Homework Exercises:

1.

- select b.state, b.city, b.review_count from business b order by b.review_count desc;

- Extra Credit:
    - select b.city, avg(b.stars) from business b group by b.city order by average_review desc;

2.

- Step 1:
    - select b.name, b.stars from business b where b.city = ‘Toronto’;
- Step 2:
    - select b.name, avg(r.stars) as actual_average_review from business b inner join review r on r.business_id = b.id where b.city = ‘Toronto’ group by b.name order by actual_average_review desc;
- Step 3:
    - select b.name, avg(r.stars) from business b left join review r on r.business_id = b.id where b.city = ‘Toronto’ group by b.name;

3.

- select b.name from business b inner join review r on b.id = r.business_id where r.text is not null order by b.review_count;

4.

- Part I:
    - select text from review order by date asc limit 1;

      (Execution time: 21.471 ms)

      - OR
    - select text from review where date = (select min(date) from review);

      (Execution time: 23.376 ms)

        - Note: Implemented the query in two ways but strange to have first one have less execution time than the other one. Whereas, same implementation is the opposite in Part II queries.
- Part II:
    - select r.text from review r inner join “user” u on u.id = r.user_id order by yelping_since asc limit 1;

      (Execution time: 455.599 ms)

      - OR
    - select r.text from review r inner join “user” u on u.id = r.user_id where yelping_since = (select min(yelping_since) from “user”);

      (Execution time: 297.891 ms)

        - Note: - ‘user’ is a SQL reversed word so need to provide “ ” over user to reference to the table.

5.
- select b.name from business b inner join category c on c.business_id = b.id inner join review r on r.business_id = b.id where city = ‘Phoenix’ and c.category = ‘Nightlife’ and r.text is not null order by b.stars asc limit 1;
    - Note: To compare two equal String values: need to use single quotation (‘ ’). E.g: city = ‘Phoenix’.

6.
- Part I:
    - select sum(review_count) from business where is_open = 0;
- Part II:
    - select stars, count(*) from business where is_open = 0 group by stars order by stars asc;

7.
- select t.rounded_stars, count(rounded_stars) from (select round(u.average_stars) as rounded_stars from “user” u) as t group by t.rounded_stars;
    - Note: Implemented the query by aliasing a new table t. There might be other optimized ways to implement the query though. Curious: how to optimize certain query?

8.
- select count(neighborhood) from business where city = ‘Toronto’;

9.
- select stars, count(stars) from (select b.neighborhood, r.stars from business b inner join review r on b.id = r.business_id where b.city = ‘Toronto’) as t group by t.stars order by t.stars asc;
    - Note: I guess I understood the question: if so I also aliased new table accordance to the purpose of query.

10.
- select c.category, count(b.name) as totatl_count from business b inner join category c on c.business_id = b.id group by c.category order by total_count desc limit 25;

11. Extra Credit:
- select b.name from business b inner join attribute a on a.business_id = b.id where b.city = ‘Pittsburgh’ and a.name = ‘Ambience’ and a.value like ‘%__hipster____true%’ order by b.stars desc limit 10;
    - Note:
        - Roadblocks: - pattern matching JSON formatted TEXT type (was not even quite in JSON format to CAST that TEXT to JSON in order to match the pattern). However, figured out to use LIKE as well as Underscore at once to match pattern ignoring \ and “ symbols.