1. [10 pts] Create an SQL query that finds customer id, customer email, flight_number, projected_departure_datetime, and the quantity of tickets for each reservation made by customers with an email starting with letter '**i**'.

```
SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and C.email like 'i%';
```

2. [10 pts] Create the same query to return the same attributes. The only difference is that the filtering condition is "email ending with **d.com**" instead of "starting with letter **i**".

```
SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and C.email like '%d.com';
```

3. [10 pts] Create the same query as question 1 to return the same attributes. The only change that you need to make is the filtering condition is the projected_departure_datetime is on or before '2015-07-01 00:00:00'.

```
SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and CRF.projected_departure_datetime <= '2015-07-01 00:00:00';
```

4. [15 pts] We want to create indexes to speed up three queries above. To make the right decision on the indexes, first check the visual query plan for each query above by copying and pasting your SQLs and clicking the lightning + magnifier button in MySQLWorkBench like the following example. **You are not required to submit the visual query plan**.

   Now, submit the textual query plan for each of the three queries. To create a textual query plan, you can use the command "EXPLAIN" in front of each query to see the query plan in text (e.g., "EXPLAIN SELECT email from Customer ..."). After adding "EXPLAIN" to a query, click "Query" -> "Execute (All or Selection) to Text" to generate the output and paste the output as an answer.

   a. Query 1:

```
Execute:
> EXPLAIN SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and C.email like 'i%'

+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| id      | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| 1       | SIMPLE      | C     |            | ALL  | PRIMARY       |     |         |     | 20   | 11.11    | Using where |
```

| 1 | SIMPLE | CRF | | ref | PRIMARY | PRIMARY | 4 | cs122a.C.cid | 1 | 100.00 | |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
2 rows

Refer to https://dev.mysql.com/doc/refman/5.5/en/explain-output.html#explain-output-columns for more details. Type:ALL means that a full table scan is conducted.

b. Query 2)

Execute:

```
> EXPLAIN SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and C.email like '%d.com'
```

+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| 1 | SIMPLE | C | | **ALL** | **PRIMARY** | | | | 20 | 11.11 | Using where |
| 1 | SIMPLE | CRF | | ref | PRIMARY | PRIMARY | 4 | cs122a.C.cid | 1 | 100.00 | |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
2 rows

c. Query 3)

Execute:

```
> EXPLAIN SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and CRF.projected_departure_datetime <= '2015-07-01 00:00:00'
```

+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| 1 | SIMPLE | CRF | | **ALL** | **PRIMARY** | | | | 25 | 33.33 | Using where |
| 1 | SIMPLE | C | | eq_ref | PRIMARY | PRIMARY | 4 | cs122a.CRF.cid | 1 | 100.00 | |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
2 rows

5. [10 pts] Write and execute a “CREATE INDEX” statement that creates an index on the Customer.email attribute. The index name should be “ix_Customer_email”.
   CREATE INDEX ix_Customer_email on Customer(email);

6. [10 pts] Write and execute a “CREATE INDEX” statement that creates an index on the Customer_Reserves_Flight.projected_departure_datetime attribute. The index name should be “ix_CRF_projected_departure_datetime”.
   CREATE INDEX ix_CRF_projected_departure_datetime on Customer_Reserves_Flight(projected_departure_datetime);

7. [15 pts] Now you have created two indexes. Check the visual query plan again for each of the three queries. Again, you are ***not*** required to submit the visual query plans. However, you ***are*** required to submit the textual “EXPLAIN” output of your queries by following the same steps in Q4.
   a. Query 1)

Execute:

```
> EXPLAIN SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and C.email like 'i%'
```

```
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| id      | select_type | table     | partitions | type      | possible_keys | key      | key_len    | ref      | rows     | filtered   | Extra      |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
| 1       | SIMPLE      | C         |            | range     | PRIMARY,ix_Customer_email | ix_Customer_email | 93       |          | 3        | 100.00     | Using where; Using index |
| 1       | SIMPLE      | CRF       |            | ref       | PRIMARY       | PRIMARY  | 4          | cs122a.C.cid | 1        | 100.00     |            |
+ ------- + ---------- + ---------- + ---------- + --------- + ---------- + -------- + ---------- + -------- + --------- + ---------- + ---------- +
2 rows
```

Refer to https://dev.mysql.com/doc/refman/5.5/en/explain-output.html#explain-output-columns for more details.
Note: Type:range means that “Only rows that are in a given range are retrieved, using an index to select the rows”.

   b. Query 2)

Execute:

```
> EXPLAIN SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and C.email like '%d.com'
```

```
+ ------- + ---------------- + ---------- + ---------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- +
------------- + ---------- +
| id      | select_type     | table     | partitions     | type     | possible_keys     | key     | key_len     | ref     | rows     |
filtered     | Extra     |
+ ------- + ---------------- + ---------- + ---------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- +
------------- + ---------- +
| 1     | SIMPLE        | C        |            | index     | PRIMARY          | ix_Customer_email | 93         |        | 20      |
11.11        | Using where; Using index |
| 1       | SIMPLE          | CRF      |            | ref      | PRIMARY          | PRIMARY  | 4          | cs122a.C.cid | 1       |
100.00      |          |
+ ------- + ---------------- + ---------- + ---------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- +
------------- + ---------- +
2 rows
```

Refer to https://dev.mysql.com/doc/refman/5.5/en/explain-output.html#explain-output-columns for more details.

Note: Type:index means that "If the index is a covering index for the queries and can be used to satisfy all data required from the table, only the index tree is scanned. In this case, the Extra column says Using index. An index-only scan usually is faster than ALL because the size of the index usually is smaller than the table data.".

c. Query 3)

Execute:

```
> EXPLAIN SELECT C.cid, C.email, CRF.flight_number, CRF.projected_departure_datetime, CRF.quantity FROM
Customer C, Customer_Reserves_Flight CRF
WHERE C.cid = CRF.cid and CRF.projected_departure_datetime <= '2015-07-01 00:00:00'

+ ------- + ---------------- + ---------- + ---------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- +
------------- + ---------- +
| id      | select_type     | table     | partitions     | type     | possible_keys     | key     | key_len     | ref     | rows     |
filtered     | Extra     |
+ ------- + ---------------- + ---------- + ---------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- +
------------- + ---------- +
| 1       | SIMPLE          | CRF      |              | range     | PRIMARY,ix_CRF_projected_departure_datetime |
ix_CRF_projected_departure_datetime | 5          |        | 1       | 100.00       | Using index condition |
| 1     | SIMPLE        | C        |            | eq_ref    | PRIMARY          | PRIMARY  | 4          | cs122a.CRF.cid | 1       |
100.00      |          |
+ ------- + ---------------- + ---------- + ---------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- +
------------- + ---------- +
2 rows
```

Refer to https://dev.mysql.com/doc/refman/5.5/en/explain-output.html#explain-output-columns for more details.

Note: Type:range means that "Only rows that are in a given range are retrieved, using an index to select the rows".

8. [10 pts] Now, by comparing the query plan of Q4 and Q7, briefly explain how the indexes are being used by the query optimizer to make these queries run faster.
   Expensive operations such as Full Table Scan can be replaced by a full index scan or index range scan if an applicable predicate exists.

9. [10 pts] If you carefully examine the execution plan of Query 1 and Query 2 after creating the ix_Customer_Email index, you can notice that the index is being used for Query 2. Can you think of a reason why this index is being used?
   Normally, the optimizer doesn't choose an index if a search predicate is a suffix search condition. However, for this case, the optimizer has decided that doing full index scan and traverses the corresponding records costs less than doing full table scan since the size of index is much smaller than that of the table.

[Extra Point Questions]

1. [10 pts] Consider the following SQL query.

```
SELECT total_amount, count(*) from DishOrder DO, Dish D, Lounge L
WHERE DO.lid=D.lid and DO.total_amount > 300 and
      D.price > 30 and DO.lid = L.lid and L.airport_IATA_code like 'S%'
GROUP BY DO.total_amount
HAVING count(*) > 1
ORDER BY DO.total_amount;
```

   First, check its visual query plan (no need to turn it in). Then find a way to improve its execution speed by creating index(es). To support your argument, attach the "EXPLAIN" output of the query after you create the index(es).

   a. The textual "EXPLAIN" plan of the original query.

```
+ ------- + ---------------- + ---------- + --------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- + ------------- +
---------- +
| id      | select_type      | table      | partitions      | type      | possible_keys      | key      | key_len      | ref      | rows      | filtered
| Extra      |
+ ------- + ---------------- + ---------- + --------------- + --------- + ------------------ + -------- + ------------ + -------- + --------- + ------------- +
---------- +
| 1      | SIMPLE      | DO      |          | ALL      | lid          |      |          |      | 18      | 33.33      | Using where; Using
temporary; Using filesort |
| 1      | SIMPLE      | L      |          | eq_ref      | PRIMARY,airport_IATA_code | PRIMARY | 4          | cs122a.DO.lid | 1
| 69.23      | Using where |
| 1      | SIMPLE      | D      |          | ref      | PRIMARY          | PRIMARY | 4          | cs122a.DO.lid | 2      | 33.33      |
Using where |
```

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

b. Your statement(s) to create index(es).

```
CREATE INDEX Do_total_amount_idx on DishOrder(total_amount);
```

c. The textual “EXPLAIN” plan of the query after creating the index(es).

```
+ ------- + ---------- + ---------- + ---------- + --------- + ------------------ + -------- + ------------ + -------- + --------- + ------------- +
---------- +
| id      | select_type      | table      | partitions      | type      | possible_keys      | key      | key_len      | ref      | rows      | filtered
| Extra      |
+ ------- + ---------- + ---------- + ---------- + --------- + ------------------ + -------- + ------------ + -------- + --------- + ------------- +
---------- +
| 1      | SIMPLE          | DO      |          | range    | lid,ta_idx      | ta_idx    | 5      |      | 3      | 100.00      | Using
index condition; Using where |
| 1      | SIMPLE          | L      |          | eq_ref    | PRIMARY,airport_IATA_code | PRIMARY    | 4      | cs122a.DO.lid | 1
| 69.23      | Using where |
| 1      | SIMPLE      | D      |          | ref      | PRIMARY      | PRIMARY    | 4      | cs122a.DO.lid | 2      | 33.33      |
Using where |
+ ------- + ---------- + ---------- + ---------- + --------- + ------------------ + -------- + ------------ + -------- + --------- + ------------- +
---------- +
```