Stevens Institute of Technology
Castle Point on Hudson
Hoboken, NJ 07030

CS561 – Programming Assignment 2

Due Dates: 4/29/2016 (Fri.)

Objectives:

- To become familiar with the concept of *database application programming* and *query processing for complex OLAP/BI queries*.

Description:

"Simple Database Application Program #2" (`sdap2.cpp`)

- Generate reports based on the following queries:
  1. For each customer, product and state combination, compute (1) the customer's average sale of this product for the state, (2) the average sale of the product and the customer but for the other states and (3) the customer's average sale for the given state, but for the other products.
  2. For customer and product, compute the average sales before and after each quarter (e.g., for Q2, show average sales of Q1 and Q3), and only display the rows if the average sales increased from previous to next quarter. For "before" Q1 and "after" Q4, display <NULL> – display the rows regardless, if the average sales of previous or next quarter is <NULL>. The "YEAR" attribute is not considered for this query – for example, both Q1 of 2007 and Q1 of 2008 are considered Q1 regardless of the year.
  3. For customer and product, find the quarter by which time, 1/3 of the sales quantities have been purchased. Again for this query, the "YEAR" attribute is not considered. Another way to view this problem (problem #2 above) is to pretend all 500 rows of sales data are from the same year.

Again, the only SQL statement you're allowed to use for your program is:

```
select * from sales;
```

That is, **no `where`** clauses, **no aggregate functions** (e.g., `avg`, `sum`, `count`), etc.

And, you cannot store the 'sales' table in memory.

The following are sample report output (NOTE: the numbers shown below are not the actual aggregate values. You can write simple SQL queries to find the actual aggregate values).

Report #1:

```
CUSTOMER PRODUCT STATE CUST_AVG  OTHER_STATE_AVG OTHER_PROD_AVG
======== ======= ===== ========  =============== ==============
Helen    Bread   NY         243              268           1493
Emily    Milk    NJ        1426              478            926

. . . .
```

Report #2:

```
CUSTOMER PRODUCT  QUARTER  BEFORE_AVG  AFTER_AVG
======== =======  =======  ==========  =========
Bloom    Bread    Q1           <NULL>       2434
Sam      Milk     Q3              254        325
Dan      Apple    Q4               56     <NULL>
Helen    Pepsi    Q2                5          6

. . . .
```

Report #3:

```
CUSTOMER PRODUCT  1/3 QUANT PURCHASED BY
======== =======  ======================
Emily    Bread    Q2
Bloom    Milk     Q3
. . . .
```

Make sure that:

1. Character string data (e.g., customer name and product name) are left justified.
2. Numeric data (e.g., Maximum/minimum Sales Quantities) are right justified.
3. The Date fields are in the format of MM/DD/YYYY (i.e., 01/02/2002 instead of 1/1/2002).

Grading:

- (80 pts.) Logic/Correctness
- (20 pts.) Programming Style (e.g., comments, indentation, use of functions, etc.). You must include a program header, function header, etc. to clearly state what your program and functions are designed to do. Also for inline comments, please state clearly the purpose of those statements – for you as the programmer and to help others better understand your programming logic.

A program with compilation errors will earn no more than 50 points.

Sample Command Line

```
$ sdap2 [sales], where 'sales' is an optional argument for the table name.
```

Submission: Submit your source code (file) (with your name and CWID on it) on Canvas.

Please include a "README" file with detailed instructions on how to compile and run the code, especially if you are using a language other than C, C++ or Java.

In addition to the source code, submit **SQL queries** to generate the same output – you should use the SQL queries to check for the correctness of your program output.

Major Area	Item	Max	Deduct	Score	%	Total
Compilation	If fails, subtract …	**50**				
Logic	Query/Report #1	30				
	Query/Report #2	20				
	Query/Report #3	50				
	"Minimal Scan" implementation (YES/NO)					
	Total	**100**			**80%**	
Style	Header Comment	20				
	Function Comment	20				
	Line Comment	20				
	Indentation	10				
	Strings – Left Justified	15				
	Numbers – Right Justified	15				
	Total	**100**			**20%**	
Total		**100**			**100%**	