This is the report for DBMS class project

Group Member: Kriangsak T., Hasdin,G. This report refers to : GitHub (https://github.com/Kriangsak1997/DatabaseClassProject)

Please treat this to be a small demo...

First of all, we used docker to run out protgresql database: used commands can be found in the file named "command_involvingDockerandPSQL.txt". In this file, you will find what is used to map data to the container.... since i have no time (more like brain) to build a proper image for it.... Please see the following

```
In [8]: # filepath  = '/Users/kriangsak1997/Documents/MUIC/Term8/DBMS/Project/command_involvingDockerandPSQL.txt'
        # with open(filepath) as fp:
        #    line = fp.readline()
        #    cnt = 1
        #    while line:
        #        print("Line {}: {}".format(cnt, line.strip()))
        #        line = fp.readline()
        #        cnt += 1
```

```
Line 1: This is when you create a table in dockerized postgres
Line 2: First thing first, let me just give you the command to run
postgress
Line 3:
Line 4: docker run --name snpContainer  -p 543x:5432 -d -e POSTGRE
S_PASSWORD=1234 postgres
Line 5:
Line 6: You now have a container named: snpContainer
Line 7: Now run
Line 8: docker exec -it snpContainer psql -U postgres
Line 9:
Line 10: and then you will create the table using the folling comm
ands
Line 11:
Line 12:
Line 13: CREATE TABLE SNP_DB(Main_Gene_name  varchar, initialAA va
rchar,finalAA varchar,position_of_Change varchar,Type_of_Variant v
archar,dbSNP varchar,Disease_name varchar);
Line 14:
Line 15: This is when you copy the file to docker container... I d
o this because I am not good at docker and fdont know how to creat
e a container for psql and persist the data
Line 16:
Line 17: docker cp snp_relation.csv 286f68d1cdd8:/snp_relation.csv
Line 18:
Line 19:
Line 20: now you can copy the file from  your container to your da
tabase
Line 21: hopefully this will work
Line 22:
Line 23: COPY snp_db  FROM '/snp_relation.csv' DELIMITER ',' CSV H
EADER;
Line 24: COPY 78244
Line 25:
Line 26:
Line 27: and that's it... you can query from the table: snp_relati
on
Line 28: i.e.,
Line 29: select * from snp_relation limit 5;
Line 30:
Line 31:
```

Connect to the Database

After having created the database and import the file into it, we will connect to the database using Java. Here with help of **Maven** dependencies, it allows to very easily connect to our database, we can dp so by
creating a Maven project. Then access the file: *pom.xml*
and we add the dependencies:

> </dependency/>
>
> > ```
> >         </groupId/> org.postgresql</groupId/>
> >          </artifactId/>postgresql</artifactId/>
> >          </version/>42.2.10</version/>
> > ```
> >
> > </dependency/>

Then we initialize a few variabls for our database connection, you can refer to any of the sources code:
notebooks/Association Rule Mining with Apriori algorithm.ipynb or
Project/SNP_Project/src/main/java/WorkingWithDB/ConnectionFactory.java

Now we can query from out database for all: CRUD queries

From this, we take a data-analytic approach to our database. That is, we will apply a small mining algorithm to it.

Associative Rule Mining with Apriori Algorithm

for those wishing to understand more about this concept, [consult (https://blog.usejournal.com/association-rule-mining-apriori-algorithm-c517f8d7c54c)](https://blog.usejournal.com/association-rule-mining-apriori-algorithm-c517f8d7c54c) There are a few things to introduce for those new to Associative Rule Mining or, perhaps also, Apriori Algorithm.
the results of out experiments is in the file: notebooks/Association Rule Mining with Apriori algorithm.ipynb with a few tials of different hyperparemeters (supports, Confidence). It turns out that the with conditional probabiity of 25%, this gives out 3 frequent item set(of 2 elements): **('Asp', 'Asn'), ('Tyr', 'Cys'), ('Glu', 'Lys')** , please refer the mentioned notebook.

Functionality

Our database offeres simple functionalities of *CRUD* but with limited time and given the panic of COVID-19 fear... we are not able to deliver an appication. Instead, we have made a switch case class that allows for visualizations of our database. There are a few premade data-visualizing queries. With additional queries, users can input their queries of interest.
I dont know what else to write for this report but to please ask you to kindly check through our GitHub link for all the work we have done. thank you.

```
In [ ]:
```