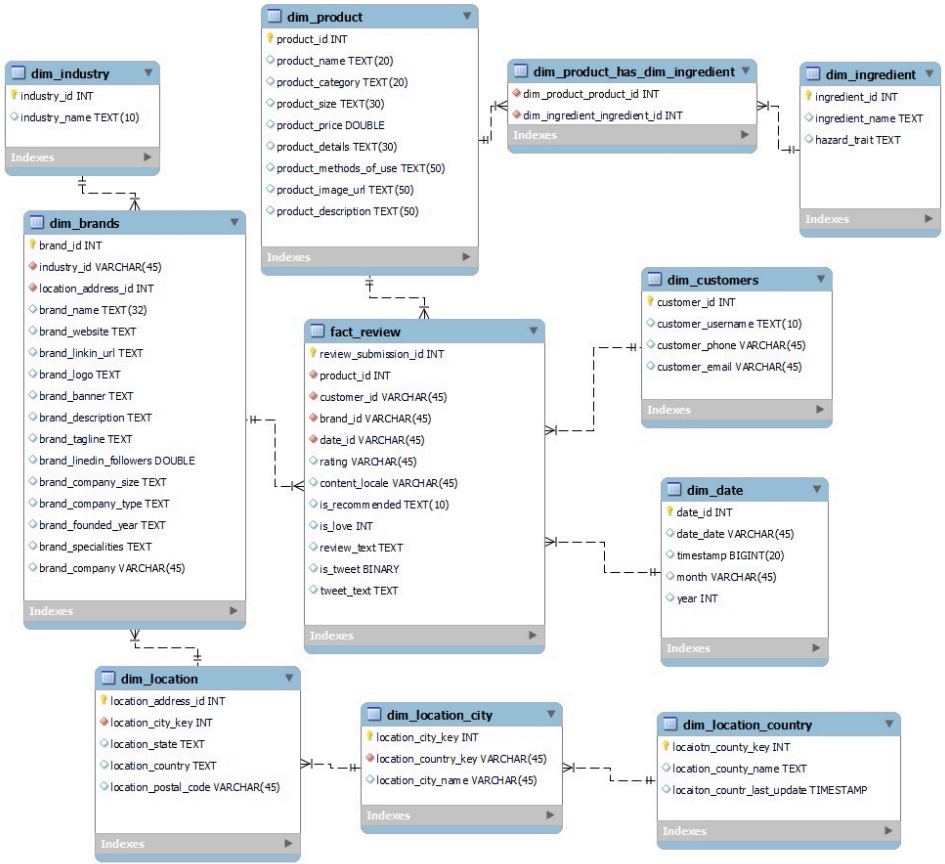
- There are 8 tables in the database
- After data collection and processing, import our data into MySQL Workbench and perform reverse enginnering to create the schema
- Define attributes with approriate data type
- Follow standard naming convention for tables and attributes
- Each table contains its unique key as primary key



SQL: Dimensional Model

OLAP

- Dimensional Model was developed based on the EER model constructed
- Used Snowflake Schema and normalized some dimension tables such as location table and product table
- “Reviews” is the fact table and there are 4 entities: customers, brands, product and date
- An extra entity, “dim_customers”, exists in dimensional diagram. Original data source included customer data such as customer username and personal info, but due to data privacy concerns, we decided not to include customer data in our project database



NoSQL: MongoDB – Reviews Customer Context

```
db.reviews_mongo.find({})
   .projection({})
   .sort({_id:-1})
   .limit(5)

db.reviews_mongo.find({}).limit(1)
db.reviews_mongo.find({ "skinType": "combination" })
```

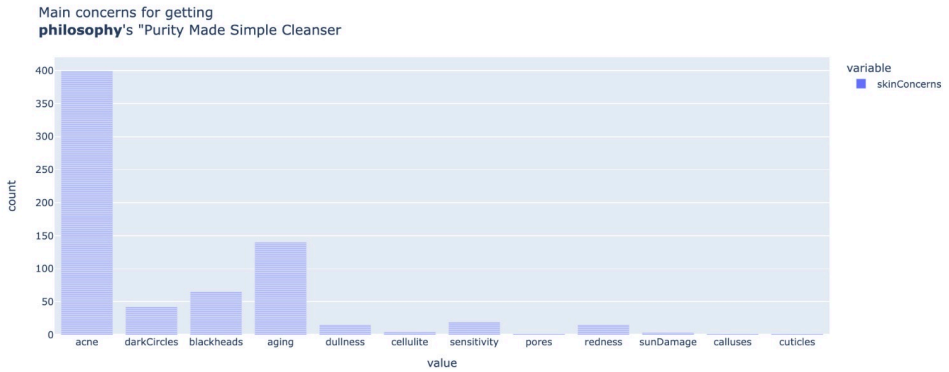
reviews_mongo 0.264 s 228,968 Docs 100 p. 1 1 - 100 Table

	_id	product_id	submission_id	skinType	eyeColor	StaffContext	hairColor	skinTone
1	61ae9dbf4e	P398965	190118856	combination	blue	false	blonde	light
2	61ae9dbf4e	P398965	186883060	combination	brown	false	brunette	light
3	61ae9dbf4e	P398965	185227544	combination	brown	false	black	dark
4	61ae9dbf4e	P398965	178037513	combination	brown	false	black	
5	61ae9dbf4e	P398965	177257921	combination	brown	false	blonde	
6	61ae9dbf4e	P398965	174681120	combination	blue	false	blonde	
7	61ae9dbf4e	P398965	172972178	combination	brown	false	black	
8	61ae9dbf4e	P398965	172925952	combination	green	false	brunette	
9	61ae9dbf4e	P398965	172613000	combination	blue	false	brunette	
10	61ae9dbf4e	P398965	167244498	combination	blue	false	blonde	
11	61ae9dbf4e	P398965	166535261	combination	green	false	red	
12	61ae9dbf4e	P398965	166421517	combination	hazel	false	brunette	
13	61ae9dbf4e	P398965	166267274	combination	blue	false	blonde	

Review Context & Metadata Stored

- MongoDB is used to store review context data
- This includes metadata on customer attributes such as their eye color, skin tone, if they are employees, etc

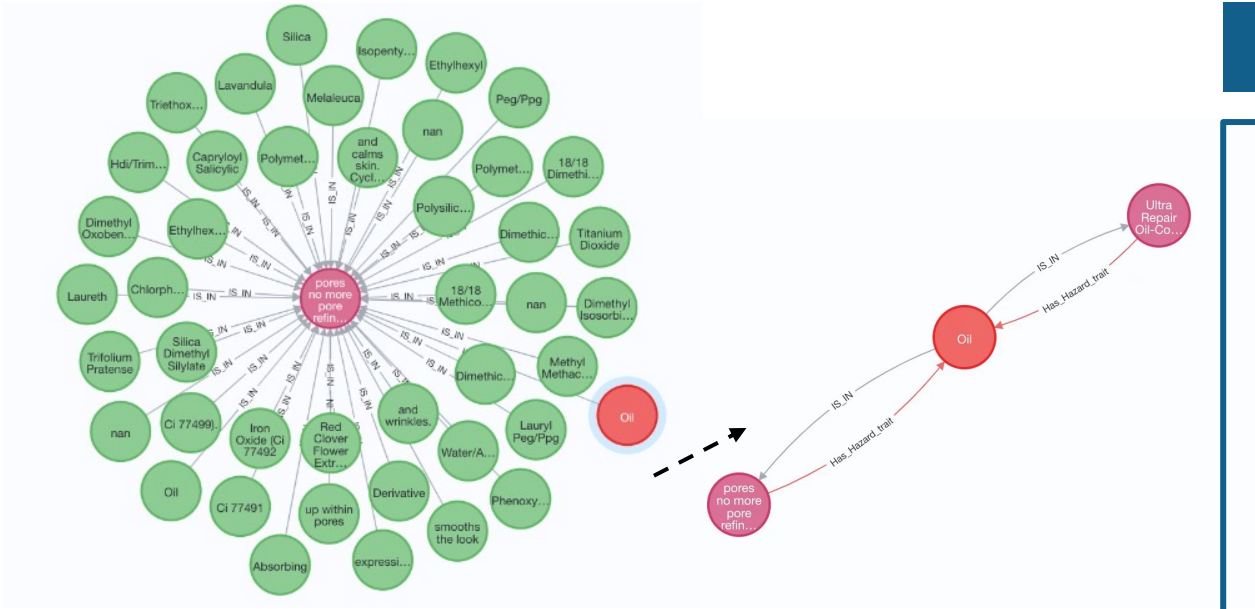
```
[12]: import pymongo
      from pymongo import MongoClient

[15]: client = MongoClient('mongodb://localhost:27017/project')
      db=client.project

[19]: mongodf = pd.DataFrame(list(db.reviews_mongo.find({ "skinType": "combination" })))

[22]: mongodf = mongodf.drop('_id', axis = 1)

[34]: px.bar(mongodf.query('product_id == "P7109"')['skinConcerns'].dropna(),
            title = f'Main concerns for getting <br><b>{brand}</b>\'s "{name}')
```



Nested data fit for document data type

- MongoDB is chosen because of their non-relational nature and compatibility with nested JSON
- Different products may have different attributes that may not make sense if put on other product categories. For example, skin type for lipstick

NoSQL: Neo4J - Product Ingredient Relationships In Graph



Neo4j

- Shows all the ingredients contained in the product
- If the ingredient has hazard traits, the node of this ingredient will be red.
- Choose an ingredient （or hazard ingredient) and we can see the products that contain it

GRAPH1

(a:Ingredient)-[:IS_IN]->(b:PRODUCT)

GRAPH2

(p:PRODUCT)-[:Has_Hazard_trait]->(h:Hazard_Ingredient)

Five Business Use Cases With Analyses

1

Branding

Quantify brand perception in the form of textual data

Output

NLP powered sentimental analysis

2

Key Performance

Look at similar products to evaluate product performance

Output

Tableau dashboards at brand and product level

3

Ingredients

Suggest products without harmful ingredients

Output

Neo4j graph & tkinter application

4

Competitor Lookup

Perform a competitor analysis on brands and their product profile

Output

Tableau dashboard with brand logo & site rendering

5

Product Reviews

Gaining insights based on user product review

Output

Analyses using Plotly, Pandas, ggplot

Case 1: Tweet Sentimental Analysis Using TextBlob And NLP

```
#Importing emoji and nltk for data cleaning process
import emoji
import nltk
import re
nltk.download('words')
words = set(nltk.corpus.words.words())
```

```
[nltk_data] Downloading package words to /Users/aj/nltk_data...
[nltk_data]   Package words is already up-to-date!
```

```
# Import TextBlob for text based sentimental analysis
from textblob import TextBlob
```

```
#Perform sentimental analysis using TextBlob
def sentiment_func(tweet):
    try:
        return TextBlob(tweet).sentiment
    except:
        return None
```

```
df8["sentiment"] = df8["clean_tweet"].apply(sentiment_func)
df8["sentiment"].head()
```

```
0                                (-0.05, 0.15)
1     (0.10666666666666666, 0.3516666666666667)
2                                 (0.35, 0.65)
3                                   (0.0, 0.4)
4    (0.06818181818181818, 0.22727272727272727)
Name: sentiment, dtype: object
```

```
#Divide the sentimental analysis figures into two columns, polarity and subj
df8["sentiment"][0][0]
df8['polarity'] = df8["sentiment"].apply(lambda x: x[0])
df8['subjectivity'] = df8["sentiment"].apply(lambda x: x[1])
df8[['clean_tweet','sentiment','polarity','subjectivity']].head()
```

	clean_tweet	sentiment	polarity	subjectivity
0	he s strange aint he	(-0.05, 0.15)	-0.050000	0.150000
1	lyse is a solid not entirely fantasy sounding ...	(0.10666666666666666, 0.3516666666666667)	0.106667	0.351667
2	true let s terrorize it	(0.35, 0.65)	0.350000	0.650000
3	if i were to go with something related it woul...	(0.0, 0.4)	0.000000	0.400000
4	is your new middle name	(0.06818181818181818, 0.22727272727272727)	0.068182	0.227273

```
#Assign words to polarity column
sentiment_word = []
for key, value in df8["polarity"].iteritems():
    if value > 0:
        value ='positive'
    elif value == 0:
        value ='neutral'
    else:
        value ='negative'
    sentiment_word.append(value)
df8["sentiment_word"] = sentiment_word
df8[['clean_tweet','sentiment_word']]
```

	clean_tweet	sentiment_word
0	he s strange aint he	negative
1	lyse is a solid not entirely fantasy sounding ...	positive
2	true let s terrorize it	positive
3	if i were to go with something related it woul...	neutral
4	is your new middle name	positive
...	...	...
20066	so good i had to share check out all the i m l...	positive
20067	so good i had to share check out all the i m l...	positive
20068	check out this listing i just added to my clos...	neutral
20069	everything you need to know the stay all day x...	neutral
20070	all day liquid in patina does not come off on ...	neutral

NLP powered tweet analysis

- Sentiment analysis is the task of determining the emotional value of a given expression in natural language.
- Using collected tweets data per brand, perform sentimental analysis using TextBlob
- **Textblob** (NLP Api) sentiment analyzer returns two properties for a given input sentence:

1. **Polarity:** -1 is negative, +1 is positive sentiment
2. **Subjectivity:** Subjective sentences generally refer to personal opinion, emotion, or judgment.

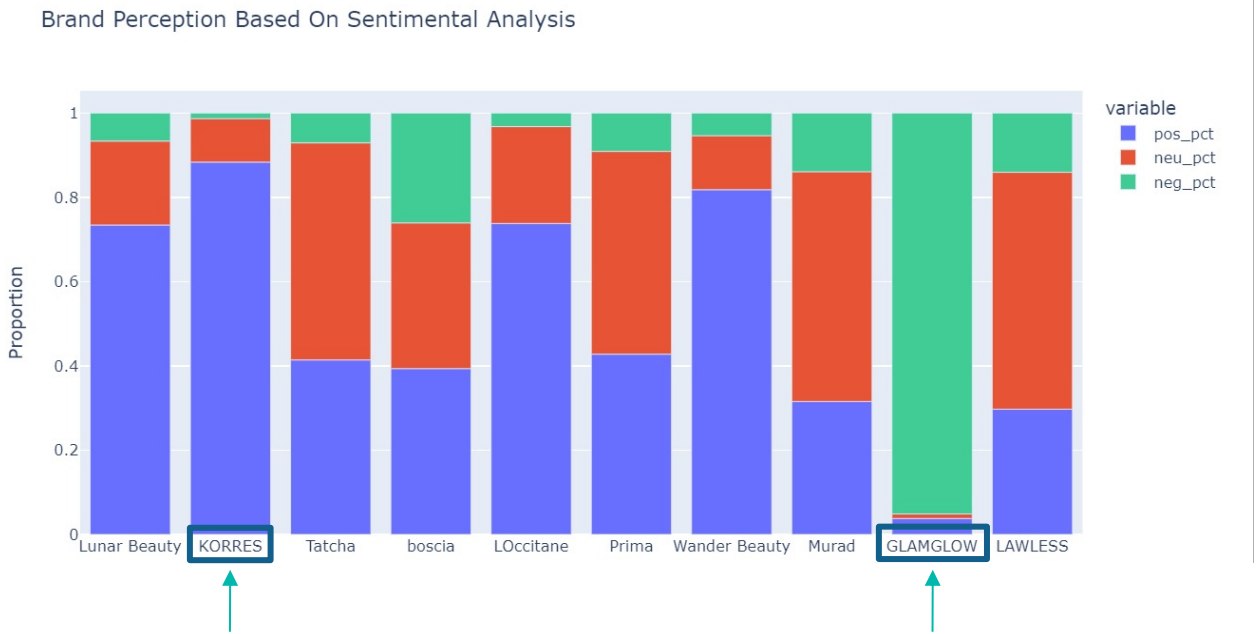
- Use polarity to assign word to each tweet: positive, negative and neutral

Case 1: Brand Perception Based On Sentiment Of Tweets

```
In [17]: fig = px.bar(merged, y=['pos_pct','neu_pct','neg_pct'],  title="Plot Title")
         fig.update_layout(title="Brand Perception Based On Sentimental Analysis", yaxis_title="Proportion")
```

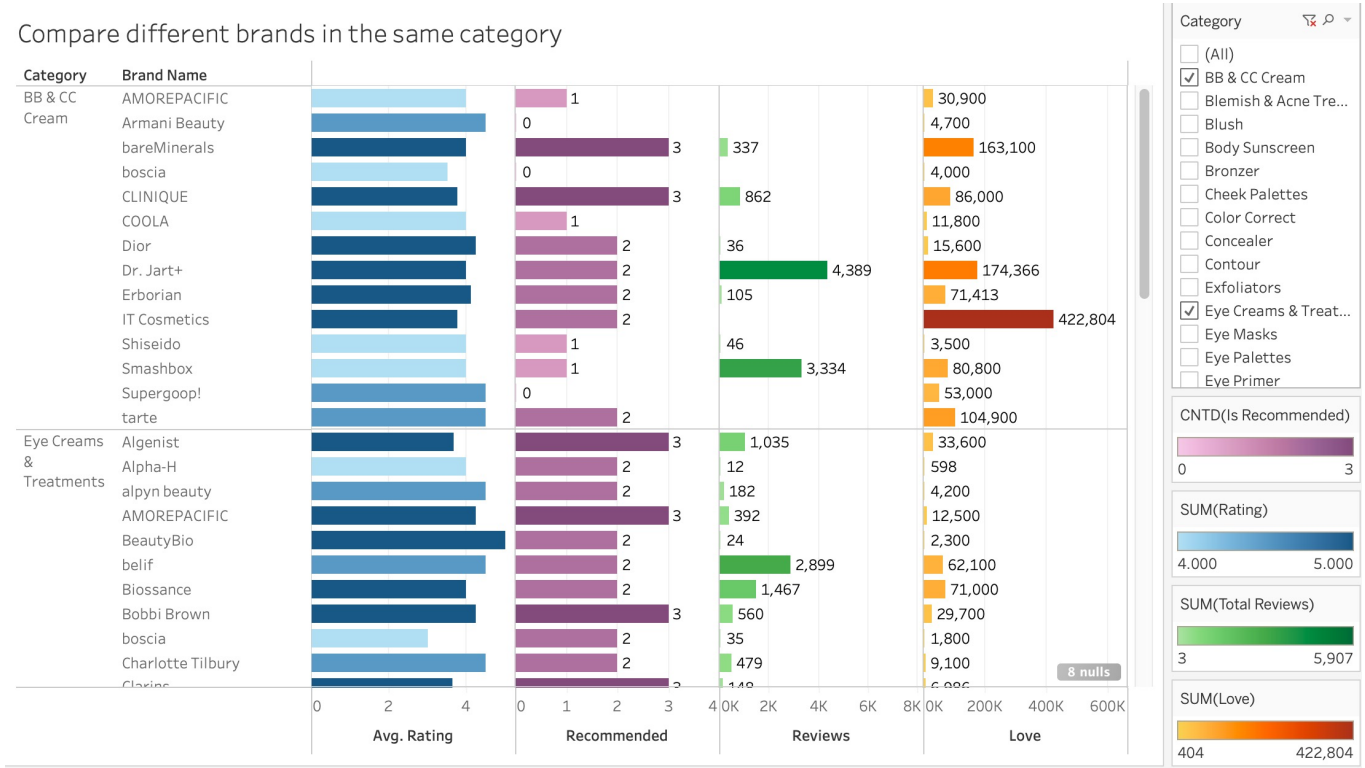
Clients can get insights from sentimental analysis of customers' tweets

- This graph shows the relative perception of brands based on tweet sentiment analysis
- Again, tweets are processed through an NLP model and classified if they were either: Positive, Neutral or Negative in sentiment
- For example, **GLAMGLOW** has received a lot of flak and bad press
- **KORRES** on the other hand has generated a lot of positive buzz on Twitter



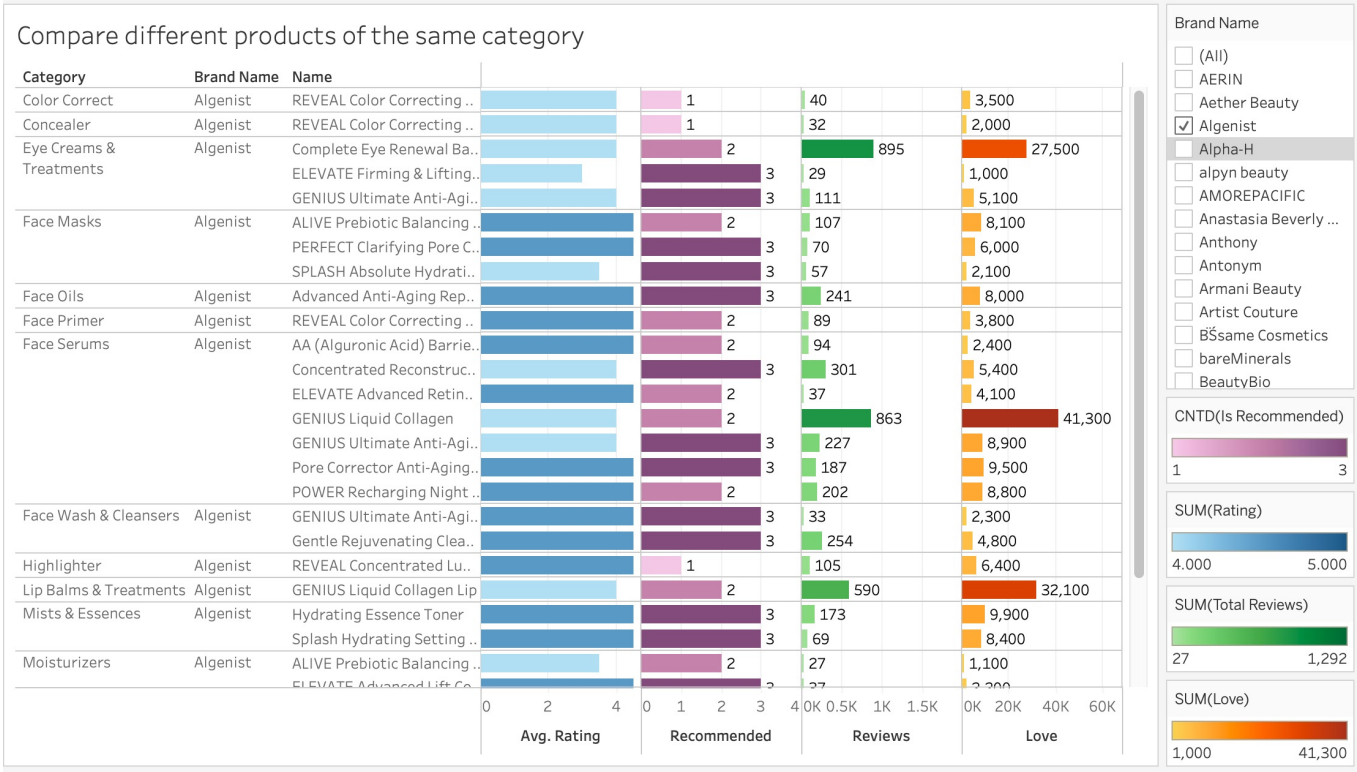
Case 2: Key Performance

Brand Level



- This graph shows the comparison of different brands in the same category
- Included metrics are average ratings, number of re commended , review, and love.
- The darker the color of the bar, the larger the number

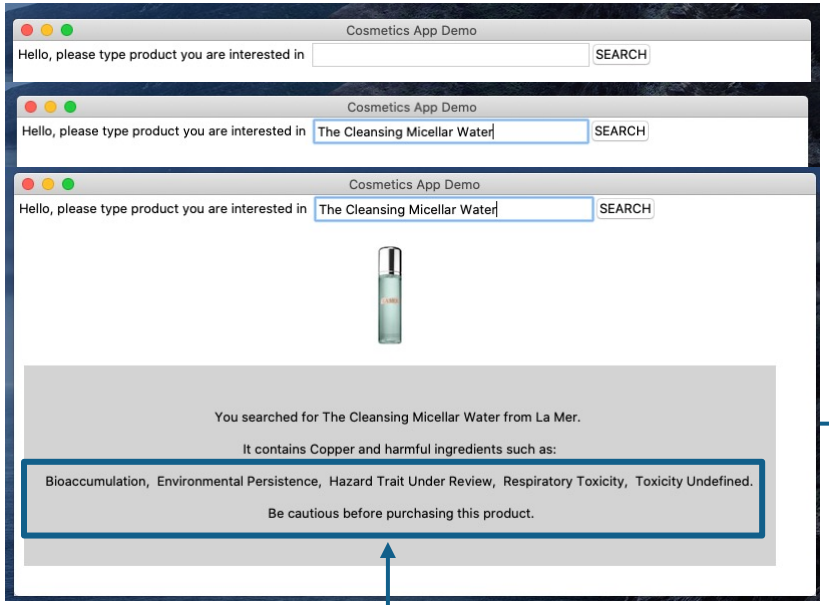
Case 2:
Key Performance



Product Level

- This graph shows the comparison of different products of the same brand through average ratings, number of recommended, reviews, and love.
- The darker the color of the bar, the larger the number associated with it

Case 3: User Journey On Demo App Built On Tkinter (Python GUI)



1. User opens the application
2. User searches for the interested product
3. Search query result is displayed with brand logo and product photo

Shows the product's ingredients and whether it contains harmful ingredients with a kind warning.
Users can get comprehensive information on interested products' ingredients and whether it contains any harmful substance

```
def clicked():
    my_conn.execute(f"""SELECT b.brand_name, p.name, h.chemical_name, h.hazard_trait, p.image_url, b.logo
    FROM products p JOIN product_ingredients i ON p.product_id = i.product_id
    JOIN harmful_ingredients h ON h.ingredient_id = i.harmful_ingredient_id
    JOIN brands b ON b.brand_index = p.brand
    WHERE p.name LIKE '%{txt.get()}%' AND h.hazard_trait IS NOT NULL LIMIT 1;""")
    result = my_conn.fetchone()
```

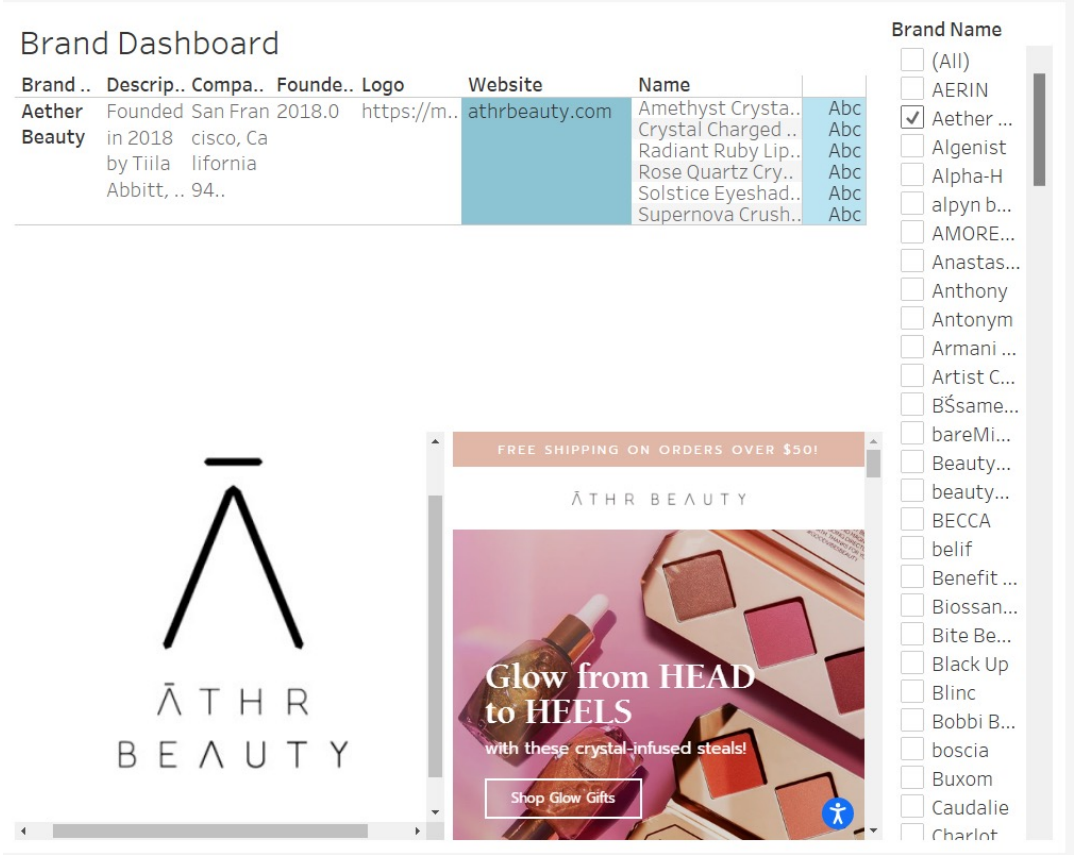
Business Case

- Convert users to purchasing products by becoming go-to place for searching cosmetics products
- Affiliate revenue is sizable with growing user base

Case 4: Competitor Lookup

Clients can get a comprehensive overview of competitor brands

- Used Tableau to create a Brand Dashboard
- Dashboard uses brand name as a filter so that the user can choose which brand to look into
- Dashboard contains information like brand description and their products
- Dashboard also shows the brand logo and its official website if present

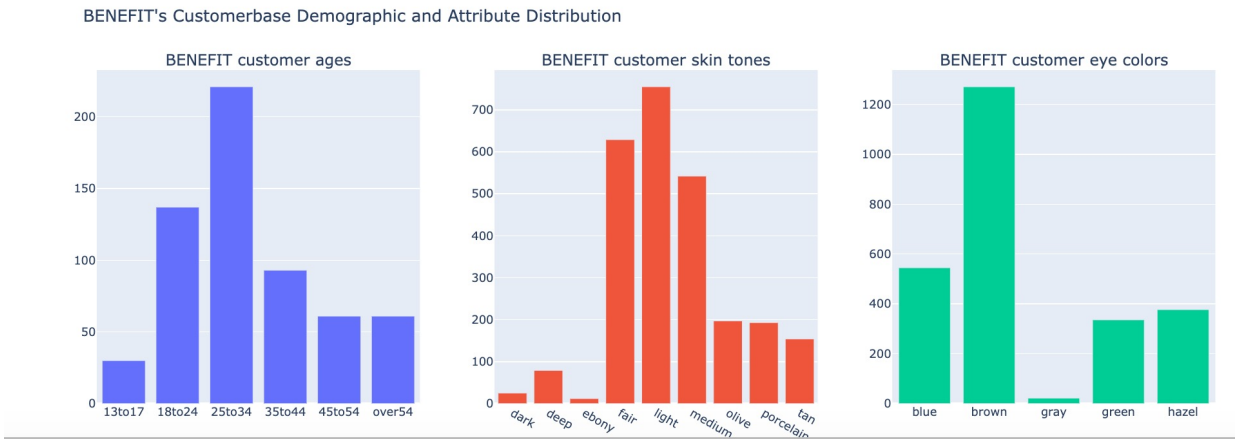


Case 5: Customer Demographic and Attribute Distribution

Clients can understand customers based on user review analysis

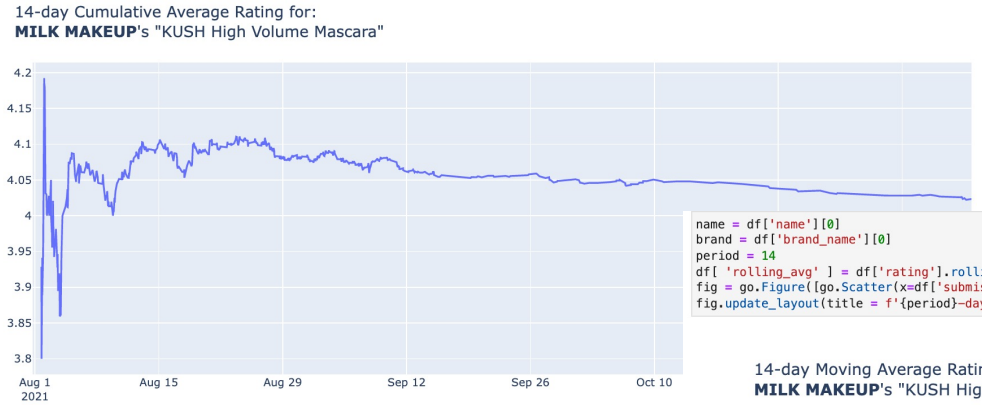
- The brand's main audience are between 18 to 34, although there are also customers from other age bands
- Most customers are lighter skinned
- Eye color distribution show that they have an area of improvement where they can release products that may match better with gray eyes

```
fig = make_subplots(rows=1, cols=3,
                    subplot_titles=("BENEFIT customer ages", "BENEFIT customer skin tones", "BENEFIT customer eye colors")
                    )
df = getMetric('age')[1].sort_values('index')
fig.add_trace(
    go.Bar(x=df['index'], y=df['count'], name='count'),
    row=1, col=1
)
df = getMetric('skinTone')[1].sort_values('index')
fig.add_trace(
    go.Bar(x=df['index'], y=df['count'], name='count'),
    row=1, col=2
)
df = getMetric('eyeColor')[1].sort_values('index')
fig.add_trace(
    go.Bar(x=df['index'], y=df['count'], name='count'),
    row=1, col=3
)
fig.update_layout(title_text = "BENEFIT's Customerbase Demographic and Attribute Distribution", showlegend=False)

fig.show()
```

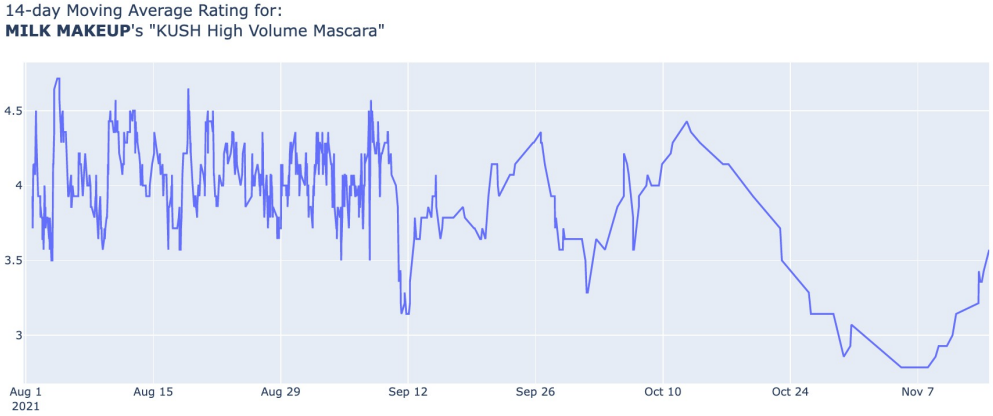


Case 5: Average Product Rating Over Time

```
df[ 'cum_avg' ] = df['rating'].expanding(period).mean()
fig = go.Figure([go.Scatter(x=df['submission_time'], y=df['cum_avg'])])
fig.update_layout(title = f'{period}-day Cumulative Average Rating for:<br><b>{brand}</b>\'s "{name}"')
```




```
name = df['name'][0]
brand = df['brand_name'][0]
period = 14
df[ 'rolling_avg' ] = df['rating'].rolling(period).mean()
fig = go.Figure([go.Scatter(x=df['submission_time'], y=df['rolling_avg'])])
fig.update_layout(title = f'{period}-day Moving Average Rating for:<br><b>{brand}</b>\'s "{name}"')
```



Some products might be seasonal →

- Product review started off strong, but tapers off with slower frequency of review and dropping average rating value
- Might suggest that product is not as good as expected or longer-term issues with the product that wasn't immediately apparent (e.g., weather incompatibility)

Case 5: Product for Specific Area of Concern

Applying MongoDB for visualization

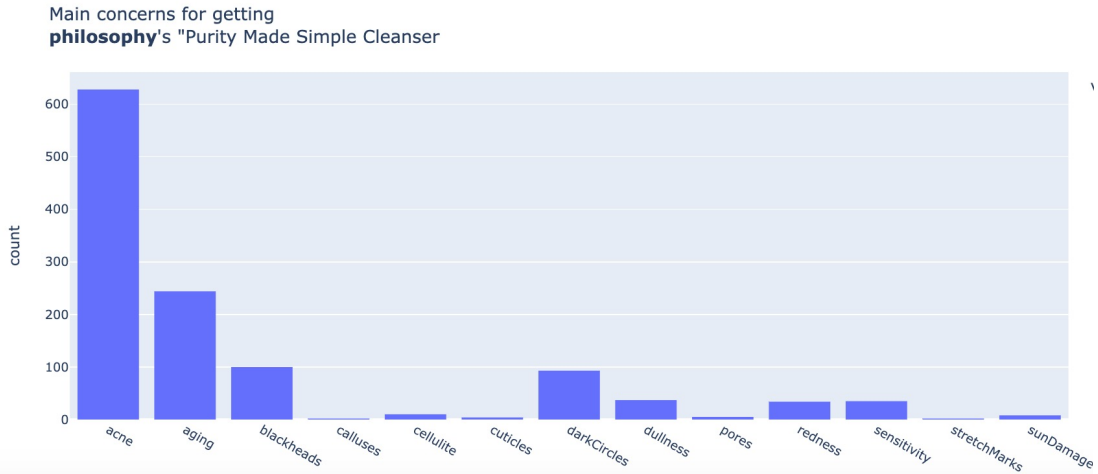
Find the most common use case for a specific product ¶

```
client = MongoClient('mongodb://localhost:27017/project')
db=client.project
```

```
mongodf = pd.DataFrame(list(db.reviews_mongo.find({ })))
```

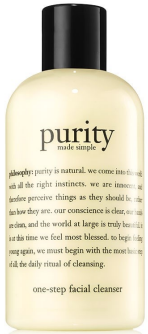
```
mongodf = mongodf.drop('_id', axis = 1)
```

```
brand = 'philosophy'
name = 'Purity Made Simple Cleanser'
px.histogram(mongodf.query('product_id == "P7109"')['skinConcerns'].dropna().sort_values(ascending=True),
       title = f'Main concerns for getting <br><b>{brand}</b>\'s "{name}')
```



Review data can tell clients where users find the product most useful

- "Purity Made Simple Cleanser" was made for acne problems in mind
- May also help with aging skin
- Not a good fit for customers who are looking to fix stretch marks or sun damage
- Brands can know consumers' pain points and use insights to target marketing and improve formula to strengthen the product positioning
- Consumer can make better informed decision about their intended purchasing before paying



philosophy®

Case 5: Top Products with Certain Attributes In Mind

Connecting MongoDB, pandas and MySQL

Top products for specific skin type, skin tone and eye color

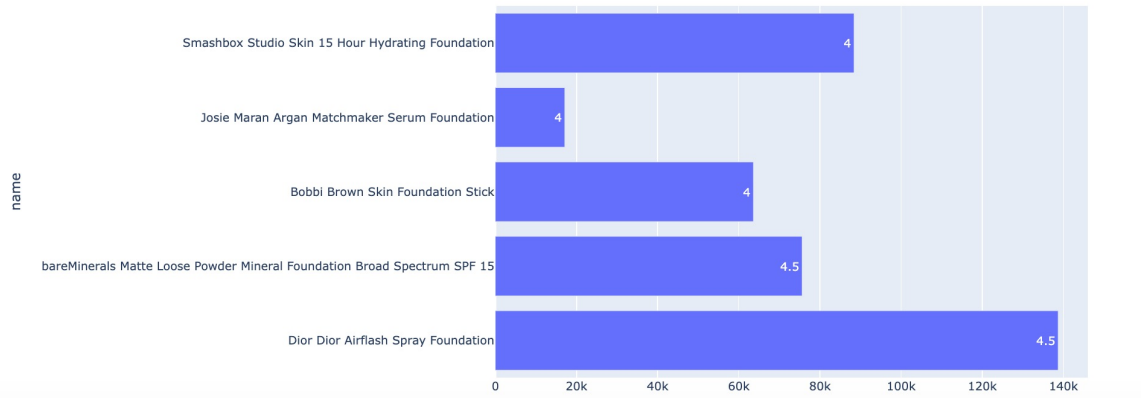
First, get the products with the right user attibute. Using the product id, find the product details on SQL database. Finally, plot a graph showing the products rating and "love" statistic

```
mongodf2 = pd.DataFrame(list(db.reviews_mongo.find({"$and" : [{ "skinType": "combination" }, {"skinTone" : "light"}, {"eyeColor" : "blue" }]})))
```

```
plist = mongodf2.groupby('product_id').count()['_id'].sort_values(ascending = False).index.to_list()
```

```
topdf = qry(f"""SELECT CONCAT(b.brand_name, " ", name) as name ,rating, love
FROM products p, brands b
WHERE b.brand_index = p.brand AND category like "foundation" AND product_id IN('{"', '".join([str(x) for x in plist])}')""").head(5)
```

```
px.bar(topdf, x = 'love', y='name', text='rating')
```



→ Review data can tell clients all time favourites based on users' attributes

- For undecided customers, they can first look up for items that are often used by others who share similar attributes
- For example, they may have a "combination" skin type, light skin tone, and blue eye color
- This insight can help them choose the right product in a specific category (e.g., foundation)
- Graph shows each product's "love" rating (a measure of how many times customers "love" a product), and their numeric rating

Lessons Learned & Recommendations

Lessons Learned

- Larger datasets require paid API calls so data costs should be considered more carefully in data planning process
- Research API restrictions in data collection is essential as expectation and reality may differ (ie. Tweepy api gives only recent tweets)
- Heavy web scraping needs ways to bypass access restrictions (ie. Proper header and referrer required to access Sephora website)
- JavaScript is not rendered with only Python requests
- Data clean-up is indispensable, yet requires the most amount of time investment after data collection
- Using a balance of MySQL and NoSQL based on data types improves data handling and ease of producing analyses
- Expectations on data completeness should be based on data availability and must be managed early on
- GCP allows for synchronized working environment for team

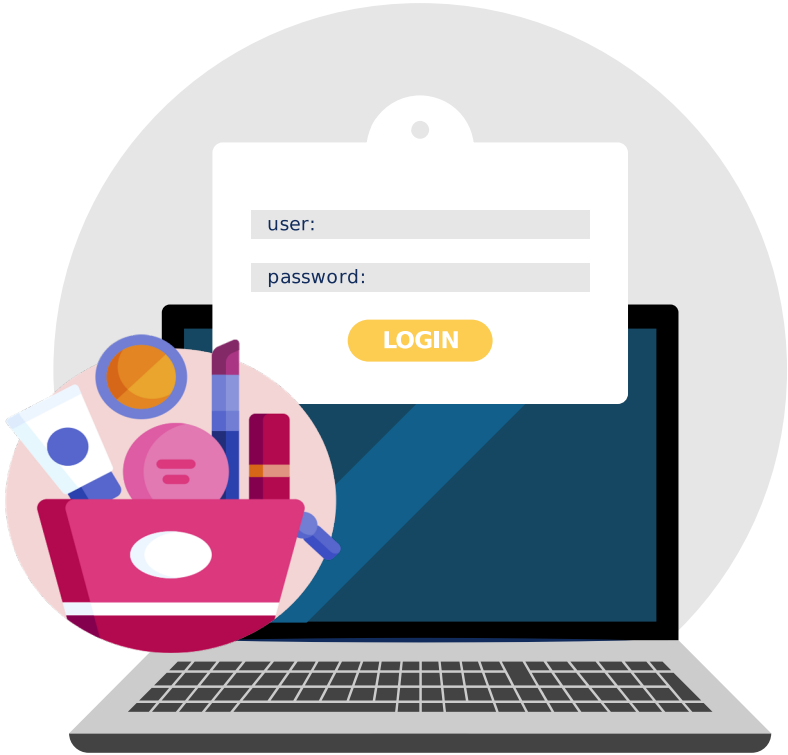
Recommendations

- **Data Scope:** Solidifying which metrics to prioritize for data collection and defining the data size should be done during initial data modelling
- **Data Parity:** Data availability of product review data and brands' tweets varied per product and brand
- **Unused Data:**

1. Analysis was limited to harmful ingredients, but good ingredients could have been used for new insights (ie. Pair the effect of ingredients and product reviews' user concerns)
2. Tweet location data was cleaned but unused; analysis around where negative sentiments on brands prevail could help brands target their marketing efforts

- **Overall, more research on industry context could have helped improve data modelling process**

Thank you



References

< Data Sources >
Sephora Products: https://www.kaggle.com/raghadalharbi/all-products-available-on-sephora-website
Harmful Ingredients: https://data.world/chhs/596b5eed-31de-4fd8-a645-249f3f9b19c4
Tweets: https://twitter.com/
Product Reviews: https://www.sephora.com/
Brands Information: https://www.linkedin.com
< Tools Used >
TextBlob: https://textblob.readthedocs.io/en/dev/
TKinter: https://docs.python.org/3/library/tkinter.html
BazaarVoice API: https://developer.bazaarvoice.com/conversations-api/home
Phantum Buster: https://phantombuster.com/
Tweepy: https://www.tweepy.org/
Selenium: https://www.selenium.dev/
Plotly: https://plotly.com/
Nltk: https://www.nltk.org/
PPTSlideTemplates: Slidego, https://slidesgo.com/