Nate Boyle
4/19/2023

SQL for Data Science: Peer Review Assignment

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

*Please note that any text I add in other than SQL code is in Verdana
The SQL code or output text as well as the text from the original txt.file is in **Consolas**

Instructions:

```
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 your 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:

i. Attribute table = 10000 total rows (Select * From attribute)
ii. Business table = 10000 total rows (Select * From business)
iii. Category table = 10000 total rows (Select * From category)
iv. Checkin table = 10000 total rows (Select * From checkin)
v. elite_years table = 10000 total rows (Select * From elite_years)
vi. friend table = 10000 total rows (Select * From friend)
vii. hours table = 10000 total rows (Select * From hours)
viii. photo table = 10000 total rows (Select * From photo)
ix. review table = 10000 total rows (Select * From review)
x. tip table = 10000 total rows (Select * From tip)
xi. user table = 10000 total rows (Select * From user)

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 total rows
--*primary key*
(Select Distinct id From business)

ii. Hours = 1562 total rows
--*foreign key*
(Select Distinct business_id From hours)

iii. Category = 2643 total rows
--*foreign key*
(Select Distinct business_id From category)

iv. Attribute = 1115 total rows
--*foreign key*
(Select Distinct business_id From attribute)

v. Review = 10000 total rows
--*primary key*
(Select Distinct id From)

= 8090 total rows
--*foreign key*
(Select Distinct business_id From review)

```
          = 9581 total rows
--*foreign key* (Select Distinct user_id From review)

vi. Checkin = 493 total rows
--*foreign key*
(Select Distinct business_id From checkin)

vii. Photo = 10000 total rows
--*primary key*
(Select Distinct id From photo)

           = 6493 total rows
--*foreign key*
(Select Distinct business_id From photo)

viii. Tip = 3979 total rows
--*foreign key*
(Select Distinct business_id From tip)

          = 537 total rows
--*foreign key*
(Select Distinct user_id From tip)

ix. User = 10000 total rows
--*primary key*
(Select Distinct id From user)

x. Friend = 11 total rows
--*foreign key*
(Select Distinct user_id From friend)

xi. Elite_years = 2780 total rows
--*foreign key*
(Select Distinct user_id From elite_years)
```

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 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
  fans 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
```

FYI: I used the query:

```
Select * From user
      Where 1=0
```

first, because this just gives all of the columns. I then copy and pasted (transposed) the columns into Excel and did some string trickery to get all those "Is Null Or"s attached, and then copy and pasted from Excel back into the SQL screen.

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
```

ii. Table: Business, Column: Stars

```
min:  1.0   max:  5.0   avg: 3.6549
```

iii. Table: Tip, Column: Likes

```
min:  0     max:  2     avg: 0.0144
```

iv. Table: Checkin, Column: Count

```
min:  1     max:  53    avg: 1.9414
```

v. Table: User, Column: Review_count

```
min:  0     max:  2000  avg: 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_review_count From business
    Group By city
    Order By total_review_count Desc
```

Copy and Paste the Result Below:

```
+-----------------+--------------------+
| city            | total_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 |
+-----------------+--------------------+
(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 star_rating_count From business
Where city = "Avon"
Group By star_rating
Order By star_rating_count Desc
```

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

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

ii. Beachwood

SQL code used to arrive at answer:

```
Select stars As star_rating, Count(stars) As star_rating_count From business
Where city = "Beachwood"
Group By star_rating
Order By star_rating_count Desc
```

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

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

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

SQL code used to arrive at answer:

```sql
--Have to include id because some users have the same name
Select Distinct name As users_name, id As user_id, review_count From user
Order By review_count Desc Limit 3
```

Copy and Paste the Result Below:

```
+------------+------------------------+--------------+
| users_name | user_id                | review_count |
+------------+------------------------+--------------+
| Gerald     | -G7Zkl1wIWBBmD0KRy_sCw |         2000 |
| Sara       | -3s52C4zL_DHRK0ULG6qtg |         1629 |
| Yuri       | -8lbUNlXVSoXqaRRiHiSNg |         1339 |
+------------+------------------------+--------------+
```

8. Does posting more reviews correlate with more fans?

Please explain your findings and interpretation of the results:

Yes, but not a super strong correlation. I ran the query I used for question 7. but without the `Limit 3` and added in `fans` into the Select clause. I additionally ran this query three more times with slight edits, one that was ascending to contrast from the descending order, and then did the same but with the ordering by `fans` instead of `review_count` for both ascending and descending. When descending by `review_count` the `fans` field was all either double or triple digits, well above the `fans` average of 1.4896, and when descending by `fans` the `review_count` field was all in the triple or quadruple digits, well above the 24.2995 average. Similar correlated results were found when ordering either field in the ascending order, as `review_count` was mostly in the single digits with a few double digits when ordered by `fans`, and `fans` was all 0s when ordered by `review_count`.

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

Answer: Love, as the 'text' field contained 1780 instances with the word "love", whereas there were only 232 instances with the word "hate".

SQL code used to arrive at answer:

```sql
    --For love query
    Select Count(text) As num_of_reviews_with_word From review
    Where text Like "%love%"

    --For hate query
    Select Count(text) As num_of_reviews_with_word From review
    Where text Like "%hate%"
```

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

SQL code used to arrive at answer:

```
/*This is the same query from question 7., just with review_count swapped out for
fans and a Limit of 10 instead of 3 For love query*/
    Select Distinct name As users_name, id As user_id, fans From user
    Order By fans Desc Limit 10
```

Copy and Paste the Result Below:

```
+------------+------------------------+------+
| users_name | user_id                | fans |
+------------+------------------------+------+
| Amy        | -9I98YbNQnLdAmcYfb324Q |  503 |
| Mimi       | -8EnCioUmDygAbsYZmTeRQ |  497 |
| Harald     | --2vR0DIsmQ6WfcSzKWigw |  311 |
| Gerald     | -G7Zkl1wIWBBmD0KRy_sCw |  253 |
| Christine  | -0IiMAZI2SsQ7VmyzJjokQ |  173 |
| Lisa       | -g3XIcCb2b-BD0QBCcq2Sw |  159 |
| Cat        | -9bbDysuiWeo2VShFJJtcw |  133 |
| William    | -FZBTkAZEXoP7CYvRV2ZwQ |  126 |
| Fran       | -9da1xk7zgnnfO1uTVYGkA |  124 |
| Lissa      | -lh59ko3dxChBSZ9U7LfUw |  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.

City choice: Las Vegas
Category choice: Restaurants

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

The distribution of hours is fairly consistent between the groups, with the 2-3 stars group being open from 11AM-12AM (midnight), and the 4-5 stars group being open 11AM-10PM and 10AM-11PM, with one business being an outlier having the hours 8AM-2PM on Sunday.

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

The distribution of reviews is more varied between the two groups than the distribution of hours. The one business in the 2-3 stars group has 123 reviews, whereas the average of the three businesses in the 4-5 stars group is 313, with one business having 168 reviews, the second having 768, and the third only having 3. A Google search was run and verified that the third business, Hibachi-San, is permanently closed, and the last review was in 2013, perhaps this is why there were only 3 reviews and no results for the hours table.

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

No discernable correlation related to location. One might think that proximity to the Las Vegas strip might correlate with more reviews, this was found to not be the case, perhaps with a bigger data set a correlation may be found.

SQL code used for analysis:

SQL code used to answer question i.

```
--To search for city:
Select Distinct city From business

--To search for category:
Select Distinct category From category

--To get list of businesses in city and category along with their stars
Select city, category, name , stars
From business b Join category c On b.id = c.business_id
Where city = "Las Vegas" And category = "Restaurants"
```

Output:

```
+-----------+-------------+---------------------+-------+
| city      | category    | name                | stars |
+-----------+-------------+---------------------+-------+
| Las Vegas | Restaurants | Jacques Cafe        |   4.0 |
| Las Vegas | Restaurants | Wingstop            |   3.0 |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 |
| Las Vegas | Restaurants | Hibachi-San         |   4.5 |
+-----------+-------------+---------------------+-------+
```

```
--To get hours for businesses between 2-3 stars
Select city, category, name, stars, hours
From business b Join category c On b.id = c.business_id
Join hours h On b.id = h.business_id
Where city = "Las Vegas" And category = "Restaurants"
And stars Between 2 And 3
```

Output:

```
+-----------+-------------+----------+-------+----------------------+
| city      | category    | name     | stars | hours                |
+-----------+-------------+----------+-------+----------------------+
| Las Vegas | Restaurants | Wingstop |   3.0 | Monday|11:00-0:00    |
| Las Vegas | Restaurants | Wingstop |   3.0 | Tuesday|11:00-0:00   |
| Las Vegas | Restaurants | Wingstop |   3.0 | Friday|11:00-0:00    |
| Las Vegas | Restaurants | Wingstop |   3.0 | Wednesday|11:00-0:00 |
| Las Vegas | Restaurants | Wingstop |   3.0 | Thursday|11:00-0:00  |
| Las Vegas | Restaurants | Wingstop |   3.0 | Sunday|11:00-0:00    |
| Las Vegas | Restaurants | Wingstop |   3.0 | Saturday|11:00-0:00  |
+-----------+-------------+----------+-------+----------------------+
--To get hours for businesses between 4-5 stars
Select city, category, name, stars, hours
From business b Join category c On b.id = c.business_id
Join hours h On b.id = h.business_id
Where city = "Las Vegas" And category = "Restaurants"
And stars Between 4 And 5
```

Output:

```
+-----------+-------------+--------------------+-------+----------------------+
| city      | category    | name               | stars | hours                |
+-----------+-------------+--------------------+-------+----------------------+
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Monday|11:00-20:00    |
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Tuesday|11:00-20:00   |
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Friday|11:00-20:00    |
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Wednesday|11:00-20:00 |
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Thursday|11:00-20:00  |
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Sunday|8:00-14:00     |
| Las Vegas | Restaurants | Jacques Cafe       |   4.0 | Saturday|11:00-20:00  |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Monday|10:00-23:00    |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Tuesday|10:00-23:00   |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Friday|10:00-23:00    |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Wednesday|10:00-23:00 |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Thursday|10:00-23:00  |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Sunday|10:00-23:00    |
| Las Vegas | Restaurants | Big Wong Restaurant |   4.0 | Saturday|10:00-23:00  |
+-----------+-------------+--------------------+-------+----------------------+
```

***Note:** Hibachi-San is absent from the table above, using the id associated with Hibachi-San from the business table an additional query was ran for the business_id in the hours table and no results were found, verifying there was no record for Hibachi-San in the hours table.

SQL code used to answer question ii.

```
--To get review_count for businesses between 2-3 stars
Select city, category, name, stars, review_count
From business b Join category c On b.id = c.business_id
Where city = "Las Vegas" And category = "Restaurants"
And stars Between 2 And 3
```

Output:

```
+-----------+-------------+----------+-------+--------------+
| city      | category    | name     | stars | review_count |
+-----------+-------------+----------+-------+--------------+
| Las Vegas | Restaurants | Wingstop |   3.0 |          123 |
+-----------+-------------+----------+-------+--------------+
--To get review_count for businesses between 4-5 stars
Select city, category, name, stars, review_count
From business b Join category c On b.id = c.business_id
Where city = "Las Vegas" And category = "Restaurants"
And stars Between 4 And 5

+-----------+-------------+----------------------+-------+--------------+
| city      | category    | name                 | stars | review_count |
+-----------+-------------+----------------------+-------+--------------+
| Las Vegas | Restaurants | Jacques Cafe         |   4.0 |          168 |
| Las Vegas | Restaurants | Big Wong Restaurant  |   4.0 |          768 |
| Las Vegas | Restaurants | Hibachi-San          |   4.5 |            3 |
+-----------+-------------+----------------------+-------+--------------+
```

SQL code used to answer question iii.

```
--To get location data for businesses between 2-3 stars
Select b.name, neighborhood, address, postal_code, latitude, longitude
From business b Join category c On b.id = c.business_id
Where city = "Las Vegas" And category = "Restaurants"
And b.stars Between 2 And 3
```

Output:

```
+----------+----------------------+-------------+----------+-----------+
| name     | address              | postal_code | latitude | longitude |
+----------+----------------------+-------------+----------+-----------+
| Wingstop | 5045 W Tropicana Ave | 89103       |  36.1003 |   -115.21 |
+----------+----------------------+-------------+----------+-----------+
```

```
--To get location data for businesses between 4-5 stars
Select b.name, address, postal_code, latitude, longitude
From business b Join category c On b.id = c.business_id
Where city = "Las Vegas" And category = "Restaurants"
And b.stars Between 4 And 5

+------------+-----------------------+-------------+----------+----------+
| name       | address               | postal_code | latitude |longitude |
+------------+-----------------------+-------------+----------+----------+
|Jacques Cafe|1910 Village Center Cir| 89134       |  36.1933 | 115.304  |
|Big Wong    |5040 Spring Mountain Rd| 89146       |  36.1267 | -115.21  |
|Hibachi-San |3480 S Maryland Pkwy   | 89169       |  36.1259 | -115.135 |
+------------+-----------------------+-------------+----------+----------+
```

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: Review count

I reasoned that the more reviews a business had the more likely it was open because most businesses that close do so early after their opening, and `review_count` is a metric that would grow and accumulate over time. Sure enough, closed businesses tended to have a smaller `review_count` while businesses that were open tended to have a higher `review_count`. I ran queries with a condition of greater than a 100, 500, and 1000 `review_count`, and the higher the `review_count` the greater the percent open, with 90.1%, 96.1%, and 100% respectively, which were all above the baseline percent of stores open when no additional conditions were added to a query which was 84.8%, clearly demonstrating a difference between businesses that were open and businesses that were closed when the `review_count` was taken into account.

ii. Difference 2: Stars

I also reasoned those businesses with a higher rating, i.e., more `stars`, would be more likely to stay open, because a higher rating typically meant better customer satisfaction which also tends to mean better customer retention which is one of the main factors for a business to stay open. Initially I ran queries with just one additional condition for `stars`, dividing the `stars` up between three different groups of three (1.0, 1.5, 2.0), (2.5, 3.0, 3.5), and (4.0, 4.5, 5.0), below this paragraph is the output of a `Select Distinct stars` query `From` the `business` table. The initial results were a bit perplexing, as while the 4.0-5.0 group did have the highest percent of open businesses, the 1.0-2.0 group had the second highest. I realized that there was no measure of how long the business had been open, or more importantly, when the business began. Because surely the longer the further back in time a business had been open the more a lower rating correlated with its likelihood to close. To account for this, I also added in the greater than `review_count` condition from the first difference but kept it at greater than 100 regardless of stars to have it be constant. Once both conditions were added to the query a clear correlation was present with the 10-2.0 group having 85.7% of businesses open, the 2.5-3.5 group having 87.4% of businesses open, and the 4.0-5.0 group having 92.1% of businesses open. It should also be noted that the first two groups were below, and the last group was above, the 90.1% baseline for percent of businesses open with a `review_count` of over 100, clearly demonstrating a difference between businesses that were open and closed when the number of `stars` was taken into account.

```
+--------+
| stars  |
+--------+
|    5.0 |
|    4.5 |
|    4.0 |
|    3.5 |
|    3.0 |
|    2.5 |
|    2.0 |
|    1.5 |
|    1.0 |
+--------+
```

SQL code used for analysis:

SQL code used to get a baseline of how many businesses were open or closed

```
--To get count and percent of businesses that are open or closed
Select Count(Distinct b.id) As total_biz,
    --Get count of businesses that are closed
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0) As biz_closed,
    /*Divide count of businesses that are closed by total businesses
    multiply by 100 and concatenate '%'*/
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
    --Get count of businesses that are open
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1) As biz_open,
    /*Divide count of businesses that are open by total businesses
    multiply by 100 and concatenate '%'*/
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
```

Output:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|     10000 |       1520 | 15.2%          |     8480 | 84.8%        |
+-----------+------------+----------------+----------+--------------+
```

SQL code used to for businesses with a review_count over 100

```
/*To get count and percent of businesses that are open or closed with a
review_count over 100*/
Select Count(Distinct b.id) As total_biz,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And review_count > 100) As biz_closed,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And review_count > 100) As biz_open,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
Where review_count > 100
```

Output:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|       615 |         61 |          9.92% |      554 |       90.08% |
+-----------+------------+----------------+----------+--------------+
```

SQL code used to for businesses with a review_count over 500

```
/*To get count and percent of businesses that are open or closed with a
review_count over 500*/
Select Count(Distinct b.id) As total_biz,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And review_count > 500) As biz_closed,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And review_count > 500) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And review_count > 500) As biz_open,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And review_count > 500) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
Where review_count > 500
```

Output:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|        51 |          2 |          3.92% |       49 |       96.08% |
+-----------+------------+----------------+----------+--------------+
```

SQL code used to for businesses with a review_count over 1000

```
/*To get count and percent of businesses that are open or closed with a
review_count over 1000*/
Select Count(Distinct b.id) As total_biz,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And review_count > 1000) As biz_closed,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And review_count > 1000) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And review_count > 1000) As biz_open,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And review_count > 1000) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
Where review_count > 1000
```

Output:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|         8 |          0 |           0.0% |        8 |       100.0% |
+-----------+------------+----------------+----------+--------------+
```

SQL code used to for businesses with 2 or less stars

```
/*To get count and percent of businesses that are open or closed when less than
or equal to 2 stars and review_count > 100*/
Select Count(Distinct b.id) As total_biz,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And stars <= 2 And review_count > 100)
            As biz_closed,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And stars <= 2 And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And stars <= 2 And review_count > 100) As biz_open,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And stars <= 2 And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
Where stars <= 2 And review_count > 100
```

Output with review_count condition added:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|         7 |          1 | 14.29%         |        6 | 85.71%       |
+-----------+------------+----------------+----------+--------------+
```

Output without review_count condition added:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|       928 |        132 | 14.22%         |      796 | 85.78%       |
+-----------+------------+----------------+----------+--------------+
```

SQL code used to for businesses with between 2.5 and 3.5 stars

```
/*To get count and percent of businesses that are open or closed when between 2.5
 and 3.5 stars and review_count > 100*/
Select Count(Distinct b.id) As total_biz,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And (stars Between 2.5 And 3.5)
            And review_count > 100) As biz_closed,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And (stars Between 2.5 And 3.5)
            And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And (stars Between 2.5 And 3.5)
            And review_count > 100) As biz_open,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And (stars Between 2.5 And 3.5)
            And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
Where (stars Between 2.5 And 3.5) And review_count > 100
```

Output with review_count condition added:

```
+-----------+-----------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+-----------+----------------+----------+--------------+
|       254 |         32 | 12.6%          |      222 | 87.4%        |
+-----------+-----------+----------------+----------+--------------+
```

Output without review_count condition added:

```
+-----------+-----------+----------------+----------+--------------+

| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+-----------+----------------+----------+--------------+
|      4064 |        735 | 18.09%         |     3329 | 81.91%       |
+-----------+-----------+----------------+----------+--------------+
```

SQL code used to for businesses with 4 or more stars

```
/*To get count and percent of businesses that are open or closed when greater
than or equal to stars and review_count > 100*/
Select Count(Distinct b.id) As total_biz,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And stars >= 4 And review_count > 100)
            As biz_closed,
        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 0 And stars >= 4 And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_closed,
        (Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And stars >= 4 And review_count > 100) As biz_open,

        (Round(Cast((Select Count(Distinct b.id)
            From business b
            Where is_open = 1 And stars >= 4 And review_count > 100) As Real)/
            Cast(Count(Distinct b.id) As Real)*100, 2))||'%'
            As percent_open
From business b
Where stars >= 4 And review_count > 100
```

Output with review_count condition added:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|       354 |         28 | 7.91%          |      326 | 92.09%       |
+-----------+------------+----------------+----------+--------------+
```

Output without review_count condition added:

```
+-----------+------------+----------------+----------+--------------+
| total_biz | biz_closed | percent_closed | biz_open | percent_open |
+-----------+------------+----------------+----------+--------------+
|      5008 |        653 | 13.04%         |     4355 | 86.96%       |
+-----------+------------+----------------+----------+--------------+
```

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:

We have all heard how being angry can drive up engagement with various websites based and that is why several social media sites configure their algorithms to provoke anger. I wanted to see if there was any correlation between a negative or low `stars` rating the `review_count` of a user.

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

Fortunately, the `user` table has both `review_count` for a `user` as well as an `average_stars` field, so no additional work will have to be done to find the average number of `stars` for a `user`.

iii. Output of your finished dataset:

Please see output on the five subsequent pages, as well as my final thoughts and analysis on the results.

Query 1:

Output when sorting by review_count in descending order:

```
+------------------------+--------------+---------------+
| id                     | review_count | average_stars |
+------------------------+--------------+---------------+
| -G7Zkl1wIWBBmD0KRy_sCw |         2000 |           3.6 |
| -3s52C4zL_DHRK0ULG6qtg |         1629 |          3.42 |
| -8lbUNlXVSoXqaRRiHiSNg |         1339 |          4.11 |
| -K2Tcgh2EKX6e6HqqIrBIQ |         1246 |          3.14 |
| -FZBTkAZEXoP7CYvRV2ZwQ |         1215 |          4.41 |
| --2vR0DIsmQ6WfcSzKWigw |         1153 |           4.4 |
| -gokwePdbXjfS0iF7NsUGA |         1116 |          3.31 |
| -DFCC64NXgqrxlO8aLU5rg |         1039 |          3.71 |
| -8EnCioUmDygAbsYZmTeRQ |          968 |          4.05 |
| -0IiMAZI2SsQ7VmyzJjokQ |          930 |          3.69 |
| -fUARDNuXAfrOn4WLSZLgA |          904 |           3.6 |
| -hKniZN2OdshWLHYuj21jQ |          864 |           4.0 |
| -9da1xk7zgnnfO1uTVYGkA |          862 |           4.1 |
| -B-QEUESGWHPE_889WJaeg |          861 |          3.36 |
| -kLVfaJytOJY2-QdQoCcNQ |          842 |           4.1 |
| -kO6984fXByyZm3_6z2JYg |          836 |          3.47 |
| -lh59ko3dxChBSZ9U7LfUw |          834 |          3.68 |
| -g3XIcCb2b-BD0QBCcq2Sw |          813 |          4.09 |
| -l9giG8TSDBG1jnUBUXp5w |          775 |          3.83 |
| -dw8f7FLaUmWR7bfJ_Yf0w |          754 |          3.62 |
| -AaBjWJYiQxXkCMDlXfPGw |          702 |          3.66 |
| -jt1ACMiZljnBFvS6RRvnA |          696 |          3.27 |
| -IgKkE8JvYNWeGu8ze4P8Q |          694 |          3.89 |
| -hxUwfo3cMnLTv-CAaP69A |          676 |          3.31 |
| -H6cTbVxeIRYR-atxdielQ |          675 |          4.06 |
+------------------------+--------------+---------------+
```

Query 2:

Output when sorting by review_count in ascending order:

```
+------------------------+--------------+---------------+
| id                     | review_count | average_stars |
+------------------------+--------------+---------------+
| --0sXNBv6IizZXuV-nl0Aw |            1 |           5.0 |
| --5BsHjOVLIGoTwjol-V2w |            1 |           1.0 |
| --6u02ZqjZRnwtX3t9bZtQ |            1 |           1.0 |
| --6_wnx2sD1rqOQAQH96wg |            1 |           4.0 |
| --7gZYIAVGCaPT4k0qbbrw |            1 |           5.0 |
| --9yZb1OLNN18HyDXgZrJA |            1 |           4.0 |
| --B1_68kNRRoT9k61Qlu3g |            1 |           5.0 |
| --C5cBJscv6TNMpF_OSJnA |            1 |           5.0 |
| --cAxfHMTBqYGmvkDNXK-g |            1 |           5.0 |
| --d2gPjSkSEQuhncWxs-kw |            1 |           5.0 |
| --DKDJlRHfsvufdGSk_Sdw |            1 |           1.0 |
| --ERQVpqAAoi262TTbLVzQ |            1 |           5.0 |
| --erV6Uzfa42Wk2nf4OIfg |            1 |           5.0 |
| --G7oWdtqbbbmNfe6efMdA |            1 |           5.0 |
| --haaCngcz4NnX_IVSFswA |            1 |           5.0 |
| --HTK7EdtoCKTsNbA17CNA |            1 |           2.0 |
| --I8oeC2I3GXWeI6seyx8g |            1 |           5.0 |
| --JgEXWTirBKGOLHOBr0Wg |            1 |           1.0 |
| --K1aJ5K8ZLIfDd_NjySbA |            1 |           5.0 |
| --K2iq2BCfOwu9CmDm3zCQ |            1 |           5.0 |
| --KKFIKZpMeRy8fLNp0brA |            1 |           5.0 |
| --kMhfqxhJ7sEDiRCSKO0A |            1 |           1.0 |
| --lLPe25C11iuKI1hwgxaQ |            1 |           5.0 |
| --LNODeTnFsPFlS_875W2Q |            1 |           5.0 |
| --LnqQQ2mVdrpPt2UC8d6A |            1 |           2.0 |
+------------------------+--------------+---------------+
```

Query 3:

Output when sorting by average_stars in descending order:

```
+------------------------+--------------+---------------+
| id                     | review_count | average_stars |
+------------------------+--------------+---------------+
| ---94vtJ_5o_nikEs6hUjg |            2 |           5.0 |
| --0sXNBv6IizZXuV-nl0Aw |            1 |           5.0 |
| --1mPJZdSY9KluaBYAGboQ |            5 |           5.0 |
| --26jc8nCJBy4-7r3ZtmiQ |            2 |           5.0 |
| --44NNdtngXMzsxyN7ju6Q |            2 |           5.0 |
| --4ww39MLTS1SBRmCrSmww |            3 |           5.0 |
| --6D_IuxyKTN53pHi904ag |            2 |           5.0 |
| --7D3lFxyMYvs2JYiRrg6Q |            2 |           5.0 |
| --7gZYIAVGCaPT4k0qbbrw |            1 |           5.0 |
| --8KXrwtbo2Szv6zakkzTQ |            2 |           5.0 |
| --AujbGl6SYRaY8SFVNHXA |            3 |           5.0 |
| --B1_68kNRRoT9k61Qlu3g |            1 |           5.0 |
| --C5cBJscv6TNMpF_OSJnA |            1 |           5.0 |
| --cAxfHMTBqYGmvkDNXK-g |            1 |           5.0 |
| --d2gPjSkSEQuhncWxs-kw |            1 |           5.0 |
| --dhSVoOFDBiMCCwDmFccQ |            2 |           5.0 |
| --ERQVpqAAoi262TTbLVzQ |            1 |           5.0 |
| --erV6Uzfa42Wk2nf4OIfg |            1 |           5.0 |
| --fF-8O8ruMCIuP45IwVMQ |            3 |           5.0 |
| --G7oWdtqbbbmNfe6efMdA |            1 |           5.0 |
| --gc9Xg3MPKCy3BPt1g_1A |            5 |           5.0 |
| --gFHlgNyKGY4QJT9pRKJA |            2 |           5.0 |
| --GwB-sktmoAOPBsbAaiow |            2 |           5.0 |
| --haaCngcz4NnX_IVSFswA |            1 |           5.0 |
| --I8oeC2I3GXWeI6seyx8g |            1 |           5.0 |
+------------------------+--------------+---------------+
```

Query 4:

Output when sorting by average_stars in ascending order:

```
+------------------------+--------------+---------------+
| id                     | review_count | average_stars |
+------------------------+--------------+---------------+
| --56y1InAvNoQOD6YYrhVQ |            3 |           1.0 |
| --5BsHjOVLIGoTwjol-V2w |            1 |           1.0 |
| --6u02ZqjZRnwtX3t9bZtQ |            1 |           1.0 |
| --bUghK7FPTx7j6f_44U4g |            2 |           1.0 |
| --DKDJlRHfsvufdGSk_Sdw |            1 |           1.0 |
| --JgEXWTirBKGOLHOBr0Wg |            1 |           1.0 |
| --JS-RvSykutl1DavCbkIg |            2 |           1.0 |
| --kMhfqxhJ7sEDiRCSKO0A |            1 |           1.0 |
| --oa_6IvREwaGutOtE6ZpA |            1 |           1.0 |
| --TvGNywm2I1iwNWZmerBA |            1 |           1.0 |
| --UjDunOU3ZuRzYppmyHgQ |            1 |           1.0 |
| --VMH5hO94TMURIMoy9iQQ |            1 |           1.0 |
| --wiweOmNCw6vg2kbzbwEA |            2 |           1.0 |
| --XroDUidjD1PcmgahDk2w |            1 |           1.0 |
| --YFO6IZDd_-14p5F51ReA |            1 |           1.0 |
| -0-XBCqCPLSGZyn56iH5dw |            1 |           1.0 |
| -05nijFl0po5PwETdKRYwg |            1 |           1.0 |
| -06AUDaZG_VaSVugGvPd0A |            1 |           1.0 |
| -08n6ETKZLHjbm9YSv-NlQ |            1 |           1.0 |
| -0By6XtiEMkjT7UVV5w8rg |            1 |           1.0 |
| -0ckjZ88S5PLSMNYBKitrg |            1 |           1.0 |
| -0cpXx6FYhxB49_9ekj-cg |            1 |           1.0 |
| -0DgO-WJ7yBjAihY_PoUpw |            1 |           1.0 |
| -0fu5TVgRGvysK8WySufOA |            1 |           1.0 |
| -0gS15b5T-Xnw0uaUy6Ymg |            1 |           1.0 |
+------------------------+--------------+---------------+
```

Query 5:

Output when review_count = 1 and average_stars = 1:

```
+----------------------+
| Count(average_stars) |
+----------------------+
|                  473 |
+----------------------+
```

Query 6:

Output when review_count = 1 and average_stars = 5:

```
+----------------------+
| Count(average_stars) |
+----------------------+
|                  937 |
+----------------------+
```

Query 7:

Output for average average_stars:

```
+--------------------+
| Avg(average_stars) |
+--------------------+
|           3.699439 |
+--------------------+
```

As we can see from the six queries above, the results were actually contrary to my initial assumptions. The lowest `review_count` amounts were typically associated with either the highest rating of 5.0 or the lowest of 1.0, I attribute this to one time users who usually would not take the time to go on Yelp and leave a review, but had an experience that was either so good, or so bad, it prompted them to go online and leave a maximally positive (5.0) or maximally negative review (1.0). To find the actual breakdown of `average_stars` for just a `review_count` of 1, I ran the additional queries 5 & 6. The results of these additional queries were refreshing however, in that the 5.0s outweighed the 1.0s by almost double. As for the `average_stars` for the users with the highest `review_counts`, typically they were between 3.0 and 4.5, which is not low or negative by any means, and I attribute this `average_stars` rating to actually a more honest rating by a user, it would be unreasonable that all ratings would be high, so it makes sense that the rating would be dragged down a little. Also, I attribute the result of the `average_stars` of users with the highest `review_counts` being above the average of `average_stars` for all users to the fact that the users with the highest `review_counts` probably were more informed customers and thus did their homework before going out into the marketplace and specifically sought out businesses of high value.

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

Query 1:
```
Select id, review_count, average_stars
From user
Where review_count > 0
Order By review_count Desc
```

Query 2:
```
Select id, review_count, average_stars
From user
Where review_count > 0
Order By review_count
```

Query 3:
```
Select id, review_count, average_stars
From user
Where review_count > 0
Order By average_stars Desc
```

Query 4:
```
Select id, review_count, average_stars
From user
Where review_count > 0
Order By average_stars
```

Query 5:
```
Select Count(average_stars)
From user
Where review_count = 1
And average_stars = 1
```

Query 6:
```
Select Count(average_stars)
From user
Where review_count = 1
And average_stars = 5
```

Query 7:
```
Select Avg(average_stars)
From user
```