程式設計（107-1）
作業九

作業設計：楊其恆，孔令傑
國立臺灣大學資訊管理學系

繳交作業時，請至 PDOGS（`http://pdogs.ntu.im/judge/`）為第一題做同儕互評，再為第二、三、四題各上傳一份 C++ 原始碼（以複製貼上原始碼的方式上傳）。第五題是 bonus 加分題。每位學生都要上傳自己寫的解答。不接受紙本繳交；不接受遲交。請以英文或中文作答。

這份作業的截止時間是 ***12 月 11 日凌晨一點***。在你開始前，請閱讀課本的第 22.1–22.6 節與第 9–10 章[1]。為這份作業設計測試資料並且提供解答的助教是楊其恆。

第一題

（0 分）請在 PDOGS 上批改你被隨機分配到的作業七第三題的程式碼，根據它在正確性以外的部份給它 1 至 5 分的評分，並且說明你給分的依據。建議在評分時參考以下六個面向。在前五個面向上，一個面向上做得好就得一分，還不錯則半分，不好則零分；在第六個面向上則在有必要時扣分。六個面向的分數合計後無條件進入即為你最後給的總分。

- 可讀性：變數與函數名稱是否具有合適的資訊量？程式碼排版是否良好且具有前後一致性？是否有合適的註解？關於註解，當然不需要每一行都有註解，但若你發現在某一大段落裡都沒有註解，或某個你感覺很不易看懂的部份沒有註解，你可以指出來；不要直接說「註解太少」但沒有說是哪邊缺乏註解。

- 模組化程度：是否有宣告合適的函數？是否有避免將非常類似的程式片段寫複數次而非寫成函數？是否有避免一個函數做非常多事情？函數間是否有合適的 decoupling？直接閱讀 main function 是否能很快地理解程式在大方向上的運算邏輯？

- 效率：程式運算是否有合理的運算效率？當然我們不要求每個同學都寫出超級有效率的精妙演算法，但至少一個程式不應該進行過多不必要的運算，也不應該耗用過多不必要的記憶體空間。如果你看不出這個程式的效率有明顯的問題，我們建議你直接給一分。

- 擴充性：當要解的問題變得更複雜的時候，我們能不能簡單地修改這個程式以解決新的問題，而不是寧可砍掉重練？這個議題當然也很主觀，所以如果你不能明確地指出在怎樣的新問題上，這個程式會有擴充性問題，我們建議你直接給一分；如果你不能指出很嚴重的問題，我們建議你至少給半分。但對批改者來說，這個關於擴充性的思考其實是很好的訓練。試試看吧！

- 其他：如果有任何其他令你想扣分的理由，請明確地寫出來並且在這個面向上扣分；沒有的話就給一分。

- 題目規範：你應該檢查那份程式碼有沒有違反題目的規範，如果有（例如題目說不可以用上課沒教過的東西，但他用了，或者題目說一定要用指標和動態記憶體配置，但他沒用），就扣他三分。當然，請明確地指出他哪邊違反了題目的規範。

[1] 課本是 Deitel and Deitel 著的 *C++ How to Program: Late Objects Version* 第七版。

該題其中 10 分取決於檢視你的程式碼的同學給你的分數總和（必要時助教會出來主持公道，請不用緊張），另外 10 分取決於你對同學的程式碼的評語和評分的合理性和建設性（原則上除非被申訴，且助教檢視後發現你確實評得很不公允，否則只要有評就會得到 10 分）。

第二至五題的前言

物件導向程式設計一個重要的功能，就是為了讓程式設計師除了自己寫 code 以外，也可以輕易地利用別人所寫之程式，在本次作業中，就要讓大家來使用助教寫好的 class，並在已有的架構中，擴展程式的功能。

在資料分析的領域中，資料庫的使用相當重要。相較於傳統的資料表格，資料庫除了可以提升資料的完整性與安全性，還能提供一致的介面供資料分析人員取用及分析資料。「MySQL」是常見的資料庫之一。簡易的語法與操作使之成為一般學生學習所接觸的第一個資料庫。一個 MySQL database 由多張「表」（table）組成，table 與我們平時所見到的資料表雷同，只是 MySQL 會對每個資料欄位（通常稱為 field）規範它的資料型別，一旦輸入資料格式不符，便會拒絕資料的加入。

在 MySQL 中下達指令（通常稱為 query）後，便可根據指令的內容將所需的資料顯示給使用者。最常用的查詢指令包含了 SELECT、FROM、WHERE 三個子句，關於其意義與使用方法可以參見以下二個網頁：

- SELECT statement: `https://www.w3schools.com/sql/sql_select.asp`
- WHERE clause: `https://www.w3schools.com/sql/sql_where.asp`

真正的 MySQL 資料庫就是用 C++ 實作而成，大家可以想像 MySQL 是一個用 C++ 寫的應用程式，就像大家平常寫的那些程式一樣；這個 MySQL 應用程式除了有強大的功能、美觀的介面之外，最基本的功能就是儲存資料表以及接受使用者用 SELECT-FROM-WHERE 語法來查詢資料。在本次與下次作業中，我們將請你實做最陽春的 MySQL。為了使查詢及運算速度更快，MySQL 事實上有使用某些資料結構進行優化。由於我們尚未學過相關的資料結構，在本題中請大家使用一般的陣列進行實作即可。

到 MySQL 這樣規模龐大的系統中，如果不使用 class 和 structure，程式碼會非常龐雜，因此我們將使用 class 和 structure 來實作一個可以接收 MySQL query 的程式。在這個程式中會有一些物件，包含一個資料庫、一些資料表，以及一堆指令，彼此的關係是一個資料庫中含有這些資料表，而一個指令會去資料庫中取出一些資料表做處理。助教會提供程式的 class 架構，大家只需要實作部分的 member function。大家所拿到的作業檔案會包含以下幾個：

1. 你正在閱讀的作業說明檔案。
2.「sampleCode.cpp」：助教寫好的程式骨架，請大家從此份程式開始延伸。程式中會標記要完成本次作業，所需要完成的部分。對於函數的實作方法，也可參考其中的註解。
3.「table1.csv」、「table2.csv」：原始資料會存在一些 CSV（comma-separated values）檔案中，在 PDOGS 上供你的程式存取並且做測試。在你自行開發程式的期間，你也需要一些測試用的資料檔，這就是這兩個 CSV 檔的用途了。程式中的 `loadcsv` 函數可以將這些資料讀入。

為了簡化處理的過程，我們將對題目中會出現的 MySQL query 做以下的限制或簡化：

1. 最多只會出現 SELECT、FROM、WHERE 三種子句，且必定照順序出現。
2. SELECT 子句後只會出現欄位名稱，不包含 SQL function 如 COUNT、SUM 等，但可能出現「*」(表示所有欄位)。
3. 資料型態只有字串
4. 在指令中表示字串時，不會使用引號「"」。
5. 資料中不會出現空值。
6. 查詢指令必定會符合 MySQL 的文法。

在本次作業中，我們基本上只處理 SELECT 和 FROM 子句（除了第五題的加分題），也只考慮整數型態的資料內容。

第二題

（20 分）首先，為了使我們的程式可以輕鬆地處理 MySQL 的指令，請大家先寫一個函數

```
Command parseCommand(char command[]);
```

這個函數會讀入代表一個 SQL query 的 `command` 字串，並將讀入的內容解析後存入 `Command` 這個 structure 之中。

一句 SQL query 包含有 SELECT 與 FROM 這兩個關鍵詞、SELECT 子句中的欄位名稱（一或多個），以及 FROM 子句中的資料表名稱（一個）。如果有多個欄位名稱，則最後一個以外的欄位名稱後面會有一個逗點。如果要表現「所有欄位」，則在欄位名稱的部份只要寫出一個「*」即可。關鍵詞、欄位名稱與資料表名稱通稱為「token」，這是資訊科學領域常用的一個詞。整句 query 的最後面有一個分號。

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中，會有二行字串，第一行的字串代表 MySQL 的指令，最後面是一個分號，各 token 之間以一個空白字元隔開。指令中只會包含 SELECT 跟 FROM 兩種子句。第二行包含一個字串，是「SELECT」或「FROM」。讀入這些資訊之後，請將第二行的該子句後的各 token 依序印出，各 token 之間以一個空白字元隔開。

舉例來說，如果輸入是

```
SELECT firstName, age FROM table1;
SELECT
```

則輸出應該是

```
firstName age
```

如果輸入是

```
SELECT firstName, age FROM table1;
FROM
```

則輸出應該是

```
table1
```

如果輸入是

```
SELECT * FROM table2;
SELECT
```

則輸出應該是

```
*
```

這一題的 20 分會根據程式運算的正確性給分。PDOGS 會直譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。針對這一題，你可以使用任何方法。

注意：如果就只是要拿到這一題的分數，你顯然不需要真的處理 structure，你甚至完全不需要用到助教給的程式。但若你真的這麼做，後續的題目就會令你感到很麻煩了。

第三題

（20 分）完成了對 SQL query 的 parsing 後，接著我們可以進入 class 的部分。要讓助教的程式可以順利執行，我們需要做兩件事：

1. 完成 class `Database` 及 `Table` 的 constructor 及 destructor。你需要為這些 class 中的某些變數配置夠大的動態記憶體空間，並且將某些變數初始化為應有的值。空間大小的決定可以參考程式中的 constant integer。如果沒有正確宣告出記憶體空間，可能（多半）會導致 loadcsv 函數發生 runtime error。
2. 完成 class `Database` 之中的 `newTable` 函數。這個函數會將一張新的資料表加入本系統的資料庫中。技術上來說，如果被傳入一個 Table 物件的指標，這個函數應該將這個張表的指標存到 Database 物件中相對應的變數中。

完成至此，現在你可以藉由呼叫 `Table` 的 `loadcsv`、`print` 函數來將資料表由檔案讀入，並輸出到螢幕上。

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中，會有一個字串，代表輸入資料（一張新的資料表）的檔名，檔名字串的長度不超過 100 個字元。讀入這些之後，請使用 `loadcsv` 函數讀入資料，並使用 `print` 函數將資料印出。

舉例來說，如果輸入是

```
table1.csv
```

則輸出應該是

```
-----------------------------------------
| field1  | field2  | field3  | field4  |
-----------------------------------------
```

```
| aaa      | xxx      | -15      | 37g9     |
| bbb      | yyy      | 27       | 1r       |
| ccc      | zzz      | -39      | a4w      |
-----------------------------------------
3 rows in set
```

這一題的 20 分會根據程式運算的正確性給分。PDOGS 會直譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。

注意：如果就只是要拿到這一題的分數，你顯然不需要真的去完成題目要求的任務。但若你真的這麼做，後續的題目就會令你感到很麻煩了。

第四題

（60 分）可以正確讀入資料後，我們來試試簡單的 MySQL query 之執行。在本題中，會給定數張資料表以及一個沒有 WHERE 子句的查詢指令，請大家根據指令的內容印出應有的查詢結果。

要完成這個任務，大家需要做以下的事情：

1. 完成 class `Table` 之中的 `addField` 及 `addEntry` 二個函數。
2. 完成 class `Database` 之中的 `executeCmd` 函數，在這個函數中，根據 query 中指定的表，將 `Command` structure 傳遞給該表，並回傳執行結果。
3. 完成 class `Table` 之中的 `executeCmd` 函數的部分功能，這個函數必須將 query 中指定的 field 選出，並將資料存到一個新的表之中，再回傳該表的位址。在這個過程中，你應該會需要用到 `addField` 及 `addEntry` 函數。

輸入輸出格式

系統會提供一共 20 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中，會有 $n+2$ 行，第一行包含一個整數 n，代表總共有幾張表。第二行開始的 n 行之中，每一行包含二個字串。第 $i+1$ 行的第一個字串代表第 i 張表的名稱，第二個字串代表該表所在的檔案名稱，請使用 `loadcsv` 函數讀取這個檔案中的資料。二個字串以一個空白字元分隔。第 $n+2$ 行包含一個字串，代表要被執行的指令。其中 $1 \le n \le 5$，名稱字串的長度不超過 10 個字元，檔名字串不超過 100 個字元，且二者不含有空白字元。讀入資料後，請根據指令內容，輸出應有的查詢結果。

舉例來說，如果輸入是

```
2
table1 D:\table1.csv
table2 D:\table2.csv
SELECT firstName, age FROM table2;
```

則輸出應該是

```
---------------------
| firstName  | age  |
---------------------
| Lois       | 36   |
| Theresa    | 25   |
| Ruby       | 37   |
| Carolyn    | 59   |
| Anne       | 28   |
| Daniel     | 28   |
| Donna      | 34   |
| Carl       | 44   |
| Alan       | 39   |
---------------------
9 rows in set
```

如果輸入是

```
2
table1 D:\table1.csv
table2 D:\table2.csv
SELECT * FROM table2;
```

則輸出應該是

```
-------------------------------------------------------------------------
| ID    | firstName  | lastName  | Gender  | age  | City         | State  |
-------------------------------------------------------------------------
| a679  | Lois       | Walker    | F       | 36   | Denver       | CO     |
| a352  | Theresa    | Lee       | F       | 25   | Toeterville  | IA     |
| a642  | Ruby       | Rogers    | F       | 37   | Woodbury     | TN     |
| a455  | Carolyn    | Hayes     | F       | 59   | Saint Cloud  | FL     |
| a253  | Anne       | Rogers    | F       | 28   | Stockholm    | SD     |
| a480  | Daniel     | Cooper    | M       | 28   | Manning      | ND     |
| a841  | Donna      | Brown     | F       | 34   | Lima         | NY     |
| a264  | Carl       | Collins   | M       | 44   | Wright       | AR     |
| a137  | Alan       | Rogers    | M       | 39   | Knoxville    | TN     |
-------------------------------------------------------------------------
9 rows in set
```

你上傳的原始碼裡應該包含什麼

你的.cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你***不可以***使用上課沒有教過的方法，但上課提過的 library 裡面的所

有功能都可以用。除此之外，你**一定要**用 class 以及完成題目指定的任務。

評分原則

- 這一題的其中 40 分會根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。
- 這一題的其中 20 分會在作業十中被評定。屆時我們會讓同學們互相檢視彼此的本題程式碼，並且就可讀性、易維護性、模組化程度、排版等面向寫評語和給評分（當然一切都是匿名的）。該任務在本題中會佔 20 分，其中 10 分取決於檢視你的程式碼的同學給你的分數（必要時助教會出來主持公道，請不用緊張），另外 10 分取決於你對同學的程式碼的評語和評分的合理性和建設性。若你在本次作業中完全沒有寫這一題，那屆時自然沒有人能檢視你的程式碼，你也就得要損失這 10 分了。

  此外，由於本題是要求使用 class，所以你必須要寫 class 並且實做題目中要求的那兩個函數，然後適當地使用它們。如果屆時你負責評分的程式沒有做到這件事，請給他 0 分；如果你的程式沒做到這件事，你也就會損失本題這 10 分了。

第五題（bonus）

（20 分）承第二題，請修改第二題的函數

```
Command parseCommand(char command[]);
```

去處理含有 SELECT、FROM 和 WHERE 的 SQL query。這個函數會讀入代表一個 SQL query 的 `command` 字串，並將讀入的內容解析後存入 `Command` 這個 structure 之中。為此，或許你也得要把 `Command` 這個 structure 做一些修改。

現在一句 SQL query 包含有 SELECT、FROM 和 WHERE 這三個關鍵詞、SELECT 子句中的欄位名稱（一或多個）、FROM 子句中的資料表名稱（一個），以及 WHERE 子句中的欄位名稱（一個）、一個等號，以及一個值。如果有多個欄位名稱，則最後一個以外的欄位名稱後面會有一個逗點。如果要表現「所有欄位」，則在欄位名稱的部份只要寫出一個「*」即可。關鍵詞、欄位名稱與資料表名稱通稱為「token」，這是資訊科學領域常用的一個詞。整句 query 的最後面有一個分號。

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中，會有二行字串，第一行的字串代表 MySQL 的指令，最後面是一個分號，各 token 之間以一個空白字元隔開。指令中只會包含 SELECT、FROM 跟 WHERE 三種子句。第二行包含一個字串，是「SELECT」、「FROM」或「WHERE」。讀入這些資訊之後，請將第二行的該子句後的各 token 依序印出，但「WHERE」子句時不要印出等號。各 token 之間以一個空白字元隔開。如果沒有 WHERE 子句，且第二行為「WHERE」時，請輸出「NULL」。

舉例來說，如果輸入是

```
SELECT firstName, age FROM table1 WHERE lastName = Kung;
SELECT
```

則輸出應該是

```
firstName age
```

如果輸入是

```
SELECT firstName, age FROM table1 WHERE lastName = Kung;
FROM
```

則輸出應該是

```
table1
```

如果輸入是

```
SELECT firstName, age FROM table1 WHERE lastName = Kung;
WHERE
```

則輸出應該是

```
lastName Kung
```

如果輸入是

```
SELECT firstName, age FROM table1;
WHERE
```

則輸出應該是

```
NULL
```

這一題的全部 20 分會根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。針對這一題，你可以使用任何方法。