DSCI 551 – HW3 (SQL for Covid-19)

(Fall 2020)

100 points, Due 10/25 23:59pm, Sunday

In this homework, you will be analyzing a data set about Covid-19 cases and related information in South Korea. The data set is from Kaggle: https://www.kaggle.com/kimjihoo/coronavirusdataset and has been converted into MySQL tables and provided to you as a single SQL file. The database is available under the Resources folder on Blackboard. You may also find this page and the Jupyter notebook available there useful for you to understand the dataset.

You will need to first create a database called "covid19" and import the data set into the database using the following command:

```
mysql covid19 -u dsci551 -p < covid19.sql
```

Note that you will also need to create a user "dsci551" with password "dsci551" before you execute the above command.

For each of the following tasks, write a Python script that uses mysql.connnector to get necessary data from the database to complete the task.

1. **Case reporting [30 points]**: Write a Python script "report.py" that takes a month (mm/yyyy format) and reports the number of confirmed and deceased cases by gender, age, and province. You will use *time, timeage, and timeprovince* tables. The report will be in JSON format file as described below.
   Execution format: python report.py "mm/yyyy" output_file_name
   For example,

   ```
   $python report.py "03/2020" "result.json"
    will generate a report: result.json:
           {"gender":
                   {"male":  {"confirmed": 100, "deceased": 2},
                    "female": {"confirmed": 50, "deceased": 1}},
            "age":
                   {"0s": {"confirmed": 1, "deceased": 0}, "10s": {…}, …},
           "province":
                    {"Seoul": {"confirmed": 10, "deceased": 0}, "Busan": {…}}
       }
   ```

   (Note that actual number might be different. Also note if there're no cases in this month found in a table, leave the value empty like "gender":{})

2. **Key finding [30 points]**: The dataset does not come with key definition for the tables. So our goal is to find which attributes may serve as the key for a table. For this, write a Python

script "key.py" that takes a table name and a list of attributes (comma separated) in the table, reports "yes" if the given list of attributes can serve as the key for the table; if not, reports the top 5 duplicate values of the given attributes and their counts (so that we are sure that they cannot be a key).

For example,

```
$ python key.py "case" "case_id"
```

will simply print "yes"

```
$ python key.py "patientinfo" "patient_id"
```

will print:

```
    1200012238, 2
```

This means that patient_id "1200012238" appears in two rows of the table.

```
$ python key.py "patientinfo" "province,city"
```

will print:

```
    Gyeongsangbuk-do, Gyeongsan-si, 638
    Gyeonggi-do, Seongnam-si, 173
    Gyeonggi-do, Bucheon-si, 162
    Seoul, Gwanak-gu, 113
    Chungcheongnam-do, Cheonan-si, 110
```

This means that there are 638 rows in patientinfo table with province = Gyeongsangbuk-do and city = Gyeongsan-si.
Note that there may be other values of (province, city) that appears more than once (duplicates) in the table, but their counts are less than 110 and thus not reported.

3. **Infection tracing [40 points]**: The *patientinfo* table records the id of the patient (patient_id attribute) and the id of the person the patient was infected from (infected_by attribute). Patient "1000000003" was infected from patient "2002000001".
Note that the data set may not record the infection source for every patient (e.g., the infected_by value of patient 1000000001 is an empty string. Note also that the data set may have errors, e.g., a patient may be directly or indirectly infected by him/herself (that is, a cycle formed by the infected_by relationship) as the tracing program will find out.

Specifically, your task is to write an infection tracking program "trace.py" that takes a patient id as the input and output the ids of patients whom the patient was infected from directly and indirectly.

For example,

```
$python trace.py "1000000006"
```

Will output:

1000000006, 1000000003, 2002000001, …

Your program should also report if it finds a cycle (in this case, it should report all patient ids that are in the cycle only once, except for the first and last ids).

For example, here is how you should ouput in the case of a cycle:

Cycle found: 1000000006, 1000000003, 2002000001, 1000000006

Note that the order shouble remain the infection trace back chain order, i.e. the original order you find the result.

Requirements:

- You should utilize the SQL queries for the analysis tasks above as much as possible and should not retrieve the entire content of the table from the database and perform the analysis in Python. You should only retrieve data that are necessary to answer the question. For example, in infection tracing, you should not retrieve ids of patients who are not infected (directly or indirectly) by the given patient. For another example, in key finding, think about how to write SQL queries to determine if given attributes form a key.
- Permitted python libraries: mysql.connector, json,sys

Submission:

Please use Python 3 and include all scripts for 3 questions in a zip folder named Firstname_Lastname.zip.