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

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:

Code used:

```
select count(*)
from Attribute

select count(*)
from Business

select count(*)
from Category

select count(*)
from Checkin

select count(*)
from elite_years

select count(*)
from friend

select count(*)
from hours

select count(*)
from photo

select count(*)
from review

select count(*)
from tip

select count(*)
from user
```

Result found:

Attribute table =	*10000*
Business table =	*10000*
Category table =	*10000*
Checkin table =	*10000*
elite_years table =	*10000*
friend table =	*10000*
hours table =	*10000*
photo table =	*10000*
review table =	*10000*
tip table =	*10000*
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.

Code used:

```
select count(distinct id)
from business

select count(distinct business_id)
from hours

select count(distinct business_id)
from category

select count(distinct business_id)
from attribute

select count(distinct id), count(distinct business_id), count(distinct user_id)
from review

select count(distinct business_id)
from checkin

select count(distinct id), count(distinct business_id)
from photo

select count(distinct user_id), count(distinct business_id)
from tip

select count(distinct id)
from user

select count(distinct user_id)
from friend

select count(distinct user_id)
from elite_years
```

Result found:

Business =	*10000*	*(id)*		
Hours =	*1562*	*(business_id)*		
Category =	*2643*	*(business_id)*		
Attribute =	*1115*	*(business_id)*		
Review =	*10000*	*(id- primary),*	*8090 (business_id- foreign),*	*9581 (user_id- foreign)*
Checkin =	*493*	*(business_id)*		
Photo =	*10000*	*(id- primary),*	*6493 (business_id- foreign)*	
Tip =	*537*	*(user_id- foreign),*	*3979 (business_id- foreign)*	
User =	*10000*	*(id)*		
Friend =	*11*	*(user_id)*		
Elite_years =	*2780*	*(user_id)*		

Note: Primary Keys are denoted in the ER-Diagram with a yellow key icon.

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 id is null
or name is null
or review_count is null
or yelping_since is null
or useful is null
or funny is null
or cool is null
or average_stars is null
or compliment_hot is null
or compliment_more is null
or compliment_profile is null
or compliment_cute is null
or compliment_list is null
or compliment_note is null
or compliment_plain is null
or compliment_cool is null
or compliment_funny is null
or compliment_writer is null
or compliment_photos is null
```

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

min: 1 *max: 5* *avg: 3.7082*

Code used:

```
select min(stars) as min, max(stars) as max, avg(stars) as avg
from review
```

Result found:

```
+-----+-----+--------+
| min | max |    avg |
+-----+-----+--------+
|   1 |   5 | 3.7082 |
+-----+-----+--------+
```

ii. Table: Business, Column: Stars

min: 1.0   max: 5.0   avg: 3.6549

```
select min(stars) as min, max(stars) as max, avg(stars) as avg
from business
```

```
+-----+-----+--------+
| min | max |    avg |
+-----+-----+--------+
| 1.0 | 5.0 | 3.6549 |
+-----+-----+--------+
```

iii. Table: Tip, Column: Likes

min: 0   max: 2   avg: 0.0144

```
select min(likes) as min, max(likes) as max, avg(likes) as avg
from tip
```

```
+-----+-----+--------+
| min | max |    avg |
+-----+-----+--------+
|   0 |   2 | 0.0144 |
+-----+-----+--------+
```

iv. Table: Checkin, Column: Count

min: 1   max: 53   avg: 1.9414

```
select min(count) as min, max(count) as max, avg(count) as avg
from checkin
```

```
+-----+-----+--------+
| min | max |    avg |
+-----+-----+--------+
|   1 |  53 | 1.9414 |
+-----+-----+--------+
```

v. Table: User, Column: Review_count

min: 0   max: 2000   avg: 24.2995

```
select min(review_count) as min, max(review_count) as max, avg(review_count) as avg
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) as total_reviews
from business
group by 1
order by 2 desc
```

Copy and Paste the Result Below:

```
+-----------------+---------------+
| city            | total_reviews |
+-----------------+---------------+
| 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 |
+-----------------+---------------+
(Output limit exceeded, 25 of 362 total rows shown)
```

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) as count
from business
where city = 'Avon'
group by 1
order by 1 desc
```

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

```
+-------------+-------+
| star_rating | count |
+-------------+-------+
|         5.0 |     1 |
|         4.5 |     1 |
|         4.0 |     2 |
|         3.5 |     3 |
|         2.5 |     2 |
|         1.5 |     1 |
+-------------+-------+
```

ii. Beachwood
SQL code used to arrive at answer:

```
select stars as star_rating, count(stars) as count
from business
where city = 'Beachwood'
group by 1
order by 1 desc
```

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

```
+-------------+-------+
| star_rating | count |
+-------------+-------+
|         5.0 |     5 |
|         4.5 |     2 |
|         4.0 |     1 |
|         3.5 |     2 |
|         3.0 |     2 |
|         2.5 |     1 |
|         2.0 |     1 |
+-------------+-------+
```

7. Find the top 3 users based on their total number of reviews:
SQL code used to arrive at answer:

```
select id, name, sum(review_count) as 'review count'
from user
group by 1
order by 3 desc
limit 3
```

Copy and Paste the Result Below:

```
+------------------------+--------+--------------+
| id                     | name   | review count |
+------------------------+--------+--------------+
| -G7Zkl1wIWBBmD0KRy_sCw | Gerald |         2000 |
| -3s52C4zL_DHRK0ULG6qtg | Sara   |         1629 |
| -8lbUNlXVSoXqaRRiHiSNg | Yuri   |         1339 |
+------------------------+--------+--------------+
```

8. Does posing more reviews correlate with more fans?

Please explain your findings and interpretation of the results: *Not really. I have written a query to find out the 5 top most review writers among the users. From the output, the user named Sara would have more fans than Yuri, .Hon and William as she wrote more reviews than they did but that didn't happen. So, apparently more reviews don't directly correlate with more fans.*

```
select id, name, sum(review_count), sum(fans)
from user
group by 1
order by 3 desc
limit 5
```

```
+------------------------+---------+--------------+------+
| id                     | name    | review_count | fans |
+------------------------+---------+--------------+------+
| -G7Zkl1wIWBBmD0KRy_sCw | Gerald  |         2000 |  253 |
| -3s52C4zL_DHRK0ULG6qtg | Sara    |         1629 |   50 |
| -8lbUNlXVSoXqaRRiHiSNg | Yuri    |         1339 |   76 |
| -K2Tcgh2EKX6e6HqqIrBIQ | .Hon    |         1246 |  101 |
| -FZBTkAZEXoP7CYvRV2ZwQ | William |         1215 |  126 |
+------------------------+---------+--------------+------+
```

9. Are there more reviews with the word "love" or with the word "hate" in them?
Answer: *There are more reviews with the word 'Love' than 'Hate'. Sounds good!*
SQL code used to arrive at answer:

```
select 'Love' Word, count (text) as 'Count'
from review
where text like '%love%'
union
select 'Hate' Word, count (text) as 'Count'
from review
where text like '%hate%'
```

```
+------+-------+
| Word | Count |
+------+-------+
| Hate |   232 |
| Love |  1780 |
+------+-------+
```

10. Find the top 10 users with the most fans:
SQL code used to arrive at answer:

```
select name, sum(fans) as total_fans
from user
group by 1
order by 2 desc
limit 10
```

Copy and Paste the Result Below:

```
+-----------+------------+
| name      | total_fans |
+-----------+------------+
| Amy       |        519 |
| Mimi      |        498 |
| Harald    |        311 |
| Gerald    |        256 |
| Lisa      |        207 |
| Nicole    |        200 |
| Christine |        187 |
| Mark      |        156 |
| Jen       |        148 |
| Linda     |        148 |
+-----------+------------+
```

11. Is there a strong relationship (or correlation) between having a high number of fans and being listed as "useful" or "funny?" Out of the top 10 users with the highest number of fans, what percent are also listed as "useful" or "funny"?

Key:
0% - 25% - Low relationship
26% - 75% - Medium relationship
76% - 100% - Strong relationship

SQL code used to arrive at answer:

Query 1: the following query finds out the total number of fans, total number of funny reviews and total number of useful reviews.

```
select sum(fans) as 'total fans', sum(funny) as 'total funny', sum(useful) as 'total
useful'
from user;
```

Result:

```
+------------+-------------+--------------+
| total fans | total funny | total useful |
+------------+-------------+--------------+
|      14896 |      247927 |       380563 |
+------------+-------------+--------------+
```

Query 2: The following query finds out the total number of fans, total number of funny reviews and total number of useful reviews for the top 10 users with the most fans. It also finds out the percentage in the following columns. For example, Amy has 3.48% fans of the total fans, 1.04% funny reviews/reactions of the total funny reviews and 0.88% useful reviews among the total reviews.

```
select name, sum(fans), sum(funny), sum(useful),
  round((sum(fans)/14896.)*100, 2)  as '%_fans',
  round((sum(funny)/247927.)*100, 2)  as '%_funny',
  round((sum(useful)/380563.)*100, 2)  as '%_useful'
from user
group by 1
order by 2 desc
limit 10
```

```
+-----------+-----------+------------+-------------+--------+---------+----------+
| name      | sum(fans) | sum(funny) | sum(useful) | %_fans | %_funny | %_useful |
+-----------+-----------+------------+-------------+--------+---------+----------+
| Amy       |       519 |       2578 |        3366 |   3.48 |    1.04 |     0.88 |
| Mimi      |       498 |        143 |         266 |   3.34 |    0.06 |     0.07 |
| Harald    |       311 |     122419 |      122921 |   2.09 |   49.38 |     32.3 |
| Gerald    |       256 |       2326 |       17530 |   1.72 |    0.94 |     4.61 |
| Lisa      |       207 |        144 |         533 |   1.39 |    0.06 |     0.14 |
| Nicole    |       200 |        187 |         944 |   1.34 |    0.08 |     0.25 |
| Christine |       187 |       6672 |        5305 |   1.26 |    2.69 |     1.39 |
| Mark      |       156 |        703 |        4250 |   1.05 |    0.28 |     1.12 |
| Jen       |       148 |       3426 |        4029 |   0.99 |    1.38 |     1.06 |
| Linda     |       148 |       3080 |        3946 |   0.99 |    1.24 |     1.04 |
+-----------+-----------+------------+-------------+--------+---------+----------+
```

Query 3: the following query finds out how much the top 10 users' fans constitute of the total fans. It also shows the percentage of funny and useful reviews. The last two columns show the relationship between fans and funny in percentage (27.35%) and the relationship between fans and useful in percentage (41.98%).

```
select
  round( sum(fans/14896.*100), 2) as 'top 10 fan total (%)',
  round(sum(funny/247927.*100), 2) as 'top 10 funny total (%)',
  round(sum(useful/380563.*100), 2) as 'top 10 useful total (%)',
  round((sum(fans/14896.*100))*100/(sum(funny/247927.*100)), 2) as '%_fantofunny',
  round((sum(fans/14896.*100))*100/(sum(funny/380563.*100)), 2) as '%_fantouseful'
```

```
from user
where name in ('Amy', 'Mimi', 'Harald', 'Gerald', 'Christine', 'Lisa', 'Cat', 'William',
'Fran', 'Lissa')
order by 2 desc;
```

```
+----------------------+-----------------------+--------------------------+--------------+----------------+
| top 10 fan total (%) | top 10 funny total (%) | top 10 useful total (%) | %_fantofunny | %_fantouseful |
+----------------------+-----------------------+--------------------------+--------------+----------------+
|                16.78 |                  61.34 |                    44.86 |        27.35 |          41.98 |
+----------------------+-----------------------+--------------------------+--------------+----------------+
```

Conclusion: Relationship between the high number of fans and being listed "funny" = 27.35% Relationship between the high number of fans and being listed "useful" = 41.98%. So, there's a Medium relationship (26% - 75%) between these factors.

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.

   City = Las Vegas
   Category = Restaurants

```
+---------------------+-------------+--------------+-----------------------+----------+-------------+----------+-----------+
| name                | star_rating | review_count | hours                 | days     | postal_code | latitude | longitude |
+---------------------+-------------+--------------+-----------------------+----------+-------------+----------+-----------+
| Big Wong Restaurant | 4-5 stars   |          768 | Friday|10:00-23:00    | weekdays | 89146       |  36.1267 |   -115.21 |
| Big Wong Restaurant | 4-5 stars   |          768 | Monday|10:00-23:00    | weekdays | 89146       |  36.1267 |   -115.21 |
| Big Wong Restaurant | 4-5 stars   |          768 | Saturday|10:00-23:00  | weekends | 89146       |  36.1267 |   -115.21 |
| Big Wong Restaurant | 4-5 stars   |          768 | Sunday|10:00-23:00    | weekends | 89146       |  36.1267 |   -115.21 |
| Big Wong Restaurant | 4-5 stars   |          768 | Thursday|10:00-23:00  | weekdays | 89146       |  36.1267 |   -115.21 |
| Big Wong Restaurant | 4-5 stars   |          768 | Tuesday|10:00-23:00   | weekdays | 89146       |  36.1267 |   -115.21 |
| Big Wong Restaurant | 4-5 stars   |          768 | Wednesday|10:00-23:00 | weekdays | 89146       |  36.1267 |   -115.21 |
| Jacques Cafe        | 4-5 stars   |          168 | Friday|11:00-20:00    | weekdays | 89134       |  36.1933 |  -115.304 |
| Jacques Cafe        | 4-5 stars   |          168 | Monday|11:00-20:00    | weekdays | 89134       |  36.1933 |  -115.304 |
| Jacques Cafe        | 4-5 stars   |          168 | Saturday|11:00-20:00  | weekends | 89134       |  36.1933 |  -115.304 |
| Jacques Cafe        | 4-5 stars   |          168 | Sunday|8:00-14:00     | weekends | 89134       |  36.1933 |  -115.304 |
| Jacques Cafe        | 4-5 stars   |          168 | Thursday|11:00-20:00  | weekdays | 89134       |  36.1933 |  -115.304 |
| Jacques Cafe        | 4-5 stars   |          168 | Tuesday|11:00-20:00   | weekdays | 89134       |  36.1933 |  -115.304 |
| Jacques Cafe        | 4-5 stars   |          168 | Wednesday|11:00-20:00 | weekdays | 89134       |  36.1933 |  -115.304 |
| Wingstop            | 2-3 stars   |          123 | Friday|11:00-0:00     | weekdays | 89103       |  36.1003 |   -115.21 |
| Wingstop            | 2-3 stars   |          123 | Monday|11:00-0:00     | weekdays | 89103       |  36.1003 |   -115.21 |
| Wingstop            | 2-3 stars   |          123 | Saturday|11:00-0:00   | weekends | 89103       |  36.1003 |   -115.21 |
| Wingstop            | 2-3 stars   |          123 | Sunday|11:00-0:00     | weekends | 89103       |  36.1003 |   -115.21 |
| Wingstop            | 2-3 stars   |          123 | Thursday|11:00-0:00   | weekdays | 89103       |  36.1003 |   -115.21 |
| Wingstop            | 2-3 stars   |          123 | Tuesday|11:00-0:00    | weekdays | 89103       |  36.1003 |   -115.21 |
| Wingstop            | 2-3 stars   |          123 | Wednesday|11:00-0:00  | weekdays | 89103       |  36.1003 |   -115.21 |
+---------------------+-------------+--------------+-----------------------+----------+-------------+----------+-----------+
```

i. Do the two groups you chose to analyze have a different distribution of hours? *There's no significant difference in the hours between these two groups. But Wingstop (2-3*) stays open until midnight whereas the (4-5*) restaurants close by 11 pm.*

ii. Do the two groups you chose to analyze have a different number of reviews? *Yes, they do. The (4-5*) restaurants have more than 160 reviews whereas the (2-3*) restaurant has 123 reviews.*

iii. Are you able to infer anything from the location data provided between these two groups? *From the latitude and longitude provided, Big Wong Restaurant is located near the strip in Las Vegas which is a reason it's a big hit. The other one, Jacques Café is located near some businesses, library and a high school. It's also surrounded by residential areas.*

SQL code used for analysis:

```
select b.name,
case
when b.stars between 2 and 3 then '2-3 stars'
when b.stars between 4 and 5 then '4-5 stars'
end as star_rating,
b.review_count,
h.hours,
case
when (hours like "%monday%"
or hours like "%tuesday%"
or hours like "%wednesday%"
or hours like "%thursday%"
or hours like "%friday%") then 'weekdays'
when (hours like "%saturday%"
or hours like "%sunday%") then 'weekends'
end as days,
postal_code,
latitude,
longitude
from business b
join hours h
on b.id = h.business_id
join category c
on c.business_id = b.id
where (b.city like 'las vegas'
and c.category like 'restaurants')
and (b.stars between 2 and 3
or b.stars between 4 and 5)
group by hours
order by review_count desc
```

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.

```
+----------+---------+-------------+-----------+---------------+-------------+---------------+
| business | is_open | total_stars | avg_stars | total_reviews | avg_reviews | sum(r.useful) |
+----------+---------+-------------+-----------+---------------+-------------+---------------+
|       61 |       0 |          71 |      3.65 |          9217 |       130.0 |            69 |
|      446 |       1 |         565 |      3.76 |        175821 |       311.0 |           484 |
+----------+---------+-------------+-----------+---------------+-------------+---------------+
```

i. Difference 1: *Closed ones have fewer stars and less average stars than the open ones.*
ii. Difference 2: *Closed ones have fewer total reviews than the open ones.*

SQL code used for analysis:

```
select count(distinct b.id) as business,
        b.is_open,
        count(r.stars) as total_stars,
        round(avg(r.stars), 2) avg_stars,
        sum(b.review_count) as total_reviews,
        round(avg(b.review_count), 0) as avg_reviews,
        sum(r.useful)
from business b
join review r
on b.id = r.business_id
group by 2
```

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 am going to answer the following question where Category is in (Asian Fusion, Chinese, Mexican, French, Italian, Indonesian, Korean, Japanese, Indian, Buffet, Coffee & Tea, Barbeque, Chicken Wings, Diners, Fast Food)

Part 1: What are the names, average ratings, average number of reviews and geographic location of the restaurants in these categories?
Part 2: Which of these restaurants are open and which are close?
Part 3: Did the customers feel any of the following emotions during their visit to these restaurants and wrote reviews about them?

Awesome!
Happy!
Delicious!
Great!
Yummy!
Best!

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

We will need to (inner) join the Business table and Category table for Part 1 and Part 2. For Part 3, we will have to (inner) join Business, Review and Category tables.

iii. Output of your finished dataset:

Part 1:

```
+---------------+----------------------------------------+------------+-------------+----------------+
| category      | name                                   | avg_rating | avg_reviews | city           |
+---------------+----------------------------------------+------------+-------------+----------------+
| Asian Fusion  | Big Wong Restaurant                    |        3.5 |       396.5 | Las Vegas      |
| Barbeque      | Bootleggers Modern American Smokehouse |       3.75 |       252.5 | Phoenix        |
| Chicken Wings | The Erin Mills Pump & Patio            |        3.0 |        75.0 | Mississauga    |
| Chinese       | Club India restaurant                  |       3.13 |       199.0 | Edinburgh      |
| Coffee & Tea  | Cabin Fever                            |       3.83 |        80.0 | Toronto        |
| Diners        | Market Street Diner                    |       3.75 |        58.5 | Sun Prairie    |
| Fast Food     | Del Taco                               |       3.21 |       26.43 | Gilbert        |
| French        | Jacques Cafe                           |        4.0 |       128.5 | Las Vegas      |
| Indian        | Club India restaurant                  |        3.6 |        12.6 | Edinburgh      |
| Italian       | Restaurant Rosalie                     |        3.5 |        74.0 | Montréal       |
| Japanese      | Hibachi-San                            |        3.8 |        30.4 | Las Vegas      |
| Korean        | Seoul Garden Korean Restaurant         |       4.25 |        31.5 | Cuyahoga Falls |
| Mexican       | Taqueria Y Cenaduria Culiacan          |        3.5 |       46.71 | Tolleson       |
+---------------+----------------------------------------+------------+-------------+----------------+
```

Part 2:

```
+--------------+----------------------------------------+-------+--------------+----------------+--------+
| category     | name                                   | stars | review_count | city           | status |
+--------------+----------------------------------------+-------+--------------+----------------+--------+
| Coffee & Tea | Cabin Fever                            |   4.5 |           26 | Toronto        | open   |
| Japanese     | Hibachi-San                            |   4.5 |            3 | Las Vegas      | closed |
| Korean       | Sushi Osaka                            |   4.5 |            8 | Toronto        | open   |
| Asian Fusion | Big Wong Restaurant                    |   4.0 |          768 | Las Vegas      | open   |
| Barbeque     | Bootleggers Modern American Smokehouse |   4.0 |          431 | Phoenix        | open   |
| French       | Edulis                                 |   4.0 |           89 | Toronto        | open   |
| Italian      | Eklectic Pie - Mesa                    |   4.0 |          129 | Mesa           | closed |
| Mexican      | Hermanos Mexican Grill                 |   4.0 |           69 | Mississauga    | open   |
| French       | Jacques Cafe                           |   4.0 |          168 | Las Vegas      | closed |
| Coffee & Tea | Koko Bakery                            |   4.0 |          162 | Cleveland      | open   |
| Mexican      | Mama Mia                               |   4.0 |            8 | Toronto        | closed |
| Japanese     | Masamune Japanese Restaurant           |   4.0 |           61 | Mississauga    | open   |
| Mexican      | Miros Cantina Mexicana                 |   4.0 |           37 | Edinburgh      | open   |
| Japanese     | Naniwa-Taro                            |   4.0 |           75 | Toronto        | open   |
| Indian       | Patiala House                          |   4.0 |           10 | Brampton       | open   |
| Diners       | Rise and Dine Cafe                     |   4.0 |           30 | Chesterland    | open   |
| Korean       | Seoul Garden Korean Restaurant         |   4.0 |           55 | Cuyahoga Falls | open   |
| Mexican      | Taqueria Y Cenaduria Culiacan          |   4.0 |           23 | Tolleson       | open   |
| Indian       | Cafe Tandoor                           |   3.5 |           32 | Aurora         | open   |
| Indian       | Club India restaurant                  |   3.5 |            3 | Edinburgh      | closed |
| Fast Food    | Five Guys                              |   3.5 |           63 | Phoenix        | open   |
| Indian       | Indian Ocean Restaurant                |   3.5 |            3 | Inverness      | open   |
| Diners       | Market Street Diner                    |   3.5 |           87 | Sun Prairie    | open   |
| Chinese      | Ping's Cafe                            |   3.5 |           21 | Fountain Hills | open   |
| Fast Food    | Poutine Lafleur                        |   3.5 |           11 | Verdun         | open   |
+--------------+----------------------------------------+-------+--------------+----------------+--------+
```

Part 3:

```
+--------------+----------------------------------------+-------+--------------+-----------+------------------+--------+
| category     | name                                   | stars | review_count | city      | customer_emotion | status |
+--------------+----------------------------------------+-------+--------------+-----------+------------------+--------+
| Asian Fusion | Big Wong Restaurant                    |   4.0 |          768 | Las Vegas | delicious!       | open   |
| Barbeque     | Bootleggers Modern American Smokehouse |   4.0 |          431 | Phoenix   | best!            | open   |
| Indian       | Cafe Tandoor                           |   3.5 |           32 | Aurora    | great!           | open   |
+--------------+----------------------------------------+-------+--------------+-----------+------------------+--------+
```

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

Part 1:

```
select c.category,
        b.name,
        round(avg(b.stars), 2) as avg_rating,
        round(avg(b.review_count), 2) as avg_reviews,
        b.city
from business b
join category c
on c.business_id = b.id
where c.category in ("Asian Fusion", "Chinese", "Mexican", "French", "Italian",
"Indonesian", "Korean", "Japanese", "Indian", 'Buffet', 'Coffee & Tea', 'Barbeque',
'Chicken Wings', 'Diners', 'Fast Food')
group by 1
```

Part 2:

```
select c.category,
        b.name,
        b.stars,
        b.review_count,
        b.city,
case
when b.is_open = 1 then 'open'
when b.is_open = 0 then 'closed'
end as 'status'
from business b
join category c
on c.business_id = b.id
where c.category in ("Asian Fusion", "Chinese", "Mexican", "French", "Italian",
"Indonesian", "Korean", "Japanese", "Indian", 'Buffet', 'Coffee & Tea', 'Barbeque',
'Chicken Wings', 'Diners', 'Fast Food')
group by 2
order by 3 desc
```

Part 3:

```
select c.category,
        b.name,
        b.stars,
        b.review_count,
        b.city,
case
when r.text like '%awesome%' then 'awesome!'
when r.text like '%pleased%'
    or r.text like '%happy%' then 'happy!'
when r.text like '%delicious%' then 'delicious!'
when r.text like '%great%' then 'great!'
when r.text like '%tasty%' then 'yummy!'
when r.text like '%best%' then 'best!'
end as 'customer_emotion',
case
when b.is_open = 1 then 'open'
when b.is_open = 0 then 'closed'
end as 'status'
from business b
join hours h
on b.id = h.business_id
join category c
on c.business_id = b.id
join review r
on b.id  = r.business_id
where c.category in ("Asian Fusion", "Chinese", "Mexican", "French", "Italian",
"Indonesian", "Korean", "Japanese", "Indian", 'Buffet', 'Coffee & Tea', 'Barbeque',
'Chicken Wings', 'Diners', 'Fast Food')
group by 2
order by 3 desc
```