ECE 356 Lab2 Yelp

Lab section 206

Group 10

Victor Yan

Lun Jing

(a) Which user has written the greatest number of reviews?

EXPLAIN

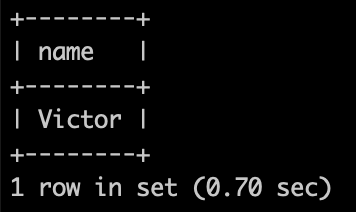
SELECT name FROM user

ORDER BY review\_count DESC

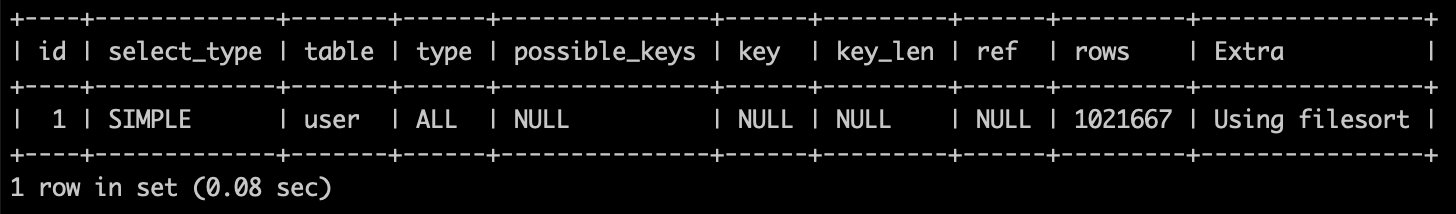
LIMIT 1;

Before adding indexes:

Running time:



Explain:



Since this query is trying to sort the review\_count, it will be helpful to add the review\_count in the index.

CREATE INDEX user\_1 on user (review\_count);

(b) Which business has received the greatest number of reviews?

EXPLAIN

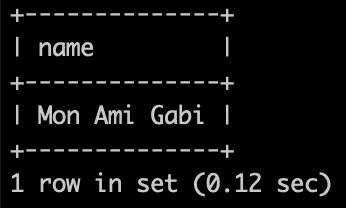
SELECT name FROM business

ORDER BY review\_count DESC

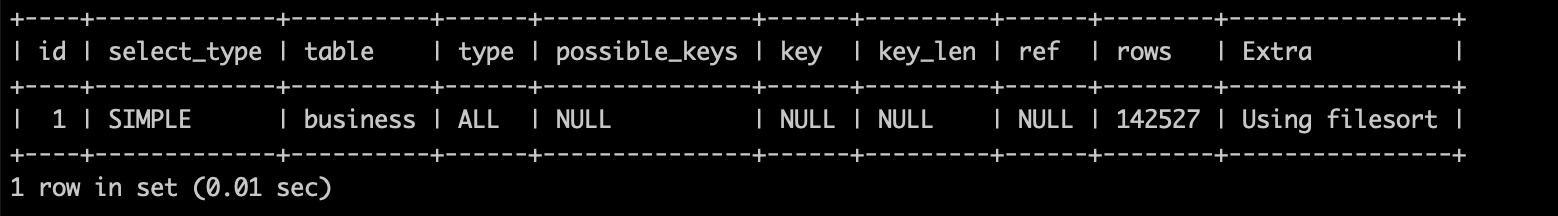
LIMIT 1;

Before adding indexes:

Running time:



Explain:



Similar as previous one, we also add review\_count into the index.

CREATE INDEX business\_1 on business (review\_count);

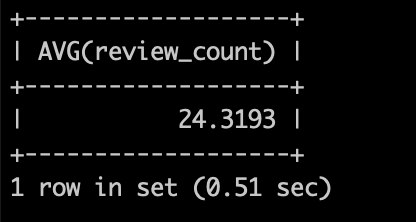
(c) What is the average number of reviews written by users?

EXPLAIN

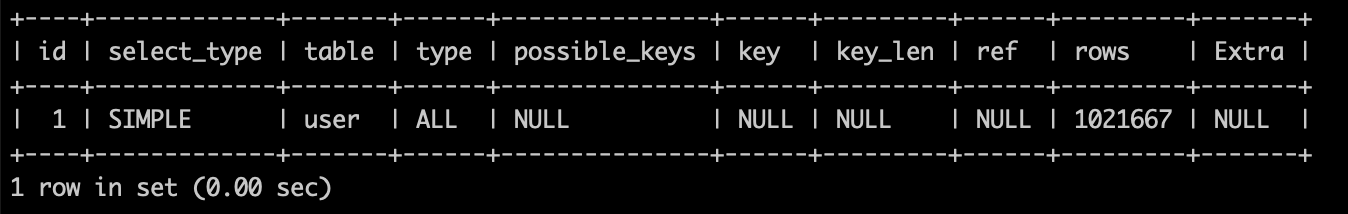
SELECT AVG(review\_count) FROM user;

Before adding indexes:

Running time:



Explain:



Since this query is aggregation, which always need to go through all input once, there is no need to add index for it.

(d) The average rating written by a user can be determined in two ways:

a. By direct reading from the Users table “average stars” column

b. By computing an average of the ratings issued by a user for businesses reviewed

For how many users is the difference between these two amounts larger than 0.5?

EXPLAIN

SELECT COUNT(\*) FROM

(SELECT user\_id, average\_stars FROM user) as A inner join

(SELECT AVG(stars) as avg\_stars, user\_id FROM review

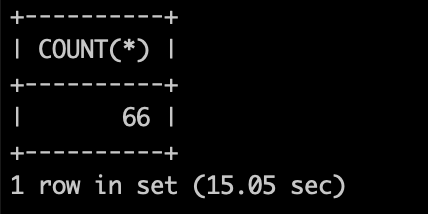
GROUP BY user\_id ) as B

USING (user\_id)

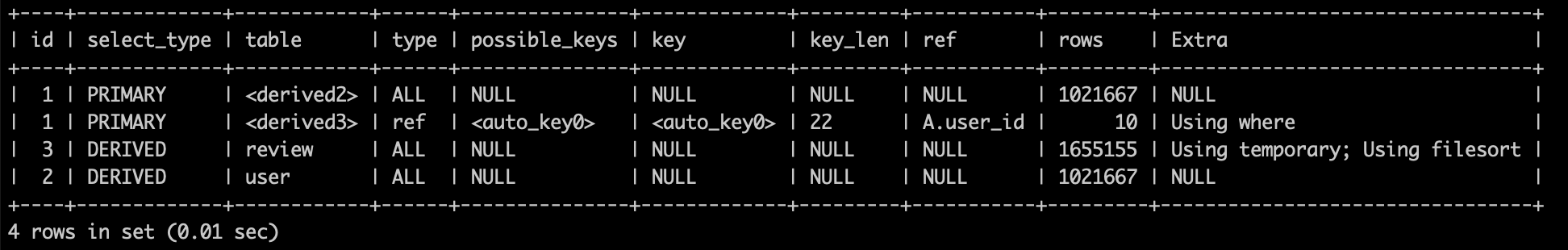
WHERE ABS(A.average\_stars - B.avg\_stars) > 0.5;

Before adding indexes:

Running time:



Explain:



Since this query joins two tables USING (user\_id), we can add indexes with user\_id for each of them.

CREATE INDEX user\_2 on user (user\_id);

CREATE INDEX review\_1 on review (user\_id);

(e) What fraction of users have written more than 10 reviews?

EXPLAIN

SELECT

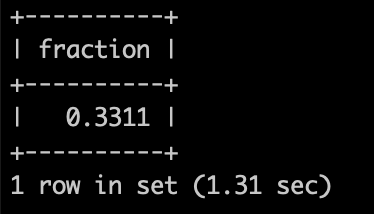
(SELECT count(\*) FROM user Where review\_count > 10)

/

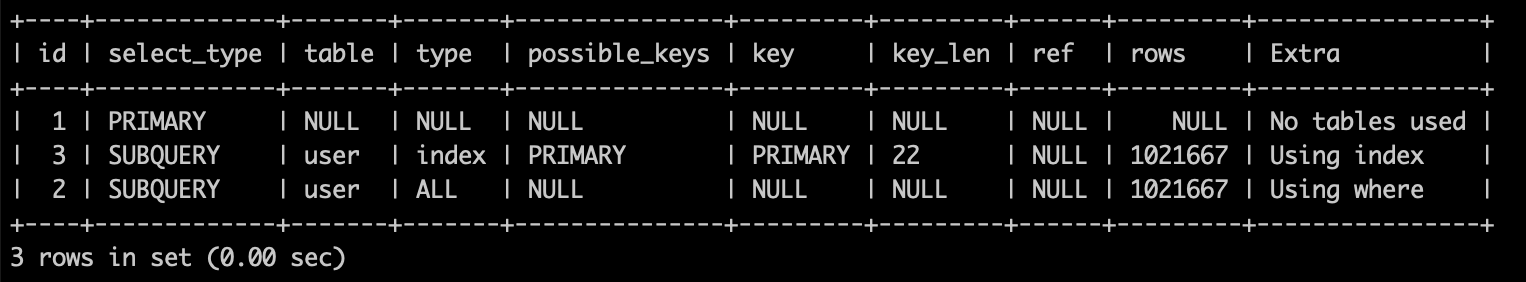
(SELECT count( distinct user\_id) FROM user) as fraction;

Before adding indexes:

Running time:



Explain:



Similar as previous one, but this time we need count the number of entries that have review\_count > 10 seperately. So we should create a seperate index for review\_count only.

CREATE INDEX user\_3 on user (review\_count);

(f) What is the average length of their reviews?

EXPLAIN

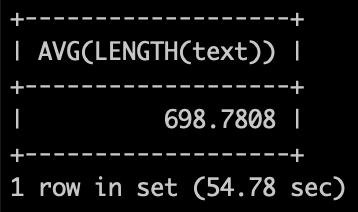
SELECT AVG(LENGTH(text)) FROM user as U inner join review as R

USING (user\_id)

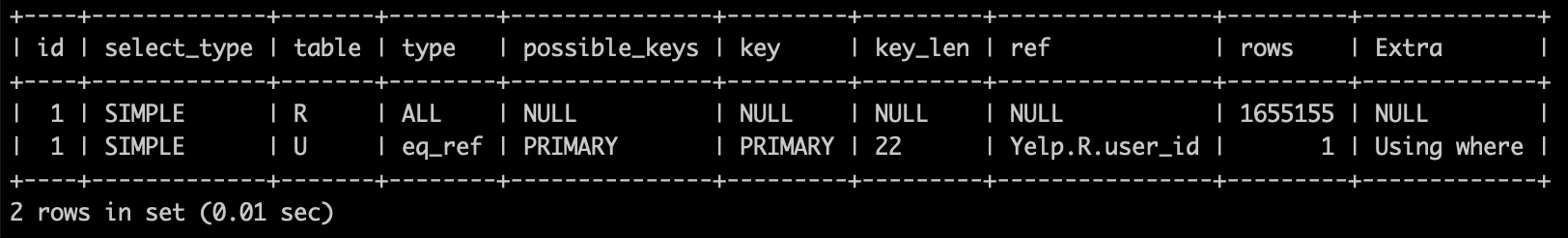
WHERE U.review\_count > 10;

Before adding indexes:

Running time:



Explain:



Since all the text will be accessed at least once, we cannot add any index for it. However, we will need to check review table and select the entries with review\_count > 10 and then join the user\_id, we can use the index with review\_count and user\_id

CREATE INDEX user\_4 on user (review\_count, user\_id);