# **Data Scientist Role Play: Profiling and Analyzing the Yelp Dataset Coursera Worksheet**

Fadi Al Salti – submitted as a final project to Coursera on 24.07.2021

This is a 2-part assignment. In the first part, you are asked a series of questions that will help you profile and understand the data just like a data scientist would. For this first part of the assignment, you will be assessed both on the correctness of your findings, as well as the code you used to arrive at your answer. You will be graded on how easy your code is to read, so remember to use proper formatting and comments where necessary.

In the second part of the assignment, you are asked to come up with your own inferences and analysis of the data for a particular research question you want to answer. You will be required to prepare the dataset for the analysis you choose to do. As with the first part, you will be graded, in part, on how easy your code is to read, so use proper formatting and comments to illustrate and communicate your intent as required.

For both parts of this assignment, use this "worksheet." It provides all the questions you are being asked, and your job will be to transfer your answers and SQL coding where indicated into this worksheet so that your peers can review your work. You should be able to use any Text Editor (Windows Notepad, Apple TextEdit, Notepad ++, Sublime Text, etc.) to copy and paste your answers. If you are going to use Word or some other page layout application, just be careful to make sure your answers and code are lined appropriately.

In this case, you may want to save as a PDF to ensure your formatting remains intact for you reviewer.

**Part 1: Yelp Dataset Profiling and Understanding**

**1. Profile the data by finding the total number of records for each of the tables below:**

SELECT COUNT(\*)

FROM table

i. Attribute table = 10000

ii. Business table = 10000

iii. Category table = 10000

iv. Checkin table = 10000

v. elite\_years table = 10000

vi. friend table = 10000

vii. hours table = 10000

viii. photo table = 10000

ix. review table = 10000

x. tip table = 10000

xi. user table = 10000

**2. Find the total distinct records by either the foreign key or primary key for each table. If two foreign keys are listed in the table, please specify which foreign key.**

i. Business = 10000 (SELECT COUNT(DISTINCT id) FROM business)

ii. Hours = 1562 (SELECT COUNT(DISTINCT business\_id) FROM hours)

iii. Category = 2643 (SELECT COUNT(DISTINCT business\_id) FROM category)

iv. Attribute = 1115 (SELECT COUNT(DISTINCT business\_id) FROM attribute)

v. Review = 10000 (SELECT COUNT(DISTINCT id) FROM review)

vi. Checkin = 493 (SELECT COUNT(DISTINCT business\_id) FROM checkin)

vii. Photo = 10000 (SELECT COUNT(DISTINCT id) FROM photo)

viii. Tip = 537 using first foreign key (SELECT COUNT(DISTINCT user\_id) FROM tip)

ix. User = 10000 (SELECT COUNT(DISTINCT id) FROM user)

x. Friend = 11 (SELECT COUNT(DISTINCT user\_id) FROM friend)

xi. Elite\_years = 2780 (SELECT COUNT(DISTINCT user\_id) FROM elite\_years)

**3. Are there any columns with null values in the Users table? Indicate "yes," or "no."**

Answer: no

SQL code used to arrive at answer:

SELECT \*

FROM user

WHERE NULL IN (id, name, review\_count, yelping\_since, useful, funny, cool, fans,average\_stars, compliment\_hot, compliment\_more, compliment\_profile, compliment\_cute,compliment\_list, compliment\_note, compliment\_plain, compliment\_cool, compliment\_funny, compliment\_writer, compliment\_photos);

**4. For each table and column listed below, display the smallest (minimum), largest (maximum), and average (mean) value for the following fields:**

i. Table: Review, Column: Stars

(SELECT MIN(stars), MAX(stars), AVG(stars) FROM review)

min: max: avg:

1 5 3.7082

ii. Table: Business, Column: Stars

(SELECT MIN(stars), MAX(stars), AVG(stars) FROM business)

min: max: avg:

1.0 5.0 3.6549

iii. Table: Tip, Column: Likes

(SELECT MIN(likes), MAX(likes), AVG(likes) FROM tip)

min: max: avg:

0 2 0.0144

iv. Table: Checkin, Column: Count

(SELECT MIN(count), MAX(count), AVG(count) FROM checkin)

min: max: avg:

1 53 1.9414

v. Table: User, Column: Review\_count (SELECT MIN(review\_count), MAX(review\_count), AVG(review\_count) FROM user)

min: max: avg:

0 2000 24.2995

**5. List the cities with the most reviews in descending order:**

SQL code used to arrive at answer:

SELECT city, SUM(review\_count)

FROM business

GROUP BY city

ORDER BY SUM(review\_count) DESC;

Copy and Paste the Result Below:

+-----------------+-------------------+

| city | SUM(review\_count) |

+-----------------+-------------------+

| Las Vegas | 82854 |

| Phoenix | 34503 |

| Toronto | 24113 |

| Scottsdale | 20614 |

| Charlotte | 12523 |

| Henderson | 10871 |

| Tempe | 10504 |

| Pittsburgh | 9798 |

| Montréal | 9448 |

| Chandler | 8112 |

| Mesa | 6875 |

| Gilbert | 6380 |

| Cleveland | 5593 |

| Madison | 5265 |

| Glendale | 4406 |

| Mississauga | 3814 |

| Edinburgh | 2792 |

| Peoria | 2624 |

| North Las Vegas | 2438 |

| Markham | 2352 |

| Champaign | 2029 |

| Stuttgart | 1849 |

| Surprise | 1520 |

| Lakewood | 1465 |

| Goodyear | 1155 |

+-----------------+-------------------+

**6. Find the distribution of star ratings to the business in the following cities:**

i. Avon

SQL code used to arrive at answer:

SELECT stars AS star\_rating, COUNT(stars)

FROM business

WHERE city = 'Avon'

GROUP BY stars;

Copy and Paste the Resulting Table Below (2 columns – star rating and count):

+-------------+--------------+

| star\_rating | COUNT(stars) |

+-------------+--------------+

| 1.5 | 1 |

| 2.5 | 2 |

| 3.5 | 3 |

| 4.0 | 2 |

| 4.5 | 1 |

| 5.0 | 1 |

+-------------+--------------+

ii. Beachwood

SQL code used to arrive at answer:

SELECT stars AS star\_rating, COUNT(stars)

FROM business

WHERE city = 'Beachwood'

GROUP BY stars;

Copy and Paste the Resulting Table Below (2 columns – star rating and count):

+-------------+--------------+

| star\_rating | COUNT(stars) |

+-------------+--------------+

| 2.0 | 1 |

| 2.5 | 1 |

| 3.0 | 2 |

| 3.5 | 2 |

| 4.0 | 1 |

| 4.5 | 2 |

| 5.0 | 5 |

+-------------+--------------+

**7. Find the top 3 users based on their total number of reviews:**

SQL code used to arrive at answer:

SELECT name, review\_count

FROM user

ORDER BY review\_count DESC

LIMIT 3;

Copy and Paste the Result Below:

+--------+--------------+

| name | review\_count |

+--------+--------------+

| Gerald | 2000 |

| Sara | 1629 |

| Yuri | 1339 |

+--------+--------------+

**8. Does posing more reviews correlate with more fans? Please explain your findings and interpretation of the results:**

Intuitively my answer would be yes, but since there are no native correlation functions in SQL, this query is the best I could adapt from the mathematical formula below. The pearson coefficient ranges between -1 (negatively correlated) and 1 (positvely correlated).

Equation

SELECT AVG((review\_count - avg\_x) \* (fans - avg\_y))

\* AVG((review\_count - avg\_x) \* (fans - avg\_y)) / (var\_x\*var\_y) as Pearson\_correlation

FROM user, (SELECT avg\_x, avg\_y,

    AVG((review\_count - avg\_x)\*(review\_count - avg\_x)) as var\_x,

    AVG((fans - avg\_y)\*(fans - avg\_y)) as var\_y

    FROM user, (SELECT

          AVG(review\_count) as avg\_x,

          AVG(fans) as avg\_y

          FROM user));

+---------------------+

| Pearson\_correlation |

+---------------------+

| 0.437136492915 |

+---------------------+

With a correlation coeff of ~0.437, we can say the two variables have *moderate* correlation – i.e. a higher review\_count means higher fans and vice versa.

**9. Are there more reviews with the word "love" or with the word "hate" in them?**

Answer: LOVE WINS. 1780 reviews mentioned the word “love”, while 232 reviews mentioned the word “hate”.

*SQL code used to arrive at answer:*

SELECT COUNT(\*)

FROM review

WHERE text LIKE '%love%';

SELECT COUNT(\*)

FROM review

WHERE text LIKE '%hate%';

**10. Find the top 10 users with the most fans:**

SQL code used to arrive at answer:

SELECT name, fans

FROM user

ORDER BY fans DESC

LIMIT 10;

Copy and Paste the Result Below:

+-----------+------+

| name | fans |

+-----------+------+

| Amy | 503 |

| Mimi | 497 |

| Harald | 311 |

| Gerald | 253 |

| Christine | 173 |

| Lisa | 159 |

| Cat | 133 |

| William | 126 |

| Fran | 124 |

| Lissa | 120 |

+-----------+------+

**Part 2: Inferences and Analysis**

1. Pick one city and category of your choice and group the businesses in that city or category by their overall star rating. Compare the businesses with 2-3 stars to the businesses with 4-5 stars and answer the following questions. Include your code.

I pick the city “Charlotte” and the category “Nightlife”.

SELECT city, category, AVG(stars), hours

FROM business

-- Joining category and hours tables to business

INNER JOIN category

ON business.id = category.business\_id

INNER JOIN hours

ON business.id = hours.business\_id

-- Only for the Nightlife category

WHERE category = "Nightlife"

GROUP BY city

ORDER BY AVG(stars) DESC

+-------------+-----------+---------------+----------------------+

| city | category | AVG(stars) | hours |

+-------------+-----------+---------------+----------------------+

| Peninsula | Nightlife | 4.5 | Saturday|15:00-23:00 |

| Mesa | Nightlife | 4.0 | Saturday|11:00-22:00 |

| Toronto | Nightlife | 3.61538461538 | Saturday|16:00-2:00 |

| Chandler | Nightlife | 3.5 | Saturday|9:00-2:30 |

| Las Vegas | Nightlife | 3.5 | Saturday|0:00-0:00 |

| Phoenix | Nightlife | 3.5 | Saturday|9:00-2:00 |

| Hudson | Nightlife | 3.0 | Saturday|11:00-2:30 |

| Mississauga | Nightlife | 3.0 | Saturday|10:00-1:00 |

| Montréal | Nightlife | 3.0 | Saturday|11:30-0:00 |

| Edinburgh | Nightlife | 2.0 | Thursday|22:30-3:00 |

+-------------+-----------+---------------+----------------------+

**i. Do the two groups you chose to analyze have a different distribution of hours?**

There are six cities with an average stars score of 3.5 and more and four cities with 3.0 or less. There is no clear difference between the opening hours except the case of Edinburgh, where the nightclub open on Thursday and has the worst rating in the table.

**ii. Do the two groups you chose to analyze have a different number of reviews?**

+-------------+-----------+---------------+--------------+

| city | category | AVG(stars) | review\_count |

+-------------+-----------+---------------+--------------+

| Peninsula | Nightlife | 4.5 | 42 |

| Mesa | Nightlife | 4.0 | 129 |

| Toronto | Nightlife | 3.61538461538 | 26 |

| Chandler | Nightlife | 3.5 | 141 |

| Las Vegas | Nightlife | 3.5 | 105 |

| Phoenix | Nightlife | 3.5 | 60 |

| Hudson | Nightlife | 3.0 | 5 |

| Mississauga | Nightlife | 3.0 | 27 |

| Montréal | Nightlife | 3.0 | 19 |

| Edinburgh | Nightlife | 2.0 | 11 |

+-------------+-----------+---------------+--------------+

The review count tend to be higher for nightclubs with higher ratings.

**iii. Are you able to infer anything from the location data provided between these two groups? Explain.**

The cities are distributed between USA, Canada and the UK, some are small and some are big, but there are no clear interesting patterns to report.

+-------------+-------+-----------+---------------+--------------+

| city | state | category | AVG(stars) | review\_count |

+-------------+-------+-----------+---------------+--------------+

| Peninsula | OH | Nightlife | 4.5 | 42 |

| Mesa | AZ | Nightlife | 4.0 | 129 |

| Toronto | ON | Nightlife | 3.61538461538 | 26 |

| Chandler | AZ | Nightlife | 3.5 | 141 |

| Las Vegas | NV | Nightlife | 3.5 | 105 |

| Phoenix | AZ | Nightlife | 3.5 | 60 |

| Hudson | OH | Nightlife | 3.0 | 5 |

| Mississauga | ON | Nightlife | 3.0 | 27 |

| Montréal | QC | Nightlife | 3.0 | 19 |

| Edinburgh | EDH | Nightlife | 2.0 | 11 |

+-------------+-------+-----------+---------------+--------------+

2. Group business based on the ones that are open and the ones that are closed. What differences can you find between the ones that are still open and the ones that are closed? List at least two differences and the SQL code you used to arrive at your answer.

i. Difference 1:

The top 10 categories for open business and closes businesses are shown in the two tables below:

+---------------------------+-----------------+

| category | category\_closed |

+---------------------------+-----------------+

| Restaurants | 18 |

| Nightlife | 8 |

| Bars | 6 |

| Shopping | 5 |

| American (New) | 3 |

| American (Traditional) | 3 |

| Event Planning & Services | 3 |

| Food | 3 |

| Desserts | 2 |

| Gluten-Free | 2 |

+---------------------------+-----------------+

+------------------+---------------+

| category | category\_open |

+------------------+---------------+

| Restaurants | 53 |

| Shopping | 25 |

| Food | 20 |

| Health & Medical | 16 |

| Home Services | 15 |

| Beauty & Spas | 12 |

| Nightlife | 12 |

| Bars | 11 |

| Active Life | 10 |

| Local Services | 10 |

+------------------+---------------+

ii. Difference 2:

There are around four more open businesses than closed ones in the business table. The average rating is however surprisingly similar across both groups, with the open businesses having 0.15 more points on average.

+---------+---------------+----------------+

| is\_open | AVG(stars) | COUNT(is\_open) |

+---------+---------------+----------------+

| 0 | 3.52039473684 | 1520 |

| 1 | 3.67900943396 | 8480 |

+---------+---------------+----------------+

SQL code used for analysis:

SELECT category, COUNT(id) AS category\_closed

FROM business

INNER JOIN category

ON business.id = category.business\_id

WHERE is\_open = 0 -- is\_open = 1 to check the top categories of open businesses

GROUP BY category

ORDER BY COUNT(id) DESC

LIMIT 10;

SELECT is\_open, AVG(stars), COUNT(is\_open)

FROM business

GROUP BY is\_open;

3. For this last part of your analysis, you are going to choose the type of analysis you want to conduct on the Yelp dataset and are going to prepare the data for analysis.

Ideas for analysis include: Parsing out keywords and business attributes for sentiment analysis, clustering businesses to find commonalities or anomalies between them, predicting the overall star rating for a business, predicting the number of fans a user will have, and so on. These are just a few examples to get you started, so feel free to be creative and come up with your own problem you want to solve. Provide answers, in-line, to all of the following:

**i. Indicate the type of analysis you chose to do:**

I was curious what type of food was the most highly rated.

**ii. Write 1-2 brief paragraphs on the type of data you will need for your analysis and why you chose that data:**

For that analysis I need the *business* and *category* tables joined by an inner join. I filtered through a subquery to get only the categories that contain the word “Food”, then grouped by category and ordered in descending manner. Voila! It seems like seafood and seamarkets score in general the highest while fast food score the lowest. However, it must be noted that this is not a symmetric analysis as is clear by the total review count. Additionally, most businesses have more than one category so some of these ratings overlap.

**iii. Output of your finished dataset:**

+-----------------+----------------+--------------------+

| category | average\_rating | total\_review\_count |

+-----------------+----------------+--------------------+

| Seafood | 4.5 | 7 |

| Seafood Markets | 4.5 | 723 |

| Comfort Food | 4.0 | 30 |

| Ethnic Food | 4.0 | 726 |

| Specialty Food | 4.0 | 896 |

| Food | 3.78260869565 | 1781 |

| Food Trucks | 3.75 | 12 |

| Soul Food | 3.75 | 10 |

| Imported Food | 3.5 | 3 |

| Fast Food | 3.21428571429 | 185 |

+-----------------+----------------+--------------------+

**iv. Provide the SQL code you used to create your final dataset:**

SELECT category, AVG(stars) AS average\_rating, SUM(review\_count) AS total\_review\_count

FROM business

INNER JOIN category

ON business.id = category.business\_id

-- Using a subquery to filter only the categories that contain the word "Food"

WHERE category IN

(SELECT DISTINCT category

from category

WHERE category LIKE "%Food%")

GROUP BY category

ORDER BY average\_rating DESC