Table of Contents

Data Background Analysis

This report analyzes a series of tables from the Yelp database. Yelp is a U.S.-based application that rates restaurants, similar to the Chinese platform, Dianping. The database consists of six tables: business information, business hours, check-ins, reviews, tips, and user information. Data ranges from 2004 to 2017, covering businesses located in the United States, including closed ones.

Data Explanation

The six tables in the Yelp database are yelp_business, yelp_business_hours, yelp_checkin, yelp_review, yelp_tip, and yelp_user. Each table contains the following key data:

yelp_business: Describes basic business information (174,567 records).

business_id	商户ID
name	商户的名称
neighborhood	商户的邻居
address	商户的地址
city	商户所在城市
state	商户所在的州
postal_code	商户的邮政编码
latitude	商户所在位置的纬度
longitude	商户所在位置的精度
stars	商户的星级
review_count	商户的评论数量
is_open	商户是否仍然开放
categories	商户的类别

yelp_business_hours: Describes business hours (174,567 records).

business_id	商户ID
monday	周一营业时间
tuesday	周二营业时间
wednesday	周三营业时间
thursday	周四营业时间
friday	周五营业时间
saturday	周六营业时间
sunday	周日营业时间

yelp_checkin: Describes check-in information (146,350 records).

business_id	商户ID
weekday	注册日
hour	注册的时间
checkins	注册的次数

yelp_review: Contains review information (5,261,668 records).

review_id	评论的ID
user_id	用户ID
business_id	商户ID
stars	评论给商户的星级
date	评论的日期
text	评论的内容
useful	评论获得的“有用”数
funny	评论获得的“有趣”数
cool	评论获得的“酷”数

yelp_tip: Contains tip information (1,098,324 records).

text	评论的内容
date	评论的日期
likes	获得“喜欢”的数量
business_id	商户ID
user_id	用户ID

yelp_user: Contains user information (1,326,100 records)

user_id	用户ID
name	用户名字
review_count	用户发出的评论数
yelping_since	注册时间
friends	好友数
useful	获得“有用”评价的数量
funny	获得“有趣”评价的数量
cool	获得“酷”评价的数量
fans	粉丝数
elite	获得“精英用户称号”的年份
average_stars	用户给出的平均星数
compliment_hot	该用户评论被评价为“很火”的次数
compliment_more	该用户评论被称赞为“想看更多评价”的次数
compliment_profile	未知
compliment_cute	该用户评论被评价为“可爱”的次数
compliment_list	未知
compliment_note	该用户评价被评论的数量
compliment_plain	该用户评论被评价为“清楚”的次数
compliment_cool	该用户评论被评价为“酷”的次数
compliment_funny	该用户评论被评价为“有趣”的次数
compliment_writer	该用户评论被称赞为“作家”的次数
compliment_photos	该用户的照片被点赞的次数

Exploration

The analysis focuses on three main areas to assess the situation and provide suggestions for improving future sales:

1. Customer Perspective:

- Average number of reviews by regular and elite users per month.
- Relationships between positive/negative reviews, review length, and ratings (useful, funny, cool).
- Distribution of star ratings and restaurant star distribution.
- Relationship between review/photo count and fan count, as well as fan count and elite users.
- Most frequently reviewed cities.

2. Restaurant Perspective:

- Distribution of restaurant ratings and their relationships to reviews, ratings, and operating days.
- States with the most restaurants and those with a high proportion of five-star restaurants.
- Relationship between operating days and star ratings.
- Characteristics of closed restaurants.

3. Platform Perspective:

- Number of users registered by year and month, and number of reviews posted each year.
- Proportion of elite users over the years.
- Retention rate of users and elite users by year.

Data Preparation

Before analysis, the data was cleaned by filtering for outliers and splitting tables into currently open and closed businesses. Anomalies in business hours, check-ins, reviews, and tips were filtered out, and irrelevant entries were removed.

1. yelp_business

```
-- Select all columns from the 'yelp_business' table
```

```
SELECT *
FROM `yelp_business`
-- Filter the results where 'name' is NULL
-- or 'state' is NULL
-- or 'stars' is less than 0 or greater than 5
WHERE
   name IS NULL
   OR state IS NULL
   OR stars < 0
   OR stars > 5;
```

No results indicate that all the data are valid values:

business_id	name	neighborhood	address	city	state	postal_code	latitude	longitude	stars	review_count	is_open	categori
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)

Table 1 Filtering outliers in yelp_business

Split the table into 'currently operating businesses' and 'closed businesses' for future use. The code is as follows:

```
SELECT *
FROM `yelp_business`
WHERE is_open=1

SELECT *
FROM `yelp_business`
WHERE is_open=0
```

2. yelp_business_hours

Remove businesses that are closed all seven days of the week and those without operating hours information. The code is as follows:

```
SELECT *
FROM `yelp_business_hours`
WHERE
   monday <> "None"
   OR tuesday <> "None"
   OR wednesday <> "None"
   OR thursday <> "None"
   OR friday <> "None"
   OR saturday <> "None"
   OR sunday <> "None";
```

A selection of the cleaned data results is shown below:

business_id	monday	tuesday	wednesday	thursday	friday	saturday	sunday
--6MefnULPED_I942VcFN	11:0-22:30	11:0-22:30	11:0-22:30	11:0-22:30	11:0-22:30	11:0-22:30	11:0-22:30
--7zmmkVg-IMGaXbuVd0	16:0-22:0	16:0-22:0	16:0-22:0	16:0-22:0	12:0-23:0	12:0-23:0	12:0-20:0
--8LPVSo5i0Oo61X01sV9	8:30-16:30	8:30-16:30	8:30-16:30	8:30-16:30	8:30-16:30	None	None
--9e1ONYQuAa-CB_Rrw7	11:30-14:0	11:30-14:0	11:30-14:0	11:30-14:0	11:30-14:0	11:30-14:0	11:30-14:0
--ab39IjZR_xUf81WyTyHg	10:0-21:0	10:0-21:0	10:0-21:0	10:0-21:0	10:0-21:0	10:0-21:0	11:0-18:0
--cgVkbWTiga3OYTkymK	8:0-17:0	8:0-17:0	8:0-17:0	8:0-17:0	8:0-16:0	None	None
--cjBEbXMI2obtaRHNSFr	None	None	17:0-2:0	17:0-2:0	17:0-2:0	18:0-2:0	None
--cZ6Hhc9F7VkKXxHMVZ	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	12:0-21:0

Table 2 Filtering outliers in yelp_business_hours

3. yelp_checkins

Filtering out the outliers. The code is as follows:

```
SELECT *
FROM `yelp_checkin`
WHERE
   weekday IS NULL
   OR hour IS NULL
   OR checkins IS NULL;
```

No results indicate that all the data are valid values:

business_id	weekday	hour	checkins
(N/A)	(N/A)	(N/A)	(N/A)

Table 3 Filtering outliers in yelp_checkins

4. yelp_review

Filtering out the outliers. The code is as follows:

```
-- Select all columns from the 'yelp_review' table
SELECT *
FROM yelp_review
-- Filter the results where 'review_id' is NULL
-- or 'business_id' is NULL
-- or 'user_id' is NULL
-- or 'date' is NULL
-- or 'stars' is less than 0 or greater than 5
WHERE
   review_id IS NULL
   OR business_id IS NULL
   OR user_id IS NULL
   OR date IS NULL
   OR stars < 0
   OR stars > 5;
```

No results indicate that all the data are valid values:

review_id	user_id	business_id	stars	date	text	useful	funny	cool
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)

Table 4 Filtering outliers in yelp_review

5. yelp_tip

Filtering out the outliers. The code is as follows:

```
-- Select all columns from the 'yelp_tip' table
SELECT *
FROM yelp_tip
-- Filter the results where 'text' is NULL
-- or 'business_id' is NULL
-- or 'user_id' is NULL
-- or 'date' is NULL
-- or 'likes' is less than 0
WHERE
   text IS NULL
   OR business_id IS NULL
   OR user_id IS NULL
   OR date IS NULL
```

```
  OR likes < 0;
```

No results indicate that all the data are valid values:

text	date	likes	business_id	user_id
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)

Table 5 Filtering outliers in yelp_tip

6. yelp_user

First, filter out the outliers. The code is as follows:

```
-- Select all columns from the 'yelp_user' table
SELECT *
FROM `yelp_user`
-- Filter the results where any of the following conditions are met:
-- 'useful' is less than 0
-- 'funny' is less than 0
-- 'cool' is less than 0
-- 'fans' is less than 0
-- 'average_stars' is less than 0
-- 'average_stars' is greater than 5
WHERE
  useful < 0
  OR funny < 0
  OR cool < 0
  OR fans < 0
  OR average_stars < 0
  OR average_stars > 5;
```

The result is shown as the following:

user_id	name	review_count	yelping_since	friends	useful	funny	cool	fans	elite	average_stars	compliment_hot	com
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	

Table 6 Filtering outliers in yelp_user

No results indicate that the values in the column are all normal. Next, remove user information without a name, review, or registration details from the table to generate a cleaned new table. The code is as follows:

```
-- Select all columns from the 'yelp_user' table
-- Where 'name', 'review_count', and 'yelping_since' are not NULL
SELECT *
FROM `yelp_user`
WHERE
  name IS NOT NULL
  AND review_count IS NOT NULL
  AND yelping_since IS NOT NULL;
```

A selection of the cleaned data results is shown below:

user_id	name	review_count	yelping_since	friends	useful	funny	cool	fans	elite	average_star
---1lKK3aKOuomHnwAkA	Monera	246	2007-06-04	Z3UUg88bC1-QGzjZzgRL	67	22	9	15	2010, 2013, 2011, 2012	
---94vtJ_5o_nikEs6hUjg	Joe	2	2016-05-27	idji1BbkbDRgvaQjcpFPyw	0	0	0	0	None	
---cu1hq55BP9DWVXXKH	Jeb	57	2009-04-18	Q9z9X8GVdTu_JtjVDS1nV	34	14	0	0	None	
---fhiwiwBYrvqhpXgcWD	Jed	8	2011-04-20	WJaH7yc4-Ut3Fxp7Ck3C2	2	3	1	0	None	
---PLwSf5gKdIoVnyRHgB	Rae	2	2015-07-31	None	1	0	0	0	None	
---udAKDsn0yQXmzbWQI	Carolyn	48	2014-07-12	nZZxmX2_o1Cj8t3l4Z1Tr	1	0	0	1	None	
--0kuuLmuYBe3Rmu0lycw	Talia	26	2010-03-08	zOvtqb2nR0Ml4r7Yv1Buc	10	2	0	2	None	
--0RtXvcOIE4XbErYca6Rw	Ryan	2	2013-05-30	None	0	0	0	0	None	
--0sXNBv6lizZXuV-nl0Aw	Joe	1	2013-01-09	WywMDFqjC_ViJUXQA60	0	0	0	0	None	
--0WZ5gklOfbUIodJuKfaC	Scott	8	2013-02-19	oBlpkxhgUkXe4SL9Vmqz	0	0	0	0	None	
--104qdWvE99vaoIsj9ZJQ	John	3	2016-04-26	None	0	0	2	0	None	
--1av6NdbEbMiuBr7Aup9	Ron	11	2010-09-26	HlA3Rik2e-T29cF3mQdlu	0	0	1	0	None	
--1mPJZdSY9KluaBYAGbc	Bryan	5	2011-07-04	ZkSwsgxdds5IY1WSMPg1	0	0	0	0	None	

Table 7 Result table after data cleansing in. yelp_user

Data Analysis

- The following code for extracting data is written in MySQL

1. From the Customers Perspecitve：

1) On average, how many reviews does a regular user post per month after creating an account? How many reviews does an elite user post per month on average? How often are their reviews considered useful, funny, or cool, and what are the proportions of each?

First, determine the final date of the data collection. In this case, the most recent user registration date is used as the reference:

```
-- Select distinct 'yelping_since' values
-- from the 'yelp_user' table, ordered by descending date
-- and limit the result to 1 row (the most recent date)
SELECT DISTINCT(yelping_since)
FROM `yelp_user`
ORDER BY yelping_since DESC
LIMIT 1;
```

The result shows that the end date of the data collection is December 11, 2017:

yelping_since
2017-12-11

Table 8 End Date of Data Collection

Calculate the number of days non-elite users have been registered and their average number of reviews per day. The code is as follows:

```
-- Select 'user_id', calculate the registration time as the difference
-- between "2017-12-11" and 'yelping_since', and calculate the average
-- review per day for non-elite users (elite="None").
SELECT
   user_id,
   DATEDIFF("2017-12-11", yelping_since) as register_time,
   (review_count / DATEDIFF("2017-12-11", yelping_since)) as avg_review_day
FROM `yelp_user`
WHERE elite = "None";
```

A snippet of the results is shown below:

user_id	register_time	avg_review_day
---94vtJ_5o_nikEs6hUjg	563	0.0036
---cu1hq55BP9DWVXXKHZg	3159	0.018
---fhiwiwBYrvqhpXgcWDQ	2427	0.0033
---PLwSf5gKdIoVnyRHgBA	864	0.0023
---udAKDsn0yQXmzbWQNSw	1248	0.0385
--0kuuLmuYBe3Rmu0Iycww	2835	0.0092
--0RtXvcOIE4XbErYca6Rw	1656	0.0012
--0sXNBv6IizZXuV-nl0Aw	1797	0.0006
--0WZ5gklOfbUIodJuKfaQ	1756	0.0046
--104qdWvE99vaoIsj9ZJQ	594	0.0051
--1av6NdbEbMiuBr7Aup9A	2633	0.0042
--1mPJZdSY9KluaBYAGboQ	2352	0.0021
--26jc8nCJBy4-7r3ZtmiQ	1226	0.0016
--2bpE5vyR-2hAP7sZZ4lA	791	0.0291

Table 9 Number of Days Regular Users Have Been Registered and Average Daily Reviews

Calculate the average value of this result:

```
-- Calculate the average of 'avg_review_day' values for non-elite users.
SELECT AVG(sub.avg_review_day) as avg_review_day_nelite
FROM (
    -- Subquery to calculate 'avg_review_day' for non-elite users
    SELECT
        user_id,
        DATEDIFF("2017-12-11", yelping_since) as register_time,
        (review_count / DATEDIFF("2017-12-11", yelping_since)) as avg_review_day
    FROM `yelp_user`
    WHERE elite = "None"
) as sub;
```

The average number of reviews posted per day by regular users is 0.00959, as shown in the figure below:

avg_review_day_nelite
0.00959149

Table 10 Average Daily Reviews Posted by Regular Users

Calculate the number of useful, funny, and cool votes these reviews received. The code is as follows:

```
-- Calculate the total review count for non-elite users,
-- as well as the rates of 'useful,' 'funny,' and 'cool' votes per review.
SELECT
    SUM(review_count) as total_review_count,
    SUM(useful) / SUM(review_count) as useful_rate_ne,
    SUM(funny) / SUM(review_count) as funny_rate_ne,
    SUM(cool) / SUM(review_count) as cool_rate_ne
FROM `yelp_user`
WHERE elite = "None";
```

It was found that the useful, funny, and cool votes accounted for 0.59, 0.21, and 0.21 times the total number of reviews, respectively:

sum(review_count)	useful_rate_ne	funny_rate_ne	cool_rate_ne
16857521	0.5905	0.2141	0.2072

Table 11 Total Reviews and Various Feedback Statistics for Regular Users

Using the same method, analyze the elite users. First, determine the average number of reviews elite users post daily. The code is as follows:

```
-- Calculate the average of 'avg_review_day' values for elite users.
SELECT AVG(sub.avg_review_day) as avg_review_day_elite
FROM (
   -- Subquery to calculate 'avg_review_day' for elite users
   SELECT
      user_id,
      DATEDIFF("2017-12-11", yelping_since) as register_time,
      (review_count / DATEDIFF("2017-12-11", yelping_since)) as avg_review_day
   FROM `yelp_user`
   WHERE elite <> "None"
) as sub;
```

It can be seen that the average number of reviews posted per day by elite users is 0.0979, which is approximately 10.2 times that of regular users.

avg_review_day_elite
0.09788794

Table 12 Average Reviews per day for Elite Users

Next, calculate the proportion of useful, funny, and cool votes received by elite users' reviews. The code is as follows:

```
-- Calculate the total review count for elite users,
-- as well as the rates of 'useful,' 'funny,' and 'cool' votes per review.
SELECT
   SUM(review_count) as total_review_count,
   SUM(useful) / SUM(review_count) as useful_rate_e,
   SUM(funny) / SUM(review_count) as funny_rate_e,
   SUM(cool) / SUM(review_count) as cool_rate_e
FROM `yelp_user`
WHERE elite <> "None";
```

The result is shown below:

sum(review_count)	useful_rate_e	funny_rate_e	cool_rate_e
13798162	2.0988	1.1426	1.6309

Table 13 The total number of reviews from premium users and various feedback statistics

It was found that the useful, funny, and cool votes received by elite users are 2.10, 1.14, and 1.63 times their total reviews, which are 3.56, 5.43, and 7.76 times higher than those of regular users, respectively. Elite users visit more businesses and post more reviews than regular users, and their reviews are more likely to receive positive feedback, generally being of higher quality.

It was also found that, for both user types, reviews are far more likely to receive a useful

vote than funny or cool votes. This indicates that when writing reviews, users tend to focus more on providing valuable content rather than emphasizing style or word choice. Likewise, other users are more interested in whether the review content is useful rather than if it is funny or cool.

2) Are reviews with a high number of useful votes more likely to be positive or negative? What are the characteristics of the length of these reviews? How do funny and cool votes correlate?

First, calculate the number of reviews with more than 1,000, 500, 200, and 100 useful votes, along with the average rating given by customers and the average length of these reviews. The code and results are as follows:

```
-- Calculate the count of reviews with 'useful' values greater than 1000,
-- the average star rating rounded to one decimal place,
-- and the average length of reviews rounded to the nearest whole number.
SELECT
   COUNT(sub.text) as review_count,
   ROUND(AVG(sub.stars), 1) as avg_stars,
   ROUND(AVG(sub.len), 0) as avg_length
FROM (
   -- Subquery to select relevant data from yelp_review
   SELECT stars, text, LENGTH(text) as len, useful
   FROM yelp_review
   WHERE useful > 1000
) as sub;
```

For reviews with more than 1,000 useful votes, there are a total of 11 reviews, with an average rating of 1.4 stars for the restaurant/business and an average review length of 2,204 words.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
11	1.3	2204

Table 14 The average star rating and review length for users when the useful value is greater than 1000

When the number of useful votes exceeds 500, the code is as follows:

```
-- Calculate the count of reviews with 'useful' values greater than 500,
-- the average star rating rounded to one decimal place,
-- and the average length of reviews rounded to the nearest whole number.
SELECT
   COUNT(sub.text) as review_count,
   ROUND(AVG(sub.stars), 1) as avg_stars,
   ROUND(AVG(sub.len), 0) as avg_length
FROM (
   -- Subquery to select relevant data from yelp_review
   SELECT stars, text, LENGTH(text) as len, useful
   FROM yelp_review
   WHERE useful > 500
) as sub;
```

For reviews with more than 200 useful votes, there are a total of 138 reviews, with an average rating of 1.8 stars for the restaurant and an average review length of 1,342 words.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
38	1.3	1456

Table 15 The average star rating and review length for users when the useful value is greater than 500

When the number of useful votes exceeds 200, the code is as follows:

```
-- Calculate the count of reviews with 'useful' values greater than 200,
-- the average star rating rounded to one decimal place,
-- and the average length of reviews rounded to the nearest whole number.
SELECT
   COUNT(sub.text) as review_count,
   ROUND(AVG(sub.stars), 1) as avg_stars,
   ROUND(AVG(sub.len), 0) as avg_length
FROM (
   -- Subquery to select relevant data from yelp_review
   SELECT stars, text, LENGTH(text) as len, useful
   FROM yelp_review
   WHERE useful > 200
) as sub;
```

For reviews with more than 200 useful votes, there are a total of 138 reviews, with an average rating of 1.8 stars for the restaurant and an average review length of 1,342 words.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
138	1.8	1342

Table 16 The average star rating and review length for users when the useful value is greater than 200

When the number of useful votes exceeds 100, the code is as follows:

```
-- Calculate the count of reviews with 'useful' values greater than 100,
-- the average star rating rounded to one decimal place,
-- and the average length of reviews rounded to the nearest whole number.
SELECT
   COUNT(sub.text) as review_count,
   ROUND(AVG(sub.stars), 1) as avg_stars,
   ROUND(AVG(sub.len), 0) as avg_length
FROM (
   -- Subquery to select relevant data from yelp_review
   SELECT stars, text, LENGTH(text) as len, useful
   FROM yelp_review
   WHERE useful > 100
) as sub;
```

For reviews with more than 100 useful votes, there are a total of 408 reviews, with an average rating of 2.4 stars for the restaurant and an average review length of 1,479 words

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
408	2.4	1479

Table 17 The average star rating and review length for users when the useful value is greater than 100

From this data, it can be seen that review length is positively correlated with the likelihood of receiving useful votes. Additionally, reviews with more useful votes tend to give lower ratings to restaurants. This suggests that compared to positive reviews, people often believe that negative reviews provide more useful information about the restaurant.

The same analysis was conducted for funny and cool votes. Since users are most likely to give useful votes, the number of funny votes is relatively small. Therefore, in this report, the thresholds of 500, 200, and 100 are chosen for funny votes, and the code and results are as follows:

When the funny count is greater than 500:

```
-- Calculate the count of reviews with 'funny' values greater than 500,
-- the average star rating rounded to one decimal place,
-- and the average length of reviews rounded to the nearest whole number.
SELECT
   COUNT(sub.text) as review_count,
   ROUND(AVG(sub.stars), 1) as avg_stars,
   ROUND(AVG(sub.len), 0) as avg_length
FROM (
   -- Subquery to select relevant data from yelp_review
   SELECT stars, text, LENGTH(text) as len, useful
   FROM yelp_review
   WHERE funny > 500
) as sub;
```

There are 16 reviews in total. On average, these reviews gave the restaurant a rating of 3.3 stars, and the average review length was 1,752 characters.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
16	3.3	1752

Table 18 Average star rating and review length when funny count is greater than 500.

When the funny count is greater than 200:

```
select count(sub.text), round(avg(sub.stars),1), round(avg(sub.len),0)
from (SELECT stars, text, length(text) as len, funny
FROM yelp_review
WHERE funny>200
) as sub
```

There are 86 reviews in total. On average, these reviews gave the restaurant a rating of 3.6 stars, and the average review length was 1,323 characters.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
86	3.6	1323

Table 19 Average star rating and review length when funny count is greater than 200

When the funny count is greater than 100:

```
select count(sub.text), round(avg(sub.stars),1), round(avg(sub.len),0)
from(SELECT stars, text, length(text) as len, funny
FROM yelp_review
WHERE funny>100
) as sub
```

There are 261 reviews in total. On average, these reviews gave the restaurant a rating of 3.6 stars, and the average review length was 1,386 characters.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
261	3.6	1386

Table 20 Average star rating and review length when funny count is greater than 100

For cool, the count is lower, so the breakpoints are still 500, 200, and 100:

```
select count(sub.text), round(avg(sub.stars),1), round(avg(sub.len),0)
from(SELECT stars, text, length(text) as len, cool
FROM yelp_review
WHERE cool>500
) as sub
```

When the cool count is greater than 500, there is only 1 review. This review gave the restaurant an average rating of 1.0 star and had a length of 4,133 characters.

This review also received both useful and funny feedback. This review can be considered an outlier, so it's not representative in this context and will be excluded from further analysis.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
1	1.0	4133

Table 21 The average star rating and review length when cool count is greater than 500

When the cool count is greater than 200:

```
select count(sub.text), round(avg(sub.stars),1), round(avg(sub.len),0)
from(SELECT stars, text, length(text) as len, cool
FROM yelp_review
WHERE cool>200
) as sub
```

There are 26 reviews in total. On average, these reviews gave the restaurant a rating of 3.4 stars, and the average review length was 1,584 characters.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
26	3.4	1584

Table 22 Average star rating and review length when cool count is greater than 200

When the cool count is greater than 100:

```
select count(sub.text), round(avg(sub.stars),1), round(avg(sub.len),0)
from(SELECT stars, text, length(text) as len, cool
FROM yelp_review
```

```
WHERE cool>100
) as sub
```

There are 151 reviews in total. On average, these reviews gave the restaurant a rating of 3.6 stars, and the average review length was 1,721 characters.

count(sub.text)	round(avg(sub.stars),1)	round(avg(sub.len),0)
151	3.6	1721

Table 23 Average star rating and review length when cool count is greater than 100

Based on the above data, it seems that the length of a review and its star rating for a restaurant do not significantly affect whether a review is perceived as "cool" or "funny" by users. This differs from how users assess whether a review is "useful."

(3) What is the average user rating? What is the distribution of restaurant star ratings for each user rating?

First, for restaurants with different user ratings, the average star rating for these restaurants is calculated. The code is as follows:

```
-- Calculate the average business star rating for each unique review star rating,
-- ordering the results by review star rating in descending order.
SELECT
   yelp_review.stars,
   ROUND(AVG(yelp_business.stars), 2) as avg_busi_star
FROM
   yelp_review
LEFT JOIN
   yelp_business
ON
   yelp_review.business_id = yelp_business.business_id
GROUP BY
   yelp_review.stars
ORDER BY
   yelp_review.stars DESC;
```

The results are as follows: The left side of the table shows the user ratings, while the right side shows the average star ratings of restaurants for each user rating:

stars	avg_busi_star
5	4.08
4	3.74
3	3.52
2	3.37
1	3.03

Table 24 Distribution of restaurant star ratings for each user rating

A chart is created with user ratings on the horizontal axis and restaurant average star ratings on the vertical axis, as shown below:

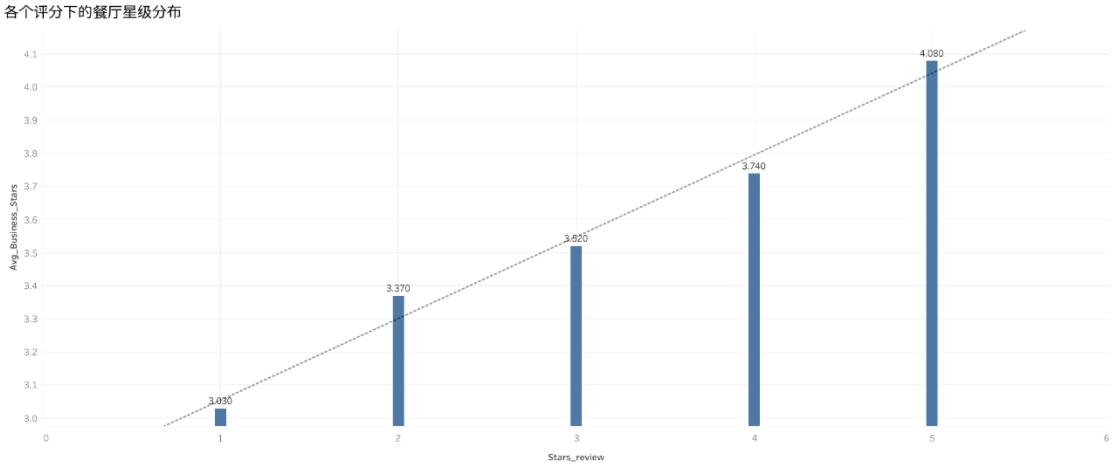

Table 25 Trend chart of restaurant star rating distribution for each user rating

It can be observed that higher user ratings correspond to higher average restaurant star ratings, with a near-uniform distribution, which aligns with common sense.

Next, for restaurants with a 5-star user rating, the proportion of restaurants for each star rating is calculated. The code is as follows:

```
-- Calculate the count of businesses, count of reviews with a star rating of 5,
-- and the ratio of businesses to reviews with a star rating of 5.
SELECT
    yelp_business.stars,
    COUNT(yelp_business.business_id) AS count_busi,
    COUNT(review_id) AS count_re,
    ROUND(COUNT(yelp_business.business_id) / COUNT(review_id), 2) AS rate_5
FROM
    yelp_review
LEFT JOIN
    yelp_business
ON
    yelp_review.business_id = yelp_business.business_id
WHERE
    yelp_review.stars = 5
GROUP BY
    yelp_business.stars
ORDER BY
yelp_business.stars DESC;
```

The results are as follows:

stars	count_busi
5	299171
4.5	620406
4	771721
3.5	364200
3	134981
2.5	46643
2	13296
1.5	2822
1	107

Table 26 Proportion of restaurant star ratings for 5-star user ratings

Histogram Plot:

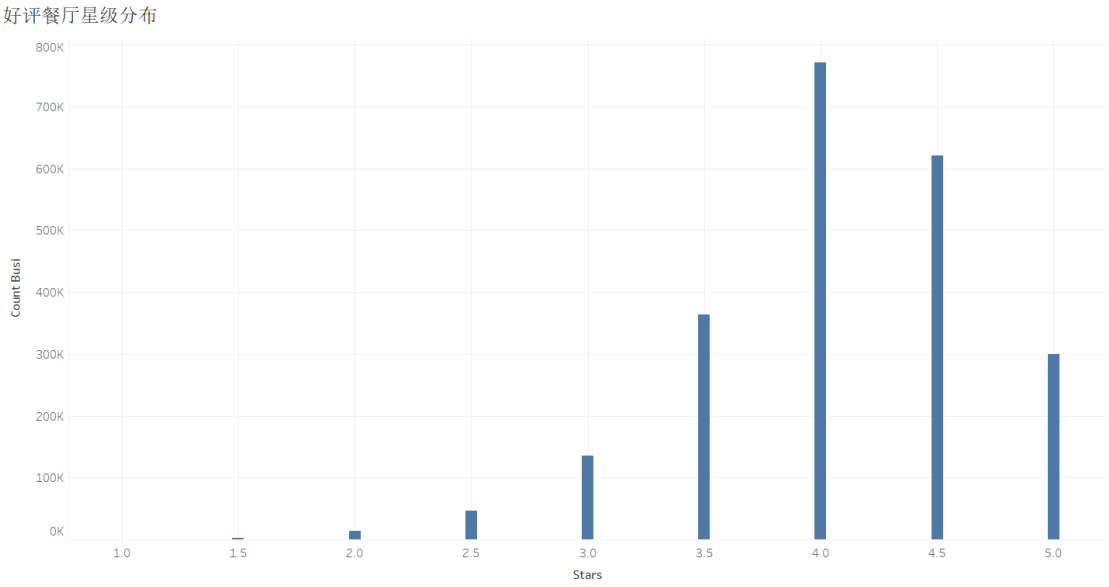

Table 27 Proportion of restaurant star ratings for 5-star user ratings

The distribution of star ratings for restaurants that received a 5-star user rating differs from the distribution of all restaurant star ratings. Specifically, 5-star ratings are more concentrated in restaurants with 4.0 or higher star ratings, while restaurants with less than 3 stars almost never receive 5-star user ratings.

Next, for restaurants with a 1-star user rating, the proportion of restaurants for each star rating is calculated. The code is as follows:

```
SELECT yelp_business.stars, count(yelp_business.business_id) AS count_busi
FROM yelp_review LEFT JOIN yelp_business
ON yelp_review.business_id=yelp_business.business_id
WHERE yelp_review.stars = 1
GROUP BY yelp_business.stars
ORDER BY yelp_business.stars DESC
```

The results are as follows:

stars	count_busi
5	2427
4.5	35125
4	122201
3.5	163031
3	152475
2.5	118159
2	73066
1.5	43833
1	21046

Table 28 Proportion of restaurant star ratings for 1-star user ratings

Histogram Plot:

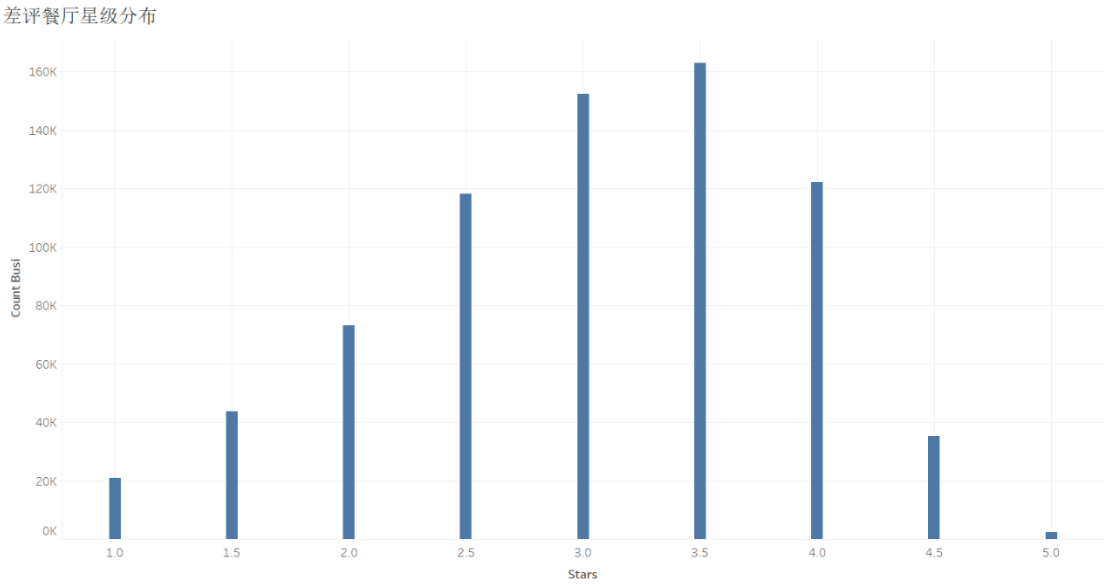

Table 29 Proportion of restaurant star ratings for 1-star user ratings

The distribution of star ratings for restaurants that received a 1-star user rating is similar to the overall distribution of restaurant star ratings. It can be observed that negative ratings are not significantly influenced by the restaurant's overall star rating. However, the low negative rating ratio for 5-star restaurants suggests that a low rate of negative reviews may be an important factor in maintaining a 5-star rating.

(4) Relationship between the number of reviews, photos, and the number of followers? Relationship between the number of followers and elite users?

First, the relationship between the number of reviews, photos, and followers for elite users is calculated using the following code:

```
SELECT sum(fans), sum(review_count), sum(compliment_photos), sum(review_count)/sum(
fans)as fans_re_rate_e, sum(compliment_photos)/sum(fans)as fans_ph_rate_e
FROM `yelp_user`
WHERE elite<>"None"
```

The results are as follows: The columns represent "total number of followers," "total number of reviews," "total number of photos," "ratio of reviews to followers," and "ratio of photos to followers":

sum(fans)	sum(review_count)	sum(compliment_photos)	fans_re_rate_e	fans_ph_rate_e
1334249	13798162	1347109	10.3415	1.0096

Table 30 Relationship between the number of reviews, photos, and followers for elite users

It can be observed that elite users have roughly 10 times as many reviews as followers, while the number of followers is almost equal to the number of photos they post.

Next, the same indicators for non-elite users are calculated using the following code:

```
SELECT sum(fans), sum(review_count), sum(compliment_photos), sum(review_count)/sum(
fans)as fans_re_rate_e, sum(compliment_photos)/sum(fans)as fans_ph_rate_e
FROM `yelp_user`
WHERE elite="None"
```

The results are as follows:

sum(fans)	sum(review_count)	sum(compliment_photos)	fans_re_rate_e	fans_ph_rate_e
598242	16857521	227730	28.1784	0.3807

Table 31 Relationship between the number of reviews, photos, and followers for non-elite users

It can be seen that non-elite users post approximately 28 times as many reviews as they have followers, and the number of photos they post is about 0.38 times their number of followers.

Comparison between elite and non-elite users:

The data shows that although the total number of reviews by elite and non-elite users is similar, reviews by elite users attract significantly more followers, indicating a higher chance of gaining followers. Elite users post far more photos than non-elite users—about six times more—while their total number of followers is less than three times that of non-elite users. This suggests that compared to reviews, photos are less likely to attract attention and followers.

The follower count for non-elite users is calculated using the following code:

```
SELECT count(user_id), sum(fans), sum(fans)/count(user_id) as fans_rate_ne
FROM `yelp_user`
WHERE elite="None"
```

The results are as follow:

count(user_id)	sum(fans)	fans_rate_ne
1265282	598242	0.4728

Table 32 Distribution of followers among non-elite users

The total number of followers for non-elite users is about 0.47 times the number of non-elite users, averaging less than one follower per user, indicating that many non-elite users have no followers at all.

Next, the follower count for elite users is calculated using the following code:

```
SELECT count(user_id), sum(fans), sum(fans)/count(user_id) as fans_rate_e
FROM `yelp_user`
WHERE elite<>"None"
```

The results are as follow:

count(user_id)	sum(fans)	fans_rate_e
60818	1334249	21.9384

Table 33 Distribution of followers among elite users

On average, elite users have about 22 followers each, more than 45 times the number for non-elite users. It can be speculated that the number of followers is a crucial factor when Yelp selects elite users each year, as a higher follower count not only reflects the user's higher engagement and contribution to the platform but also brings more traffic, which is vital for the platform's sustainability.

(5) Which cities are most frequently reviewed by users?

First, the number of reviews for each state is calculated using the following code:

```
SELECT yelp_business.state, COUNT(yelp_review.review_id) AS count_re_state
FROM yelp_review LEFT JOIN yelp_business
ON yelp_review.business_id = yelp_business.business_id
GROUP BY yelp_business.state
ORDER BY count_re_state DESC
```

Here is an excerpt of the results:

state	count_re_state
NV	1824442
AZ	1627792
ON	634366
NC	307665
OH	243768
PA	229850
QC	146371
WI	109751
EDH	47889
IL	36467
BW	35400
SC	10860

Table 34 Cities most frequently reviewed by users

The percentage distribution is represented in a pie chart as follows:

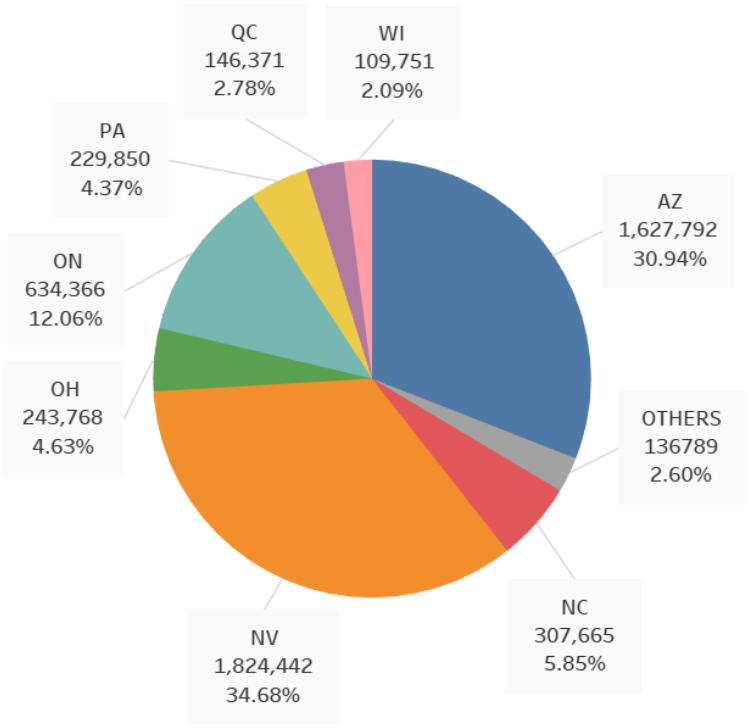

Table 35 Percentage distribution of cities most frequently reviewed by users

The top two states account for 65.62% of all reviews: Nevada, with 34.68%, and Arizona, with 30.94%. Following these are North Carolina, Ohio, Pennsylvania, and others with high percentages. The high number of reviews in these states indicates that users are mainly active in these cities.

2. From the Business Perspective

(1) Number of restaurants by star rating? What is the average star rating for restaurants? What is the relationship between a restaurant's star rating and the number of reviews, the average review rating, and the ratio of positive to negative reviews?

First, the average star rating of restaurants is calculated using the following code:

```
SELECT round(avg(stars),1)
FROM yelp_business
```

The result shows that the average star rating for all restaurants is 3.6 stars:

round(avg(stars),1)
3.6

Table 36 Average star rating of business

Since the star ratings range from 1 to 5 stars with 0.5-star increments, if the ratings were normally distributed, the average rating would be around 3 stars. The data shows an average rating higher than 3 stars, indicating that most restaurants receive a relatively high rating of 3

stars or above, suggesting that the platform tends to avoid giving low ratings to restaurants.

Next, the number of restaurants by each star rating is calculated:

```
SELECT stars, count(business_id)
FROM yelp_business
GROUP BY stars
ORDER BY stars DESC
```

The results show the star rating on the left and the corresponding number of restaurants on the right:

stars	count(business_id)
5.0	27540
4.5	24796
4.0	33492
3.5	32038
3.0	23142
2.5	16148
2.0	9320
1.5	4303
1.0	3788

Table 37 Number of restaurants by star rating

A bar chart with star rating on the horizontal axis and number of restaurants on the vertical axis is created as follows:

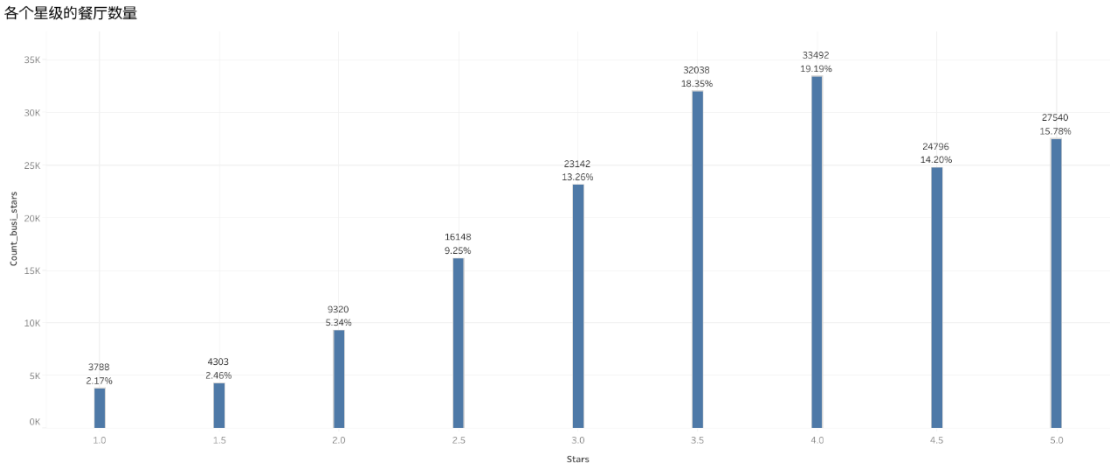

Table 38 Bar chart of the number of restaurants by star rating

The histogram shows a left-skewed distribution, with higher-star restaurants being more prevalent than lower-star ones, which aligns with the information presented by the average rating. It can be seen that 4.0-star restaurants are the most common, accounting for 19.19%, followed closely by 3.5-star restaurants, which account for 18.35%. Most restaurants achieve a level of service that generally satisfies customers.

Notably, there are a significant number of 5-star restaurants, comprising 15.78% of all

restaurants, even surpassing the number of 4.5-star restaurants. On one hand, user ratings can influence a restaurant's star rating, and when users have an excellent experience, they tend to give a perfect score as a sign of appreciation, support, and admiration, often overlooking minor flaws. In such cases, these shortcomings are often considered as "minor blemishes" or "unique characteristics." On the other hand, the platform tends to promote 5-star restaurants over 4.5-star ones, as it benefits the platform to advertise a restaurant as being "perfect," which is a powerful marketing tool. Restaurants promoted by the platform can also offer exclusive discounts on Yelp as a way to reciprocate the platform, attracting significant traffic for both the business and Yelp.

For each star rating of restaurants, the average user review rating and the number of reviews are calculated using the following code:

```
-- Calculate the average review star rating and count of reviews for each business star rating,
-- ordering the results by business star rating in descending order.
SELECT
   yelp_business.stars,
   ROUND(AVG(yelp_review.stars), 2) as avg_re_star,
   COUNT(review_id) as count_re
FROM
   yelp_business
RIGHT JOIN
   yelp_review
ON
   yelp_business.business_id = yelp_review.business_id
GROUP BY
   yelp_business.stars
ORDER BY
   yelp_business.stars DESC;
```

The results show the restaurant star rating on the left, the average user review rating in the middle, and the number of user reviews on the right:

stars	avg_re_star	count_re
5.0	4.91	315730
4.5	4.45	909666
4.0	4.00	1670164
3.5	3.52	1193114
3.0	3.03	650162
2.5	2.53	311991
2.0	2.03	130194
1.5	1.54	58598
1.0	1.07	22049

Table 39 Average user review rating and number of reviews for each star rating of restaurants

A chart is created with restaurant star rating on the horizontal axis, and average user rating and number of reviews on the vertical axes:

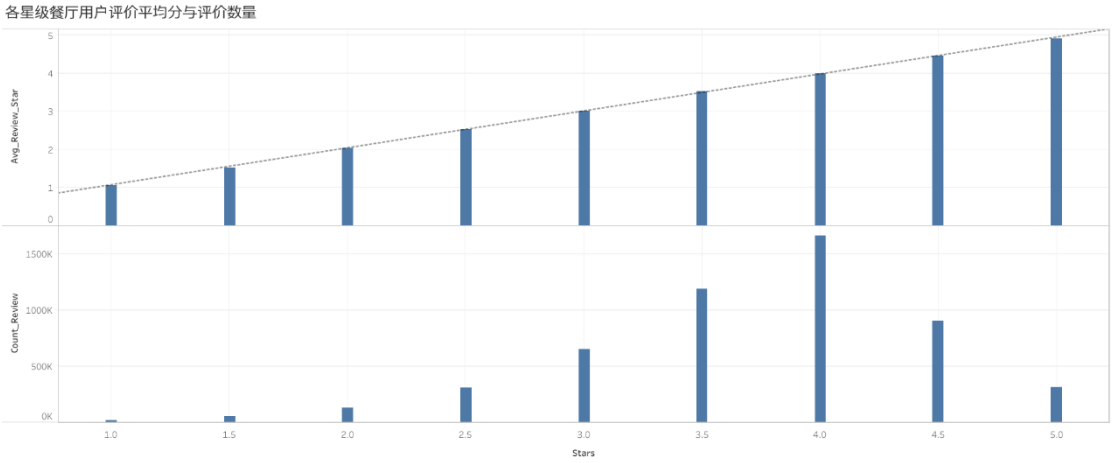

Table 40 Average user rating and number of reviews for each star rating of restaurants

The top histogram shows the trend of average user ratings as restaurant star ratings increase. The trend line indicates that average user ratings increase with higher restaurant star ratings, roughly following a uniform distribution, which is consistent with common sense.

The lower trend line shows the change in the number of user reviews as restaurant star ratings vary, displaying a left-skewed distribution that peaks at 4.0-star restaurants. This can be attributed to 4.0-star restaurants being the most common and most accessible, while also meeting the needs of most users for everyday dining, exploration, and socializing. The chart also reveals that the number of reviews for 5-star restaurants is relatively low, especially considering the large number of 5-star restaurants. Combining this with the earlier analysis, it further suggests that the platform plays a significant role in the selection of 5-star restaurants. Other factors may include the higher price point and limited seating capacity of 5-star restaurants.

(2) Which states have the most restaurants? Which states have the highest proportion of 5-star restaurants? Which states have the highest positive review rates? How does this relate to geographic location?

First, the total number of restaurants in each state is calculated using the following code:

```
SELECT state, count(state)
FROM yelp_business
GROUP BY state
ORDER BY count(state) DESC
```

Here is an excerpt of the results ranked from high to low:

state	count(state)
AZ	52214
NV	33086
ON	30208
NC	12956
OH	12609
PA	10109
QC	8169
WI	4754
EDH	3795
BW	3118
IL	1852
SC	679
MLN	208

Table 41 Ranking of the number of restaurants by state (excerpt)

The results are displayed as a pie chart:

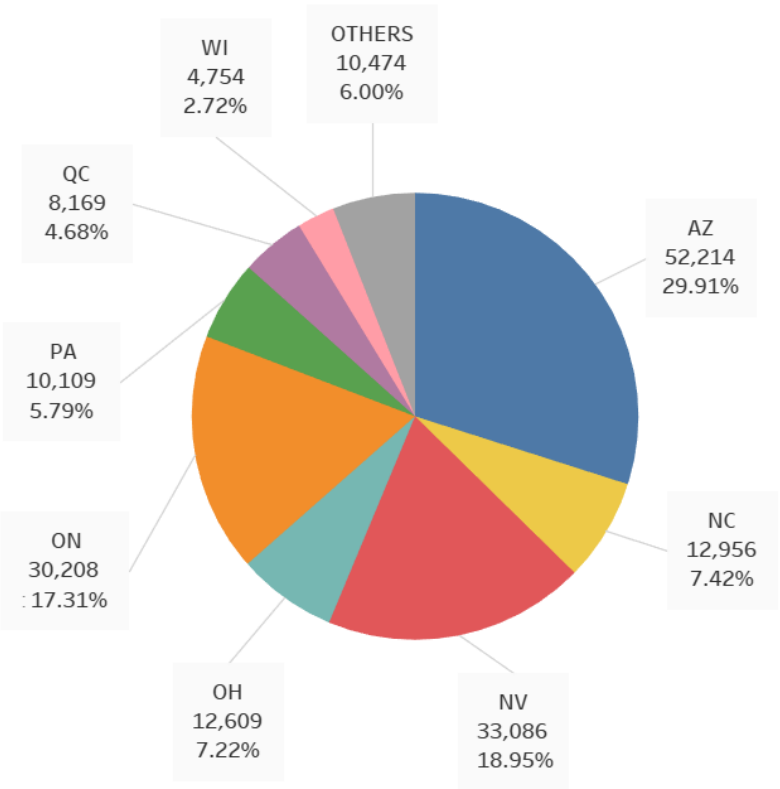

Table 42 Percentage distribution of restaurants by state

The results are similar to the most frequently reviewed states. The top two states account for 48.86% of all reviews: Arizona, with 29.91%, and Nevada, with 18.95%. Next in line are North Carolina, Ohio, Pennsylvania, and others, which also correlate with the number of user reviews. For restaurants, being established in these states could mean more user attention but also fiercer competition among peers.

Next, the proportion of restaurants with different star ratings in each state is calculated using the following code:

```
-- Calculate statistics related to star ratings for businesses grouped by state.
SELECT
    state,
    COUNT(state) as count_state,
    SUM(CASE stars WHEN 5.0 THEN 1 ELSE 0 END) as sum_5,
    (SUM(CASE stars WHEN 5.0 THEN 1 ELSE 0 END) / COUNT(state)) as rate_5,
    SUM(CASE stars WHEN 4.5 THEN 1 ELSE 0 END) as sum_4_5,
    (SUM(CASE stars WHEN 4.5 THEN 1 ELSE 0 END) / COUNT(state)) as rate_4_5,
    SUM(CASE stars WHEN 4.0 THEN 1 ELSE 0 END) as sum_4,
    (SUM(CASE stars WHEN 4.0 THEN 1 ELSE 0 END) / COUNT(state)) as rate_4,
    SUM(CASE stars WHEN 3.5 THEN 1 ELSE 0 END) as sum_3_5,
    (SUM(CASE stars WHEN 3.5 THEN 1 ELSE 0 END) / COUNT(state)) as rate_3_5,
    SUM(CASE stars WHEN 3.0 THEN 1 ELSE 0 END) as sum_3,
    (SUM(CASE stars WHEN 3.0 THEN 1 ELSE 0 END) / COUNT(state)) as rate_3,
    SUM(CASE stars WHEN 2.5 THEN 1 ELSE 0 END) as sum_2_5,
    (SUM(CASE stars WHEN 2.5 THEN 1 ELSE 0 END) / COUNT(state)) as rate_2_5,
    SUM(CASE stars WHEN 2.0 THEN 1 ELSE 0 END) as sum_2,
    (SUM(CASE stars WHEN 2.0 THEN 1 ELSE 0 END) / COUNT(state)) as rate_2,
    SUM(CASE stars WHEN 1.5 THEN 1 ELSE 0 END) as sum_1_5,
    (SUM(CASE stars WHEN 1.5 THEN 1 ELSE 0 END) / COUNT(state)) as rate_1_5,
    SUM(CASE stars WHEN 1.0 THEN 1 ELSE 0 END) as sum_1,
    (SUM(CASE stars WHEN 1.0 THEN 1 ELSE 0 END) / COUNT(state)) as rate_1
FROM
    yelp_business
GROUP BY
    state
ORDER BY
    count_state DESC;
```

The results show the number of restaurants, the number of restaurants for each star rating, and the corresponding proportions for each state:

state	count(state)	sum_5	rate_5	sum_4_5	rate_4_5	sum_4	rate_4	sum_3_5	rate_3_5	sum_3	rate_3	sum_2_5	rate_2_5	sum_2	rate_2	sum_1_5	rate_1_5	sum_1	rate_1
AZ	52214	11698	0.224	7441	0.1425	8928	0.171	8207	0.1572	5995	0.1148	4632	0.0887	2765	0.053	1275	0.0244	1273	0.0244
NV	33086	6631	0.2004	4924	0.1488	5978	0.1807	5426	0.164	4040	0.1221	2827	0.0854	1790	0.0541	783	0.0237	687	0.0208
ON	30208	2219	0.0735	3268	0.1082	5954	0.1971	6724	0.2226	5258	0.1741	3332	0.1103	1869	0.0619	860	0.0285	724	0.024
NC	12956	1893	0.1461	1706	0.1317	2422	0.1869	2445	0.1887	1767	0.1364	1313	0.1013	733	0.0566	368	0.0284	309	0.0238
OH	12609	1578	0.1251	1676	0.1329	2473	0.1961	2474	0.1962	1711	0.1357	1284	0.1018	750	0.0595	343	0.0272	320	0.0254
PA	10109	1310	0.1296	1561	0.1544	1934	0.1913	2011	0.1989	1372	0.1357	952	0.0942	534	0.0528	234	0.0231	201	0.0199
QC	8169	542	0.0663	1579	0.1933	2128	0.2605	1687	0.2065	1064	0.1302	621	0.076	304	0.0372	161	0.0197	83	0.0102
WI	4754	673	0.1416	699	0.147	993	0.2089	923	0.1942	570	0.1199	464	0.0976	226	0.0475	120	0.0252	86	0.0181
EDH	3795	229	0.0603	745	0.1963	1230	0.3241	836	0.2203	435	0.1146	198	0.0522	78	0.0206	33	0.0087	11	0.0029
BW	3118	330	0.1058	654	0.2097	799	0.2563	614	0.1969	421	0.135	186	0.0597	78	0.025	19	0.0061	17	0.0055
IL	1852	222	0.1199	265	0.1431	313	0.169	338	0.1825	279	0.1506	193	0.1042	123	0.0664	70	0.0378	49	0.0265
SC	679	132	0.1944	85	0.1252	115	0.1694	111	0.1635	103	0.1517	64	0.0943	32	0.0471	19	0.028	18	0.0265
MLN	208	12	0.0577	41	0.1971	58	0.2788	55	0.2644	16	0.0769	10	0.0481	9	0.0433	5	0.024	2	0.0096
HLD	179	9	0.0503	31	0.1732	43	0.2402	42	0.2346	32	0.1788	16	0.0894	4	0.0223	0	0	2	0.0112
NYK	152	17	0.1118	28	0.1842	36	0.2368	30	0.1974	20	0.1316	7	0.0461	6	0.0395	6	0.0395	2	0.0132
CHE	143	10	0.0699	27	0.1888	29	0.2028	26	0.1818	23	0.1608	20	0.1399	6	0.042	2	0.014	0	0

Table 43 Number and proportion of restaurants by star rating and state

This table contains a wealth of information. It shows that no state has a significantly higher proportion of low-star restaurants, and the proportions of non-5-star restaurants fluctuate only slightly across states, irrespective of the total number of restaurants in each state. Arizona and Nevada not only have a large number of restaurants but also a higher proportion of 5-star restaurants. This indicates that the selection of 5-star restaurants is influenced by the state where the restaurant is located.

Both Arizona and Nevada are located in the southwestern United States, with large

areas adjacent to California and Hawaii, and Yelp's headquarters are in California. These coastal locations benefit from the traffic from California, and their proximity to Yelp's headquarters may lead the platform to focus on business development in these states, resulting in a larger number of registered restaurants and 5-star establishments on Yelp.

(3) What is the relationship between a restaurant's star rating and the number of days it operates?

First, generate the number of operating days:

The code below calculates the number of operating days for each restaurant per week:

```
-- Calculate the number of open days for each business.

CREATE TABLE open_days as
(SELECT
   business_id,
   (
      (CASE WHEN monday = 'None' THEN 0 ELSE 1 END) +
      (CASE WHEN tuesday = 'None' THEN 0 ELSE 1 END) +
      (CASE WHEN wednesday = 'None' THEN 0 ELSE 1 END) +
      (CASE WHEN thursday = 'None' THEN 0 ELSE 1 END) +
      (CASE WHEN friday = 'None' THEN 0 ELSE 1 END) +
      (CASE WHEN saturday = 'None' THEN 0 ELSE 1 END) +
      (CASE WHEN sunday = 'None' THEN 0 ELSE 1 END)
   ) AS open_days
FROM
   yelp_business_hours);
SELECT * FROM open_days;
```

Here is an excerpt of the results saved as the table open_days:

business_id	open_days
--6MefnULPED_I942VcFNA	7
--7zmmkVg-IMGaXbuVd0SQ	7
--8LPVSo5i0Oo61X01sV9A	5
--9e1ONYQuAa-CB_Rrw7Tw	7
--9QQLMTbFzLJ_oT-ON3Xw	0
--ab39IjZR_xUf81WyTyHg	7
--cgVkbWTiga3OYTkymKqA	5
--cjBEbXMI2obtaRHNSFrA	4
--cZ6Hhc9F7VkKXxHMVZSQ	7
--DaPTJW3-tB1vP-PfdTEg	7
--DdmeR16TRb3LsjG0ejrQ	3
--e8PjCNhEz32pprnPhCwQ	7
--EF5N7P70J_UYBTPypYlA	7
--EX4rRznJrltyn-34Jz1w	0
--FBCX-N37CMYDfs790Bnw	0
--FLdgM0GNpXVMn74ppCGw	5
--g-a85VwrdZJNf0R95GcQ	7
--g8DrU2SDtAH615TFC0dQ	6
--GM_ORV2cYS-h38DSaCLw	7
--i1tTcggBi4cPkd-h5hDg	7
--I7YYLada0tSLkORTHb5Q	7
--j-kaNMCo1-DYzddCsA5Q	6
--KCl2FvVQpvjzmZSPyviA	7
--kinfHwmtdjz03g8B8z8Q	5
--KQsXc-clkO7oHRqGzSzg	0
--lpHMVmkCuji0ZrpHtXEA	5
--LY7PrnEegglB7vnPCjQw	7
--Ni3oJ4VOqfOEu7Sj2Vzg	7

Table 44 Excerpt of the number of operating days for restaurants per week

Next, use open_days to analyze the relationship between star ratings and the number of operating days:

```
-- Calculate the count of businesses with different star ratings
-- based on the number of open days.
SELECT
    open_days.open_days,
    SUM(CASE yelp_business.stars WHEN 5.0 THEN 1 ELSE 0 END) as sum_5,
    SUM(CASE yelp_business.stars WHEN 4.5 THEN 1 ELSE 0 END) as sum_4_5,
    SUM(CASE yelp_business.stars WHEN 4.0 THEN 1 ELSE 0 END) as sum_4,
    SUM(CASE yelp_business.stars WHEN 3.5 THEN 1 ELSE 0 END) as sum_3_5,
    SUM(CASE yelp_business.stars WHEN 3.0 THEN 1 ELSE 0 END) as sum_3,
    SUM(CASE yelp_business.stars WHEN 2.5 THEN 1 ELSE 0 END) as sum_2_5,
    SUM(CASE yelp_business.stars WHEN 2.0 THEN 1 ELSE 0 END) as sum_2,
    SUM(CASE yelp_business.stars WHEN 1.5 THEN 1 ELSE 0 END) as sum_1_5,
    SUM(CASE yelp_business.stars WHEN 1.0 THEN 1 ELSE 0 END) as sum_1
FROM
    yelp_business
LEFT JOIN
    open_days
ON
    yelp_business.business_id = open_days.business_id
GROUP BY
    open_days.open_days;
```

The results show the number of operating days per week and the corresponding number of restaurants for each star rating:

open_days	sum_5	sum_4_5	sum_4	sum_3_5	sum_3	sum_2_5	sum_2	sum_1_5	sum_1
0	4405	4688	7039	8162	7234	5927	3888	2020	1944
1	73	69	62	47	34	13	10	4	10
2	64	54	59	61	37	25	15	4	6
3	193	90	80	85	53	38	15	9	8
4	801	392	303	207	137	70	42	13	19
5	6108	3338	3143	2640	1418	1272	691	343	414
6	6676	5709	6139	4671	2652	1839	1023	460	529
7	9220	10456	16667	16165	11577	6964	3636	1450	858

Table 45 Number of restaurants by weekly operating days and star rating

Using the number of operating days per week on the horizontal axis and the number of restaurants on the vertical axis, a chart is created showing the distribution of operating days for each star rating:

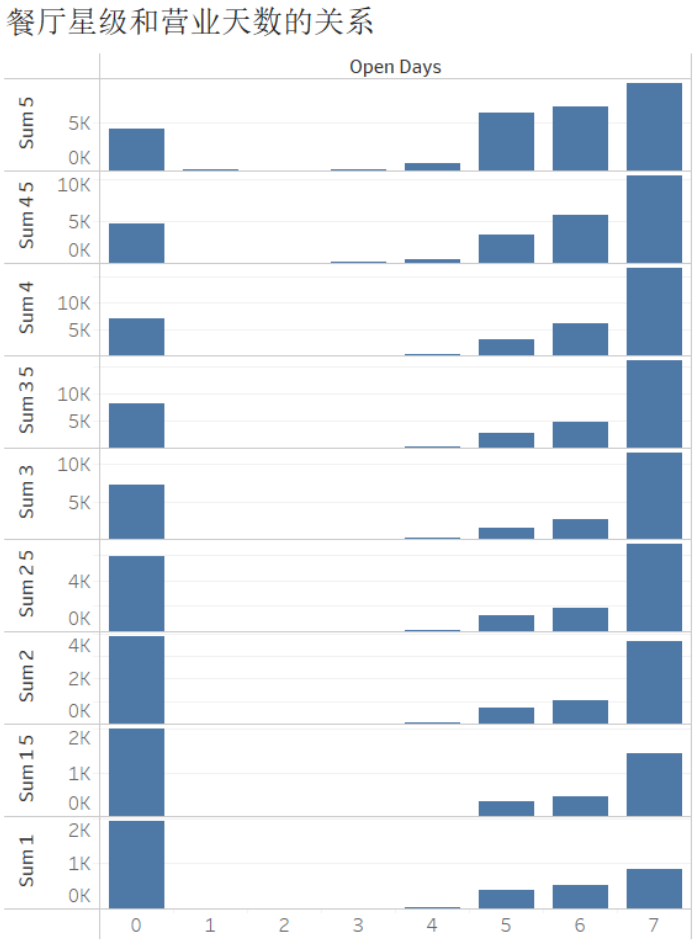

Table 46 Relationship between restaurant star ratings and operating days

The data with 0 operating days includes restaurants with no available data and can be ignored. From the chart, it can be observed that 5-star restaurants often operate 5, 6, or 7 days per week. In contrast, for 3- to 5-star restaurants, the proportion of those operating 5 or 6 days per week decreases as the star rating decreases, while a significant number of restaurants

operate 7 days per week. This suggests that a restaurant's star rating is not solely dependent on the number of days it operates. While maintaining a certain number of operating days (at least 5), other factors play a more important role in achieving a higher rating. For 1- to 2-star restaurants, the proportion operating 5 or 6 days per week increases compared to 3- to 4-star restaurants, indicating that for average restaurants, consistent operation can also contribute to maintaining a stable reputation.

(4) What are the characteristics of closed restaurants in terms of operating days and star ratings?

The average weekly operating days and star ratings for closed restaurants are calculated using the following code:

```
-- Calculate the average number of open days and the average star rating
-- for businesses that are not open (is_open = 0).
SELECT
   ROUND(AVG(open_days.open_days), 1) as avg_open,
   ROUND(AVG(yelp_business.stars), 2) as avg_stars
FROM
   yelp_business
LEFT JOIN
   open_days
ON
   yelp_business.business_id = open_days.business_id
WHERE
   yelp_business.is_open = 0;
```

The results show that, on average, closed restaurants operated about 4 days per week with an average star rating of approximately 3.5:

avg_open	avg_stars
3.9	3.51

Table 47 Average weekly operating days and star ratings for closed restaurants

Comparing these results with the data for all restaurants, both metrics are slightly lower, suggesting that there is a slight correlation between restaurant closure, operating days, and average star rating, with all three factors influencing one another.

3. From the Platform Perspective:

(1) Number of users registered each year and month? Number of user reviews?

First, the number of users registered each year is calculated using the following code:

```
SELECT user_id, YEAR(yelping_since) as year_since, count(user_id)
FROM yelp_user
GROUP BY YEAR(yelping_since)
ORDER BY YEAR(yelping_since) ASC
```

The results are as follows:

user_id	year_since	count(user_id)
-vlsy_TqgkxQ7XOC8T48fA	2004	75
-0Nvk7jIo79LaxChQtDyLA	2005	979
--KQJPdrU0Md97DiOliDzv	2006	5951
---1lKK3aKOuomHnwAkA	2007	16364
--4q8EyqThydQm-eKZpS-	2008	32433
---cu1hq55BP9DWVXXKH	2009	60905
--0kuuLmuYBe3Rmu0lycw	2010	99785
---fhiwiwBYrvqhpXgcWDC	2011	152353
--2vR0DIsmQ6WfcSzKWig	2012	161880
--0RtXvcOIE4XbErYca6Rw	2013	175483
---udAKDsn0yQXmzbWQI	2014	198976
---PLwSf5gKdIoVnyRHgBA	2015	196149
---94vtJ_5o_nikEs6hUjg	2016	147593
--4ww39MLTS1SBRmCrSr	2017	77174

Table 48 Number of users registered each year

The histogram is plotted as follows:

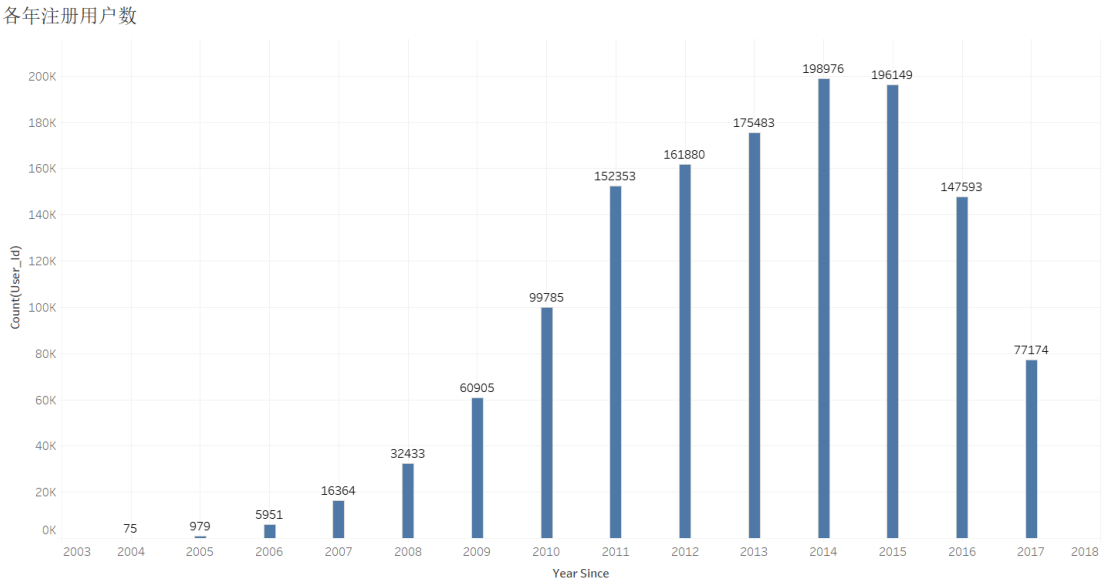

Table 49 Histogram of the number of users registered each year

It can be seen that the number of registered users increased yearly before 2014 and then declined after 2014. The platform saw the highest growth in users in 2014, with 198,976 new users. The growth in 2015 was also substantial, reaching 196,149 new users. However, the decline became more noticeable after 2015. Although the 2017 data only extends up to December 11, as mentioned in the first section of this chapter, it is unlikely that there would be a sudden surge in new users during the last 20 days of the year, so it can be inferred that the total number of registrations continued to decline.

Next, the number of users registered each year and month is calculated using the following

code:

```
-- Retrieve the count of users who joined Yelp in each year and month
-- along with their user IDs, the year they joined, and the month they joined.
SELECT
   user_id,
   YEAR(yelping_since) as year_since,
   MONTH(yelping_since) as month_since,
   COUNT(user_id)
FROM
   yelp_user
GROUP BY
   YEAR(yelping_since),
   MONTH(yelping_since);
```

The results are as follows:

user_id	year_since	month_since	count(user_id)
0sidDfoTIHle5vvHEBvF0w	2004	10	48
anP7rMS6XZkabmtjy4UJy	2004	11	6
-vlsy_TqgkxQ7XOC8T48fA	2004	12	21
0GMMy-FD8ImVsF4M6CA	2005	1	25
-MQPWNRr3O4PVD2qBul	2005	2	28
-M7-eB1J7k9avBrYWSaBi(	2005	3	67
-ydCYfBsk4BpRhnlDeD2fC	2005	4	76
-0Nvk7jIo79LaxChQtDyLA	2005	5	69
0V4Ra6jCmSAgFtCiORr4C	2005	6	59
-a9IN_IZSLphqDCq5r1DU\	2005	7	95
-3i9bhfvrM3F1wsC9XIB8g	2005	8	117
-4R8Jo83-4-mnUvqErtfoC	2005	9	90
0Lq3mADXrj7HRiEI7JRdO	2005	10	82
-deQ6hSaz3b_mDR888-N	2005	11	94
-34NV3A5a5gcL-s8UH4Kl	2005	12	177
--KQJPdrU0Md97DiOliDz\	2006	1	271

Table 50 Number of users registered each year and month

The trend of new user registrations for each year and month is shown below:

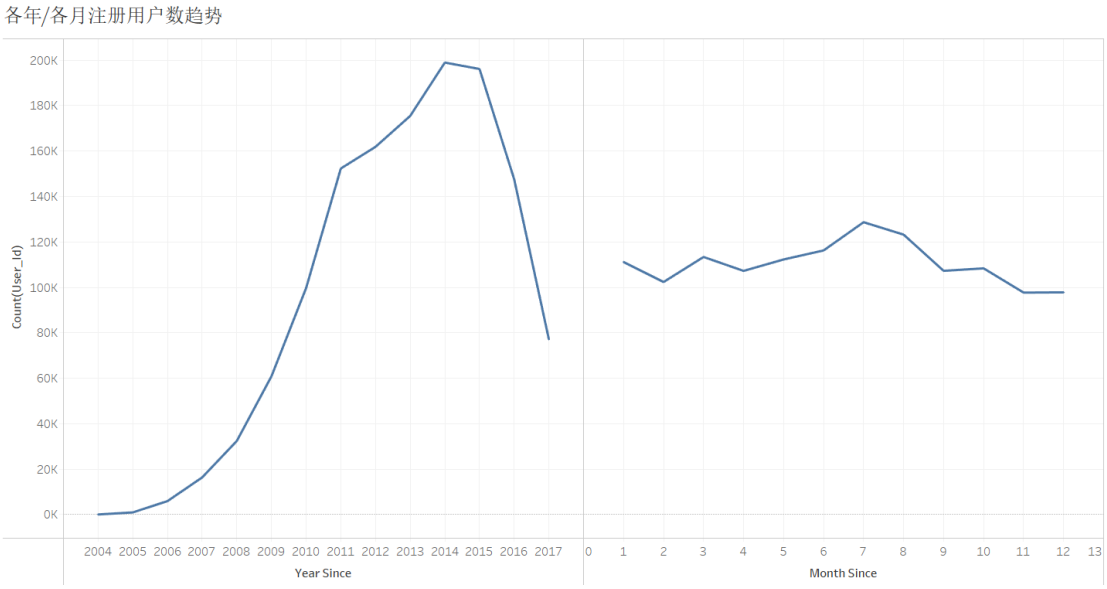

Table 51 Trend of the number of users registered each year and month

The analysis of annual new user growth has been discussed above and will not be repeated here. Regarding monthly growth, it can be seen that around July is a peak period, likely influenced by the summer heat and the concentration of holidays. During this time, people's desire and frequency to dine out increase, leading to greater demand and usage of this app.

Next, the total number of reviews posted by users each year on the platform is calculated using the following code:

```
SELECT YEAR(date) as year_review, count(review_id)
FROM yelp_review
GROUP BY YEAR(date)
ORDER BY YEAR(date) ASC
```

The results are as follows:

year_review	count(review_id)
2004	14
2005	870
2006	5669
2007	23020
2008	61553
2009	98288
2010	187073
2011	290933
2012	350381
2013	472595
2014	678351
2015	911487
2016	1052916
2017	1128518

Table 52 Total number of reviews posted by users each year on the platform

A histogram is plotted with the year on the horizontal axis and the total number of reviews for that year on the vertical axis, along with a trend line:

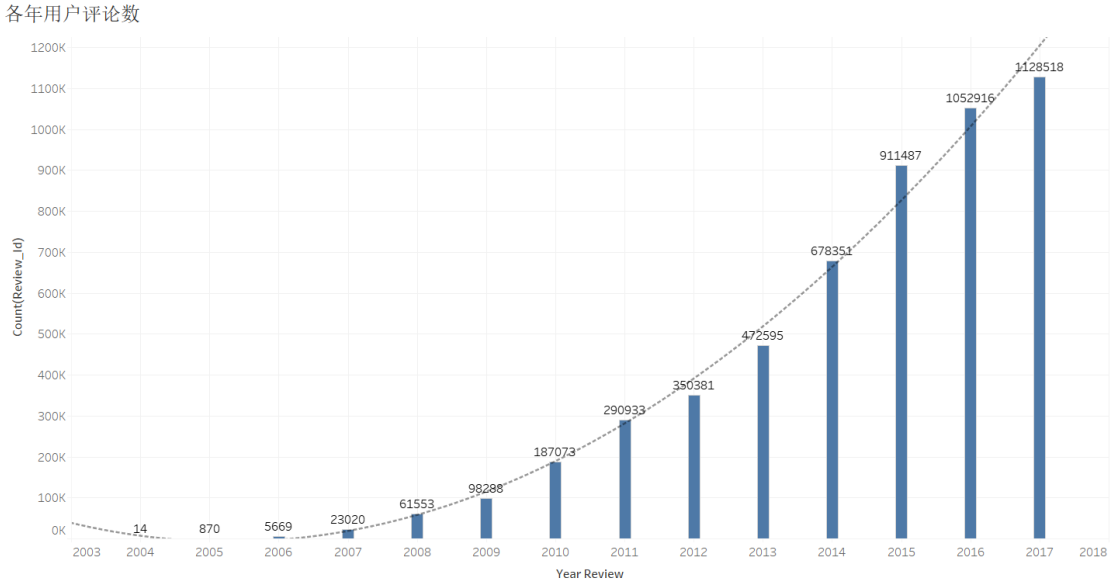

Table 53 Histogram of the total number of reviews posted by users each year

Here, a power function (α=2) is used for fitting, which closely aligns with the trend of increasing yearly reviews. It is clear that the number of reviews posted by users on the platform is steadily rising, and the growth rate has not decreased over time. This indicates that user engagement with the platform remains strong.

(2) The number and proportion of elite users each year?

First, filter out the usernames of those who have been selected as elite users in at least one year:

```
SELECT user_id, elite
FROM yelp_user
WHERE elite <> 'None'
```

The results are stored in a table named elite:

user_id	elite
---1lKK3aKOuomHnwAkAow	2010, 2013, 2011, 2012
--2vR0DIsmQ6WfcSzKWigw	2014, 2017, 2015, 2016, 2013
--3l8wysfp49Z2TLnyT0vg	2016
--3WaS23LcIXtxyFULJHTA	2016, 2013, 2014, 2017, 2015, 2012
--41c9Tl0C9OGewIR7Qyzg	2016, 2015, 2012, 2014, 2013
--4q8EyqThydQm-eKZpS-A	2010, 2013, 2011, 2012
--4uW4yJiRT2oXMYkCPq1Q	2017
--56mD0sm1eOogphi2FFLw	2015, 2014, 2016, 2017, 2013
--A4pFATzQJx9n4l1IAC3A	2017
--BumyUHiO_7YsHurb9Hkw	2017
--cPqjzKHqHKmGala65zwg	2016, 2015, 2017, 2014
--kedvpjB1PT28X_gArafA	2012, 2011, 2010
--KQJPdrU0Md97DiOliDzw	2010, 2009, 2011
--mQ4S5h1tXzvE9VDYVwdQ	2016, 2017, 2015
--Qh8yKWAvIP4V4K8ZPfHA	2012, 2014, 2013, 2016, 2015, 2017
--u09WAjW741FdfkJXxNmg	2017

Table 54 Elite User Selection Table (elite)

The number of elite users for each year is calculated from the elite table using the following code:

```
-- Calculate the count of users with elite status in each year from 2004 to 2017.
SELECT
   SUM(CASE WHEN elite LIKE '%2004%' THEN 1 ELSE 0 END) as 2004_elite,
   SUM(CASE WHEN elite LIKE '%2005%' THEN 1 ELSE 0 END) as 2005_elite,
   SUM(CASE WHEN elite LIKE '%2006%' THEN 1 ELSE 0 END) as 2006_elite,
   SUM(CASE WHEN elite LIKE '%2007%' THEN 1 ELSE 0 END) as 2007_elite,
   SUM(CASE WHEN elite LIKE '%2008%' THEN 1 ELSE 0 END) as 2008_elite,
   SUM(CASE WHEN elite LIKE '%2009%' THEN 1 ELSE 0 END) as 2009_elite,
   SUM(CASE WHEN elite LIKE '%2010%' THEN 1 ELSE 0 END) as 2010_elite,
   SUM(CASE WHEN elite LIKE '%2011%' THEN 1 ELSE 0 END) as 2011_elite,
   SUM(CASE WHEN elite LIKE '%2012%' THEN 1 ELSE 0 END) as 2012_elite,
   SUM(CASE WHEN elite LIKE '%2013%' THEN 1 ELSE 0 END) as 2013_elite,
   SUM(CASE WHEN elite LIKE '%2014%' THEN 1 ELSE 0 END) as 2014_elite,
   SUM(CASE WHEN elite LIKE '%2015%' THEN 1 ELSE 0 END) as 2015_elite,
   SUM(CASE WHEN elite LIKE '%2016%' THEN 1 ELSE 0 END) as 2016_elite,
   SUM(CASE WHEN elite LIKE '%2017%' THEN 1 ELSE 0 END) as 2017_elite
FROM
   elite;
```

The results are as follows:

2004_elite	2005_elite	2006_elite	2007_elite	2008_elite	2009_elite
0	140	887	2363	3621	6536

Table 55 Number of elite users from 2004 to 2009

2010_elite	2011_elite	2012_elite	2013_elite	2014_elite	2015_elite	2016_elite	2017_elite
10485	13185	17777	19841	20488	26018	30856	34928

Table 56 Number of elite users from 2010 to 2017

It is evident that the selection of elite users began in 2005. The number of users selected as

elite users has increased each year, but since 2008, the growth rate has not seen significant changes, fluctuating between 2,000 and 6,000, mostly around 4,000, which is much smaller than the variation in new users.

(3) What is the annual retention rate of users? What is the annual retention rate for elite users?

For all users, the year in which they left a review is used as the basis for determining whether they are retained. The number of reviews by each user per year is calculated using the following code:

```
-- Calculate the count of reviews posted by each user in each year from 2008 to 2017.
SELECT
    user_id,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2008 THEN 1 ELSE 0 END) as 2008_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2009 THEN 1 ELSE 0 END) as 2009_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2010 THEN 1 ELSE 0 END) as 2010_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2011 THEN 1 ELSE 0 END) as 2011_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2012 THEN 1 ELSE 0 END) as 2012_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2013 THEN 1 ELSE 0 END) as 2013_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2014 THEN 1 ELSE 0 END) as 2014_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2015 THEN 1 ELSE 0 END) as 2015_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2016 THEN 1 ELSE 0 END) as 2016_exist,
    SUM(CASE WHEN YEAR(yelp_review.date) = 2017 THEN 1 ELSE 0 END) as 2017_exist
FROM
    yelp_review
GROUP BY
    user_id;
```

Here is an excerpt of the results saved as the user_keep table:

user_id	2008_exist	2009_exist	2010_exist	2011_exist	2012_exist	2013_exist	2014_exist	2015_exist	2016_exist	2017_exist
---1lKK3aKOuomHnwAkAow	1	0	38	41	22	5	1	6	3	2
---94vtJ_5o_nikEs6hUjg	0	0	0	0	0	0	0	0	1	0
---cu1hq55BP9DWVXXKHZg	0	0	0	0	0	0	3	0	0	0
---fhiwiwBYrvqhpXgcWDQ	0	0	0	0	0	1	0	0	0	0
---PLwSf5gKdIoVnyRHgBA	0	0	0	0	0	0	0	0	1	1
---udAKDsn0yQXmzbWQNSw	0	0	0	0	0	0	0	0	0	2
--0kuuLmuYBe3Rmu0lycww	0	0	0	0	0	6	6	0	0	0
--0RtXvcOIE4XbErYca6Rw	0	0	0	0	0	1	0	0	0	0
--0sXNBv6lizZXuV-nI0Aw	0	0	0	0	0	0	0	0	1	0
--0WZ5gklOfbUlodJuKfaQ	0	0	0	0	0	0	0	0	1	0
--104qdWvE99vaoIsj9ZJQ	0	0	0	0	0	0	0	0	3	0
--1av6NdbEbMiuBr7Aup9A	0	0	0	0	0	0	0	0	4	0
--1mPJZdSY9KluaBYAGboQ	0	0	0	0	0	4	1	0	0	0
--26jc8nCJBy4-7r3ZtmiQ	0	0	0	0	0	0	1	0	0	0

Table 57 user_keep table with user IDs and review years

The retention rate is calculated as the proportion of users who posted reviews in a given year and continued to post reviews in the following year:

```
-- Calculate the proportion of users who posted reviews in consecutive years for each pair of years from 2008 to
2017.
SELECT
    ROUND(SUM(CASE WHEN (2008_exist <> 0 AND 2009_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2008_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2009,
    ROUND(SUM(CASE WHEN (2009_exist <> 0 AND 2010_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2009_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2010,
    ROUND(SUM(CASE WHEN (2010_exist <> 0 AND 2011_exist <> 0) THEN 1 ELSE 0 END) /
```

```
SUM(CASE WHEN 2010_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2011,
   ROUND(SUM(CASE WHEN (2011_exist <> 0 AND 2012_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2011_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2012,
   ROUND(SUM(CASE WHEN (2012_exist <> 0 AND 2013_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2012_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2013,
   ROUND(SUM(CASE WHEN (2013_exist <> 0 AND 2014_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2013_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2014,
   ROUND(SUM(CASE WHEN (2014_exist <> 0 AND 2015_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2014_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2015,
   ROUND(SUM(CASE WHEN (2015_exist <> 0 AND 2016_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2015_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2016,
   ROUND(SUM(CASE WHEN (2016_exist <> 0 AND 2017_exist <> 0) THEN 1 ELSE 0 END) /
SUM(CASE WHEN 2016_exist <> 0 THEN 1 ELSE 0 END), 2) AS exist_2017
FROM
   user_keep;
```

The results are as follows:

exist_2009	exist_2010	exist_2011	exist_2012	exist_2013	exist_2014	exist_2015	exist_2016	exist_2017
0.30	0.31	0.31	0.28	0.30	0.33	0.33	0.32	0.32

Table 58 Annual retention rate of all users from 2009 to 2017

It can be observed that the annual retention rate of regular users has remained around 30% over the past decade.

For the annual retention rate of elite users, it is calculated by determining whether elite users in one year retained their elite status in the following year:

```
-- Calculate the proportion of users who had elite status in consecutive years for each pair of years from 2005 to
2017.
SELECT
   ROUND(SUM(CASE WHEN (elite LIKE '%2005%' AND elite LIKE '%2006%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2005%' THEN 1 ELSE 0 END), 2) AS exist_2006,
   ROUND(SUM(CASE WHEN (elite LIKE '%2006%' AND elite LIKE '%2007%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2006%' THEN 1 ELSE 0 END), 2) AS exist_2007,
   ROUND(SUM(CASE WHEN (elite LIKE '%2007%' AND elite LIKE '%2008%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2007%' THEN 1 ELSE 0 END), 2) AS exist_2008,
   ROUND(SUM(CASE WHEN (elite LIKE '%2008%' AND elite LIKE '%2009%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2008%' THEN 1 ELSE 0 END), 2) AS exist_2009,
   ROUND(SUM(CASE WHEN (elite LIKE '%2009%' AND elite LIKE '%2010%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2009%' THEN 1 ELSE 0 END), 2) AS exist_2010,
   ROUND(SUM(CASE WHEN (elite LIKE '%2010%' AND elite LIKE '%2011%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2010%' THEN 1 ELSE 0 END), 2) AS exist_2011,
   ROUND(SUM(CASE WHEN (elite LIKE '%2011%' AND elite LIKE '%2012%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2011%' THEN 1 ELSE 0 END), 2) AS exist_2012,
   ROUND(SUM(CASE WHEN (elite LIKE '%2012%' AND elite LIKE '%2013%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2012%' THEN 1 ELSE 0 END), 2) AS exist_2013,
   ROUND(SUM(CASE WHEN (elite LIKE '%2013%' AND elite LIKE '%2014%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2013%' THEN 1 ELSE 0 END), 2) AS exist_2014,
   ROUND(SUM(CASE WHEN (elite LIKE '%2014%' AND elite LIKE '%2015%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2014%' THEN 1 ELSE 0 END), 2) AS exist_2015,
   ROUND(SUM(CASE WHEN (elite LIKE '%2015%' AND elite LIKE '%2016%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2015%' THEN 1 ELSE 0 END), 2) AS exist_2016,
   ROUND(SUM(CASE WHEN (elite LIKE '%2016%' AND elite LIKE '%2017%') THEN 1 ELSE 0 END) /
SUM(CASE WHEN elite LIKE '%2016%' THEN 1 ELSE 0 END), 2) AS exist_2017
FROM
   elite;
```

The results are as follows:

exist_2006	exist_2007	exist_2008	exist_2009	exist_2010	exist_2011	exist_2012	exist_2013	exist_2014	exist_2015	exist_2016	exist_2017
0.94	0.89	0.72	0.82	0.83	0.80	0.80	0.85	0.77	0.82	0.84	0.82

Table 59 Annual retention rate of elite users

From 2006 onwards, the year after the platform started selecting elite users, the annual retention rate for elite users has remained around 80%, which is quite high and significantly higher than the retention rate of all users.

Conclusion

1. From the Users Perspective

For users, the most important goal might be figuring out how to quickly and efficiently earn the title of elite user to enjoy more promotions and benefits brought by increased traffic on the platform. According to the analysis, users should focus on posting more reviews and photos, especially reviews, as the quantity is a significant indicator. For each review, users should not only focus on writing lengthy reviews but also on enhancing the informational content within them. Rather than merely polishing the writing style, the review should provide valuable information to the public. Reviews that point out issues in highly-rated restaurants, in particular, are more likely to attract attention from other users. Residing in states like Nevada or Arizona could also increase the chances of being selected as an elite user, as these states have a higher concentration of high-traffic businesses.

For regular users who are not concerned with achieving elite status, their main goal in using Yelp is to find restaurants that are delicious and meet their needs. Generally speaking, restaurant star ratings and user ratings are consistent and can reasonably reflect the quality of a restaurant, allowing users to make selections based on their preferences. It’s worth noting that 5-star restaurants may have inflated ratings, so users should be discerning based on feedback.

2. From the Merchants Perspective

From the merchant's perspective, to attract more customers, they could consider opening branches in larger states like Nevada, Arizona, North Carolina, Ohio, and Pennsylvania. To upgrade to a 5-star business, special attention should be paid to Arizona and Nevada. Besides ensuring basic factors like high foot traffic and sufficient operating days, 5-star restaurants also need to focus on other criteria for improvement; lowering the rate of negative reviews is

particularly important. For restaurants that want to avoid low ratings and maintain high ratings, the choice of geographic location is less influential, but it's best to operate more days per week, while also ensuring a high rate of positive reviews.

3. From the Platform Perspective

The platform still has considerable appeal to existing users, as usage is strong. However, the registration of new users has been declining year by year, necessitating some measures, such as collaborating with restaurants during popular outdoor seasons like summer to promote the app or offering discounts to attract new users. The decline in registration numbers may also signal a saturated market. Expanding business to more states or even other countries could help the platform achieve further growth.

For existing users, especially active regular users, the platform could periodically launch activities that allow users to earn rewards by increasing their engagement on the platform or offer incentives to returning users. Additionally, the platform could further enhance the benefits for elite users to attract more users to strive for elite status.