Advanced Database System Homework2
Collaborators: Mengna Qiu (mq438), Juanlu Yu (jy2234)

Files included
adv database hw2.pdf -- Answer sheet
Fractal.java -- Generate trade table
Q1-aquery-timing.txt -- Q1 Aquery timing script
query1.a -- Q1 Aquery (a) code
query2.a -- Q1 Aquery (b) code
query3.a -- Q1 Aquery (c) code
query4.a -- Q1 Aquery (d) code
Q1-postgresql-timing.txt -- Q1 Postgresql timing script
Q1-postgresql.sql -- Q1 Postgresql (a) (b) (c) (d) code
q3-1.a -- Q3 solution 1 code (slow)
q3-2.a -- Q3 solution 2 code (fast)
repro-pack-q1-aquery.rpz -- Reprozip for Q1 Aquery part
repro-pack-q1-psql.rpz -- Reprozip for Q1 Postgresql part
repro-pack-q3.rpz -- Reprozip for Q3 Aquery part

Q1
We choose Aquery and Postgresql to complete this question.

Fractal.java is the code to generate the trade table.

For the Aquery part, *query*.a* are the code files, and *Q1-aquery-timing.txt* is the script file.
First using aquery.jar to compile the *.a files into *.q files in command line:

```
java -jar aquery.jar -a 1 -s -c -o query1.q query1.a
java -jar aquery.jar -a 1 -s -c -o query2.q query2.a
java -jar aquery.jar -a 1 -s -c -o query3.q query3.a
java -jar aquery.jar -a 1 -s -c -o query4.q query4.a
```

and then use vi query.q for each .q file to add \\ at the end of the file
and then timing the execution of the file *q3-1.q* and *q3-2.q*, using command line:

```
time q q3-1.q
time q q3-2.q
```

For the Postgresql part, *Q1-postgresql.sql* is the code file, and *Q1-postgresql-timing.txt* is the script file.
First, set up the server and database.

```
module load postgresql-9.6.1
initdb -D /home/mq438/pgsql/data -U mq438
pg_ctl -D /home/mq438/pgsql/data -l logfile -o "-F -p 2000" start
createdb my_db -p 2000
psql my_db -p 2000
```

```
pg_ctl -D /home/mq438/pgsql/data stop
```
And then turn on timing using
```
\timing
```
And, for the first question, we copy and paste each query to get the answer. Finally, we add up queries for each subquestion to get the timing.

The Answer sheet, *adv database hw2.pdf*, includes our report.

Q2

We experimented three rules of thumb of database tuning using PostgreSQL and MySQL. All experiments, reports are included in the *adv database hw2.pdf.*

Q3

We came up with two solutions using Aquery. The solutions are written in *q3-1.a* and *q3-2.a*
First using aquery.jar to compile the *.a files into *.q files in command line:
```
java -jar aquery.jar -a 1 -s -c -o q3-1.q q3-1.a
java -jar aquery.jar -a 1 -s -c -o q3-2.q q3-2.a
```
and then execute the file *q3-1.q* and *q3-2.q*, using command line
```
q q3-1.q
q q3-2.q
```
the results are written into *result1.txt* and *result2.txt* separately.
The *adv database hw2.pdf* will include the performance number for two solutions and reasoning about the difference in performance.