Our datasets

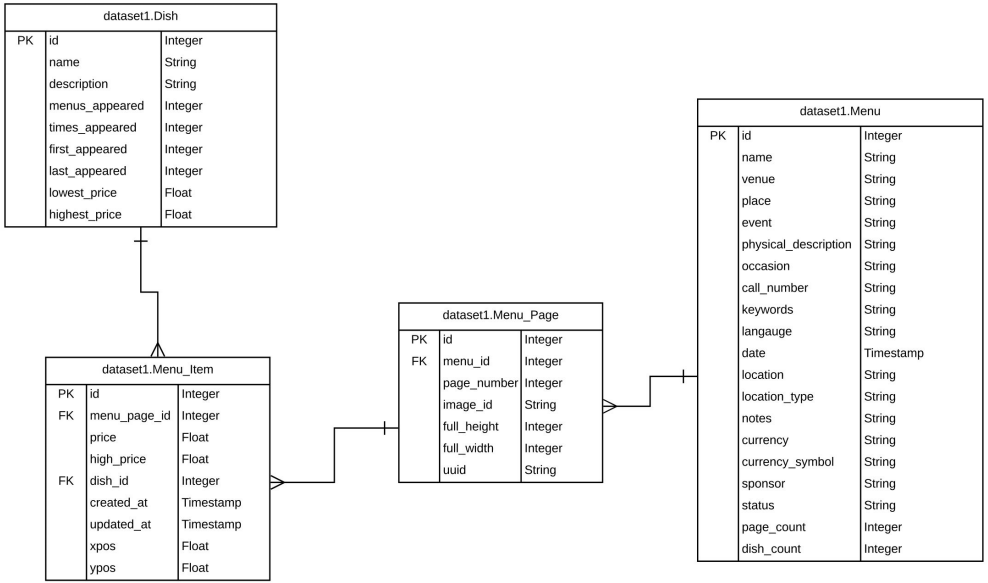
Dataset 1 - New York Public Library Restaurant Menus

- Tables:
  - Dish
  - Menu Item
  - Menu Page
  - Menu

Dataset 2 - Yelp Review Data

- Tables:
  - Business
  - User
  - Review

ERD - v1


dataset1.Dish
PK id Integer
name String
description String
menus_appeared Integer
times_appeared Integer
first_appeared Integer
last_appeared Integer
lowest_price Float
highest_price Float
dataset1.Menu_Item
PK id Integer
FK menu_page_id Integer
price Float
high_price Float
FK dish_id Integer
created_at Timestamp
updated_at Timestamp
xpos Float
ypos Float
dataset1.Menu_Page
PK id Integer
FK menu_id Integer
page_number Integer
image_id String
full_height Integer
full_width Integer
uuid String
dataset1.Menu
PK id Integer
name String
venue String
place String
event String
physical_description String
occasion String
call_number String
keywords String
langauge String
date Timestamp
location String
location_type String
notes String
currency String
currency_symbol String
sponsor String
status String
page_count Integer
dish_count Integer

Our questions

- Were ratings of a restaurant the same between a user and their friends?
- Does having more items on a menu correlate with lower ratings?
- Does having fewer dishes on a menu page correlate with higher ratings?
- Do ratings of a restaurant have anything to do with whether the menu is vertical or horizontal?

Dataset1 Transformations

Apache Beam:

- Price range of food item
  - Calculate range between price and high price for a menu item
- Orientation of Menu
  - Determine whether a menu is horizontally or vertically oriented
- Average Number of Dishes per Page of Menu
  - Calculate the average number of dishes per page from the Menu's dish count and page count

SQL:

- Currency table
  - Remove currency information from Menu, create new table
- Location table
  - Remove location information from Menu, create new table

Dataset2 Transformations

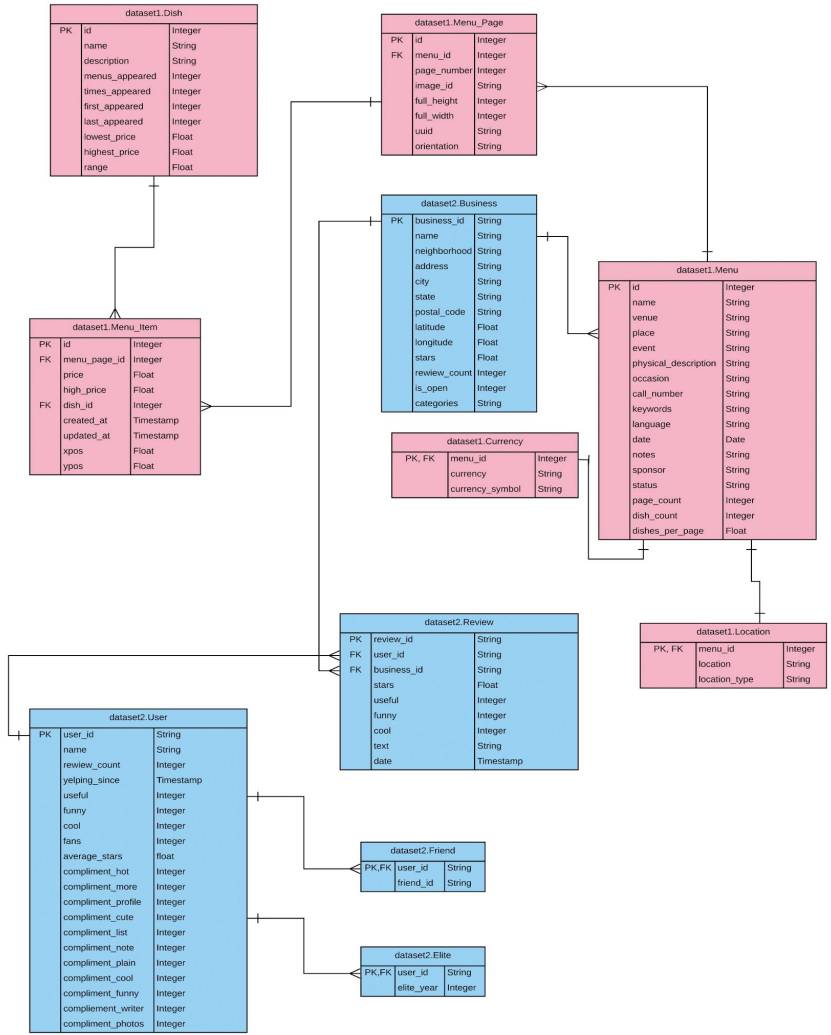
Apache Beam:

- Business Table
  - Transform Business name to match the formatting in the NYPL dataset
- Elite Table
  - Transform Elite Year list in User into its' own table
- Friend Table (demo)
  - Transform Friend list in User into its' own table
  - https://bit.ly/2V8k8Ap

SQL:

- User table changes
  - Remove Friend & Elite

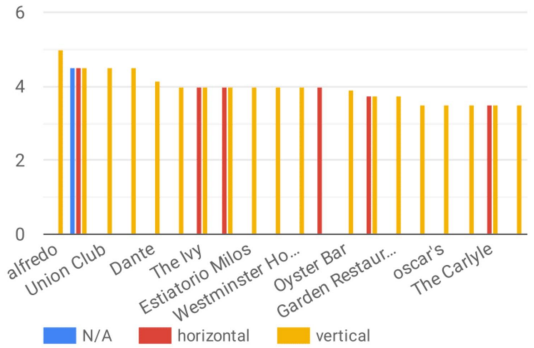
ERD - v5


dataset1.Dish
PK id Integer
name String
description String
menus_appeared Integer
times_appeared Integer
first_appeared Integer
last_appeared Integer
lowest_price Float
highest_price Float
range Float
dataset1.Menu_Page
PK id Integer
FK menu_id Integer
page_number Integer
image_id String
full_height Integer
full_width Integer
uuid String
orientation String
dataset2.Business
PK business_id String
name String
neighborhood String
address String
city String
state String
postal_code String
latitude Float
longitude Float
stars Float
rewiew_count Integer
is_open Integer
categories String
dataset1.Menu
PK id Integer
name String
venue String
place String
event String
physical_description String
occasion String
call_number String
keywords String
language String
date Date
notes String
sponsor String
status String
page_count Integer
dish_count integer
dishes_per_page Float
dataset1.Menu_Item
PK id Integer
FK menu_page_id Integer
price Float
high_price Float
FK dish_id Integer
created_at Timestamp
updated_at Timestamp
xpos Float
ypos Float
dataset1.Currency
PK, FK menu_id Integer
currency String
currency_symbol String
dataset2.Review
PK review_id String
FK user_id String
FK business_id String
stars Float
useful Integer
funny Integer
cool Integer
text String
date Timestamp
dataset1.Location
PK, FK menu_id Integer
location String
location_type String
dataset2.User
PK user_id String
name String
rewiew_count Integer
yelping_since Timestamp
useful Integer
funny Integer
cool Integer
fans Integer
average_stars float
compliment_hot Integer
compliment_more Integer
compliment_profile Integer
compliment_cute Integer
compliment_list Integer
compliment_note Integer
compliment_plain Integer
compliment_cool Integer
compliment_funny Integer
compliement_writer Integer
compliment_photos Integer
dataset2.Friend
PK,FK user_id String
friend_id String
dataset2.Elite
PK,FK user_id String
elite_year Integer

Cross-Dataset Findings

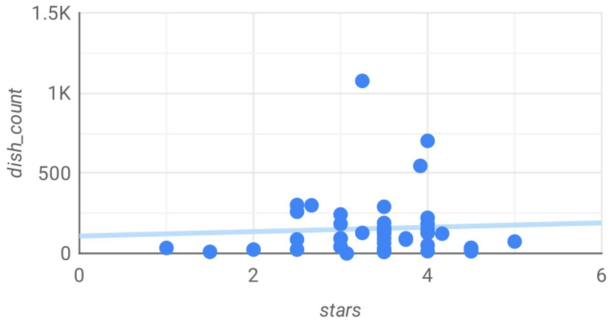
```
SELECT b.newname, b.stars, mp.orientation
FROM `dogwood-theorem-230119.dataset1.Menu` m
JOIN `dogwood-theorem-230119.dataset2.Business_1` b on UPPER(m.place) = UPPER(b.newname)
JOIN `dogwood-theorem-230119.dataset1.Menu_Page` mp on mp.menu_id = m.id
ORDER BY b.stars DESC
```

Menu Page Orientation vs. Average Star Rating for the Restaurant




```
SELECT b.newname, m.dish_count, b.stars
FROM `dogwood-theorem-230119.dataset1.Menu` m
JOIN `dogwood-theorem-230119.dataset2.Business_1` b on UPPER(m.place) = UPPER(b.newname)
WHERE dish_count is not null
```

Dishes on a Menu vs. Average Star Rating for the Restaurant



Airflow

```
# DAG section
with models.DAG('workflow',
        schedule_interval=datetime.timedelta(days=1),
        default_args=default_dag_args) as dag:

    # Beam Tasks
    business_beam = BashOperator(
            task_id='business_beam',
            bash_command='python /home/stoddartisabel/' + business_script)

    elite_beam = BashOperator(
            task_id='elite_beam',
            bash_command='python /home/stoddartisabel/' + elite_script)

    friend_beam = BashOperator(
            task_id='friend_beam',
            bash_command='python /home/stoddartisabel/' + friend_script)

    # SQL Task
    create_user_table = BashOperator(
            task_id='create_user_table',
            bash_command=sql_cmd_start + '"' + sql_user + '"')

    [business_beam, elite_beam, friend_beam] >> create_user_table
```

Future improvements

This solution might be improved by:

- More comprehensive location data for our restaurants, looking at demographic information for neighborhoods
- More information for each Menu from NYPL
- More current menus
  - A lot of our data was from old menus, which wouldn't exist in the Yelp database since Yelp was founded in 2004