Write down the following queries in SQL and run them in SQLite3 over the database created. You will be writing ONE SQL statement for every query (here One SQL statement starts with a SELECT and ends with a semicolon but may include multiple select statements combined in the form of subqueries and/or using set operations). Your SQL queries for questions 1-3 cannot use any of aggregation, grouping, or nesting (set operations are ok).

1. Find (name and email of) members who have at least two cars and at least one car is associated with a ride.
2. Find (name and email of) members who have cars and bookings but have not offered any ride.
3. Find (email of) members who are booked on a ride from *Edmonton* to *Calgary* in *November 2018*. *Hint*: Check out the date and time functions in SQLite.
4. Find requests that are served by the rides offered. A request is served by a ride if the dates are the same, the ride price is not greater than the requested amount, and the pickup and the drop off cities and provinces are also the same (the street addresses may not be the same). For each qualifying request, list the rid of the request, the requester's email, the location codes for the requested pickup and drop-off locations, and the rno of the rides that serve it.
5. Find top 3 destination cities with the largest number of rides. For the qualifying cities, list both the province and the city. *Hint*: Check out the *limit* clause for SQLite.
6. For every city, list the city, the province it is in, the number of rides from, the number of rides to, and the number of rides enroute that city. Include every city where at least one of the counts is non-zero. *Hint*: you may find outer join useful.
7. Find (the rno of) the cheapest ride(s) from *Edmonton* to *Calgary* in *October 2018* with available seats (meaning not all seats are booked).

8. Find (email of) members who have offered rides to more than half of the locations in *Alberta* and all their rides take place in *2016* or after.
9. Create a view called *ride_info* with columns *rno, booked, available, rdate, price, src*, and *dst*. The view includes for each ride in future, respectively the rno of the ride, the number of seats booked, the number of seats available, the date, the price, the source city and the destination city.
10. Using the view created in Q9, find all ride(s) from *Edmonton* to *Calgary* in *December 2018* with available seats (meaning not all seats are booked). For each qualifying ride, list all information in the view, the email of the member offering the ride, and the number of days from the date of the ride to *Jan 1, 2019*. Sort the result on price from the cheapest to more expensive ones.

(upto 5 bonus marks for the first 3 people) Preparing test data

Written queries should be tested for correctness and bug fixes, very much like programs written in any programming language. For testing, you need to have enough data in your tables such that all your queries are meaningful and non-trivial (e.g. the returned answers are not empty). You are encouraged to share your data with your classmates or use data prepared by them. *To make this collaboration happen, there will be up to 5 bonus marks (at the instructor's discretion) to the first 3 people who prepare a test data and share it with the rest of the class. Make sure your data is correct and meets the expectation of the assignment. If you are sharing your test data, please post it to the course discussion forum.* Put all your *insert* statements in a file called *a2-data.sql*. Make sure to put down your name, email and a date when it is published or revised at the beginning of the file as a comment line (e.g. -- Data prepared by <firstname lastname>, <email address>, and published on <date>). If you are using data prepared by someone else, leave the identification line unchanged.

(10 marks) Testing and report

Starting from scratch, create your database as

```
sqlite3   a2.db   <a2-tables.sql
```

and populate your tables using data file a2-data.sql (prepared in the previous step) as

```
sqlite3   a2.db   <a2-data.sql
```

Put all your SQL queries in a file named a2-queries.sql; Add the following line at the beginning of the file

```
.echo on
```

and the following line before each SQL query (replacing X with the query number).

```
--Question X
```

Run your queries on your data file as

```
sqlite3   a2.db   <a2-queries.sql >a2-script.txt
```

You will be submitting both a2-data.sql and a2-script.txt electronically as described in the instructions for submissions.