INF-2700 Mandatory Assignment Nr. 1

Deadline: Monday, 25 September 2023 23.59

In this assignment, you are going to perform SQL queries on a given database, through a SQLite shell. You are also going to implement a tiny SQL tester in C.

PART 0. Oranizing Your Files

We suggest that you work in Linux. The descriptions below assume you are using Linux. MacOS should work with minor changes. Windows? We wish you good luck.

We are going to use the Gitlab server `https://inf2700.cs.uit.no` for course materials and assignments.

In what follows, we will refer to `{USERID}` as your username, assuming your UIT email address is `{USERID}@post.uit.no` or `{USERID}@uit.no` (where `{USERID}` could be something like `abc012` or `xyz345`).

Make a new directory `inf2700`, on your lab PC, your own PC, or both. This is where you are going to store your INF-2700 materials and all your assignments.

Shared materials

On the git server, there is a project `shared` for course materials available to everybody of this course.

Under `inf2700/` that you have just created on your PC, run:

- `git clone git@inf2700.cs.uit.no:2023/shared.git shared`

or

- `git clone https://inf2700.cs.uit.no/2023/shared.git shared`

  This second option requires you to add your public SSH key to `inf2700.cs.uit.no`.

  Run the following to generate strong keys:

  ```
  ssh-keygen -C "{USERID}@uit.no" -t ed25519 -o -a 256
  ```

Under your local `inf2700/shared/` you will find the materials you have got from the `shared` project. For example, you can find this document as

- `inf2700/shared/assignments/assignment-1/docs/specification-assignment1.pdf`.

Your own project

There has been craeted your own *private* project with the name `{USERID}`. This is where you are going to sumbit your assignment solutions.

Your project is given access to group `ta2023`. This will allow your TA to grade your work and give feedback.

Again, under your local `inf2700/`, run:

- `git clone https://inf2700.cs.uit.no/{USERID}/{USERID}.git {USERID}`

or

- `git clone git@inf2700.cs.uit.no:{USERID}/{USERID}.git {USERID}`

You might get a warning that you are cloning an empty project. That is fine.
On your PC, `inf2700/{USERID}/` is the place hosting all your assignments.
Now under your local `inf2700/`, do the following:

- `cp -R shared/assignments/assignment-1 {USERID}/`

You are going to work under `inf2700/{USERID}/assignment-1/` for this assignment. Here, you will find

- `docs/` for your project documents, including your final report (along with this specification that is already included here).
- `scripts/` for your SQL scripts (PART I).

  You will also find the database of this assignment here. You can, for example, work on the database with the following command:

  - `sqlite3 inf2700_orders.sqlite3`

    You are now inside the `sqlite3` shell. Run `.help` to learn how to use the shell and `.quit` to leave the shell.

- `sql-tester/` for your C source code, Makefile etc. (PART II).

  The pre-code and some related files are already included here.

When you are working on the assignment, you should make frequent commits to your private `{USERID}` project. This way, your are able to keep track of your progress. You are also able to revert to a previous state if you regret your latest updates. This also makes it easy for you to work from different PCs.
In fact, you can (should) `git add` all files you have just copied and make your first `git commit`. This is your starting point of the first assignment.

PART I. SQL

In PART I, you are going to familiarize yourself with basic database operations and SQL using SQLite.
Check SQLite documentation for more information about SQLite.

Task 1. Database schema

Describe the `inf2700_orders` database schema.
It is sufficient to show the `CREATE TABLE` statements that defined the schema.

Task 2. Run the given SQL queries

Run the queries listed below.
For each query, write in your own words in a few lines that describe its purpose and how it works.

```
a)  SELECT customerName, contactLastName, contactFirstName
    FROM   Customers;

b)  SELECT *
    FROM   Orders
    WHERE  shippedDate IS NULL;

c)  SELECT C.customerName AS Customer, SUM(OD.quantityOrdered) AS Total
    FROM   Orders O, Customers C, OrderDetails OD
    WHERE  O.customerNumber = C.customerNumber
           AND O.orderNumber = OD.orderNumber
    GROUP BY O.customerNumber
    ORDER BY Total DESC;
```

```
d)  SELECT P.productName, T.totalQuantityOrdered
    FROM   Products P NATURAL JOIN
           (SELECT productCode, SUM(quantityOrdered) AS totalQuantityOrdered
            FROM   OrderDetails GROUP BY productCode) AS T
    WHERE  T.totalQuantityOrdered >= 1000;
```

Task 3. Write your own SQL queries

Solve the following problems in SQL:

1. Retrieve all customers in Norway.
2. Retrieve all *classic car* products and their *scale*.
3. Create a list of incomplete orders (order status is "In process"). The list must contain orderNumber, requiredDate, productName, quantityOrdered and quantityInStock.
4. Create a list of customers where the difference between the total price of all ordered products and the total amount of all payments exceeds the credit limit. The list must contain the customer name, credit limit, total price, total payment and the difference between the two sums.
5. Create a list of customers who have ordered all products that customer 219 has ordered.

You are encouraged to run as many SQL queries as you like, such as the ones similar to the examples in the text book, though you do not have to report this effort in your hand-ins.

PART II. Database Programming

In PART II, you are going to implement a SQL tester in C.

Task: Implement a SQL tester

The SQL tester tests if SQL queries give the expected results.

In order to perform such tests, we need

- A database to test against.

  You can find the sample database at `test/messages.sqlite3`.
- The queries.

  You can find the sample queries at `test/queries_messages.sql`.
- Test cases that specify the expected results of the queries.

  You can find sample test cases at `test/test_queries_messages`.

You can find the pre-code of the SQL tester at `src/testsql.c`. At places with comments starting with `TODO:`, you must implement the missing pieces to make a complete tester.

Running a test with the given sample data should look like the following:

```
$ ./testsql test test_queries_messages
test "all_countries" succeeded
test "messages_to_people_other_than_the_author" succeeded
test "empty_result" succeeded
test "wrong_result" failed
exepected:
|correct result|

got:
|wrong result|

test "erroneous_query" error: no such table: fake_tbl
test "non_existent" not found.
$
```

Hand-in

When you have finished, commit all your hand-in files and push your commits to `inf2700.cs.uit.no`.

Your commits should include the SQL and C source code, Makefile etc. In addition, you must hand in a report that includes your answers and any special observations you made, problems you experienced etc. The report must be a PDF file.

Make sure you have committed and pushed all your final working before the deadline.

Please *do not* include irrelevant files or directories. For example, if you are using MacOS, add a line containing `.DS_Store` to your `.gitignore` file.

Enjoy coding and good luck!