Homework D – Database Query Processor

TA: Jay Huang(hwc105m@cs.ccu.edu.tw)
Deadline:2017 Jan. 5, 11:59pm

1. **Introduction:**

In this homework, you are going to implement a database query processor. A CSV file of contacts is provided as a database table. **First**, you need to provide a user an interface to query data from your database by inputting the query command. **Second**, you need to read the raw data from a database (in this homework, read from a file) and to process the query command, and in the end, output them to user. The detailed SPECs are listed below.

1.1 **Database file format (CSV file):**

Each line in the file is considered as a raw data. The columns of each raw data are defined as below:

Column Name	Data Type
id	Int(11)
FirstName	Varchar(10)
LastName	Varchar(10)
Gender	Varchar(1)
Age	Int(3)
PhoneNum	Varchar(15)

Notice that the column names are **case-sensitive**.

1.2 **Query command format**

A user should follow the format when query data. Otherwise, your program should print out the error message. Query format are defined as below:

SELECT columns
FROM [FileName].csv
ORDER BY col1 [ASC/DESC] –sort1 [, col2 [ASC/DESC] –sort2]

Notice:

1 ‘SELECT’, ‘FROM’, ‘ORDER BY’ are case-insensitive, and others are case-sensitive.

2 All the query commands are inputted as a single line.

3 In the query command, SELECT and FROM clauses are required. In case of any incomplete commands, **print out** an **error message**.

2. **Implement the query processor:**
   A test file (Contacts.csv) is provided.

2.1 **User Interface**

Your program needs to allow the user to query as **many times** as they want **until** they input '**quit**'.

2.2 **SELECT** columns

SELECT columns clause indicates what columns need to be shown. User can either input the specific column names, or simply input * to show all the columns.

Examples:

SELECT Age, PhoneNum, id FROM Contacts.csv

SELECT * FROM Contacts.csv

2.3 **FROM** FileName.csv

FROM FileName.csv indicates the data source. You should read the raw data from the FileName.csv.

2.4 **ORDER BY** col1 [ASC/DESC] –sort1 [, col2 [ASC/DESC] –sort2]

ORDER BY clause requires three parameters: the column, sorting type, and sorting algorithm that you need to process the data before printing them. In this homework, you need to implement **selection sort** and **insertion sort** to sort your data. If the column value of two data tuples is the same, keep their original order in the data set.

2.4.1 **Implement two algorithms**

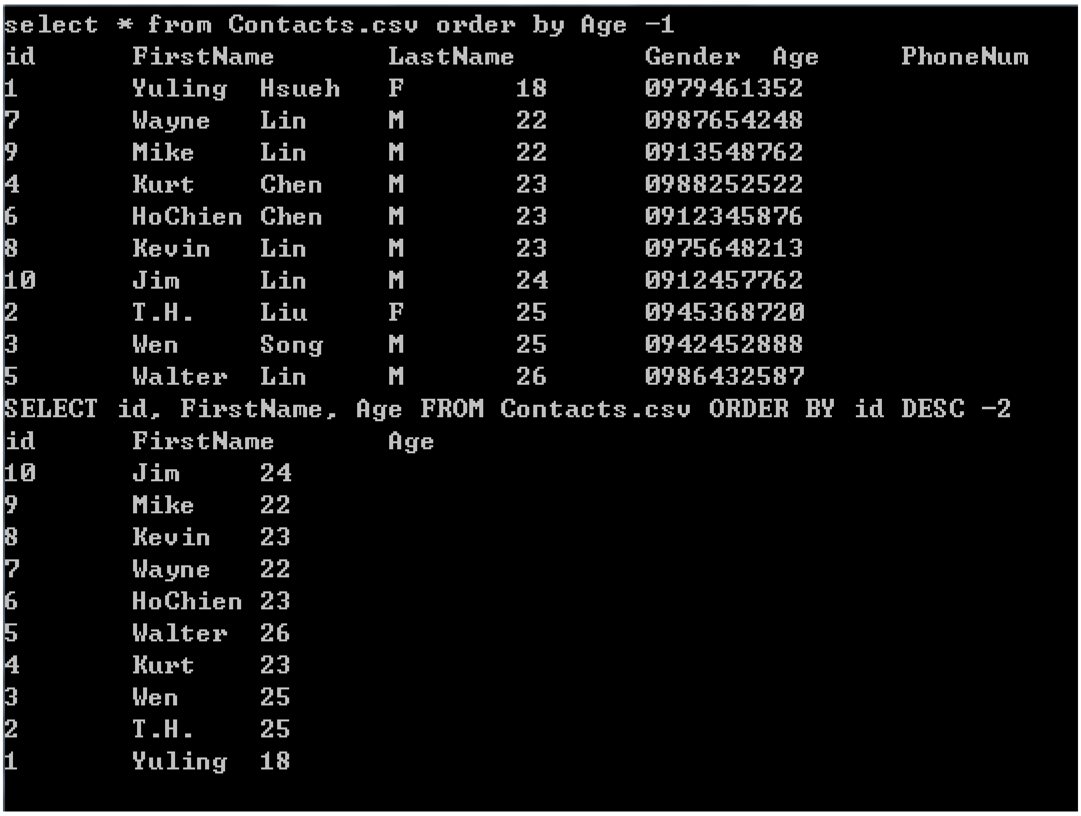
The ORDER BY command format consists of the following three parameters in order:

1. **Sorted column**: the specific column the user wants to sort
2. **Sorting Type**: **ASC** means an ascending order, while **DESC** means a descending order. Notice that sorting type can be ignored, and if it does, **the default value is ASC**.
3. **Sorting algorithm**: you need to implement two algorithms (**selection sort** and **insertion sort**) which can be chosen by the user.

The command format is given here:

SELECT * FROM Contacts.csv ORDER BY columns [ASC/DESC] [-1/-2]

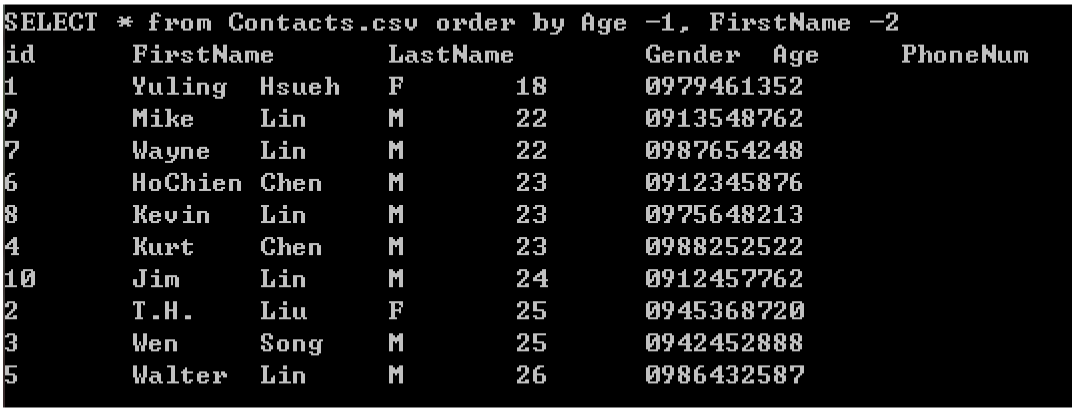
Examples:



```
select * from Contacts.csv order by Age -1
id      FirstName       LastName        Gender  Age     PhoneNum
1       Yuling  Hsueh   F       18      0979461352
7       Wayne   Lin     M       22      0987654248
9       Mike    Lin     M       22      0913548762
4       Kurt    Chen    M       23      0988252522
6       HoChien Chen    M       23      0912345876
8       Kevin   Lin     M       23      0975648213
10      Jim     Lin     M       24      0912457762
2       T.H.    Liu     F       25      0945368720
3       Wen     Song    M       25      0942452888
5       Walter  Lin     M       26      0986432587
SELECT id, FirstName, Age FROM Contacts.csv ORDER BY id DESC -2
id      FirstName       Age
10      Jim     24
9       Mike    22
8       Kevin   23
7       Wayne   22
6       HoChien 23
5       Walter  26
4       Kurt    23
3       Wen     25
2       T.H.    25
1       Yuling  18
```

2.4.2 **Double sort**

The second sorting command is optional for the user. When the first sort is finished, **the second sort looks up the columns based upon the first sort**. If there existing multiple data records have the same value based upon the first sort, the second sort sorts those records by the second specified column.

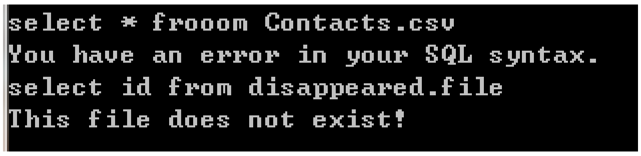


```
SELECT * from Contacts.csv order by Age -1, FirstName -2
id      FirstName       LastName        Gender  Age     PhoneNum
1       Yuling  Hsueh   F       18      0979461352
9       Mike    Lin     M       22      0913548762
7       Wayne   Lin     M       22      0987654248
6       HoChien Chen    M       23      0912345876
8       Kevin   Lin     M       23      0975648213
4       Kurt    Chen    M       23      0988252522
10      Jim     Lin     M       24      0912457762
2       T.H.    Liu     F       25      0945368720
3       Wen     Song    M       25      0942452888
5       Walter  Lin     M       26      0986432587
```

2.5 **Mistake-proofing**

You should do some basic mistake-proofing.

1. Typo of 'SELECT', 'FROM', 'ORDER BY'
2. A valid Query command requires both 'SELECT' and 'FROM' clauses to work.
3. Check whether the input file exists.



```
select * frooom Contacts.csv
You have an error in your SQL syntax.
select id from disappeared.file
This file does not exist!
```

2.6 **Print out format**

1. **Data record**:

You should print the header for the first line, and then print out the data record each for a line. You should print '**\t**' between columns to make it look better.

2. **Error message**:

There are some messages for the mistake-proofing function. When there's a mistake in user's input, print out "**You have an error in your SQL syntax**". On the other hand, if the data source file does not exist, you should print out "**This file does not exist!**".

2.7 **Readme, comments and style**

An indicator for good source code is readability. To keep source code maintainable and readable, you should **add comments to your source code** where reasonable. For this assignment, please also compose a small "**README.txt**" which contains a brief explanation of how to compile your program and what problem you met. Please remember to have your **source code comments** in English.

3. Grading

The TA(s) will mark and give points according to the following grading polices:

90% Run your program with 5 testing data

15% User interface, Mistake proofing

25% 'SELECT','FROM' clauses

25% Single sort with ASC(default), DESC

25% Double sort based upon the first sort

10% Readme, comments and style.

Provide a README.TXT that contains information about the program and brief reviews. Source code is readable and has ENGLISH comments where reasonable.

4. How to submit

The files you MUST submit:

1. Source Code
2. README.TXT

To submit your file electronically, enter the following command from csie workstation:

- turnin ds.hw4 [your files…]

To check the file you turnin, enter the following command from csie workstation:

- turnin –ls ds.hw4

You can see other description about turnin from following link:

https://www.cs.ccu.edu.tw/lab401/doku.php?id=turninhowto