CS561 – Programming Assignment 2

Objectives:

- You will continue with evaluating simple report queries and produce the output. As with the assignment #1, you will also express the queries in SQL. The reports below are similar in nature with the reports from the assignment #1; however, there are two main differences between the two: (1) the new reports will require aggregation "outside" the groups (in assignment #1, all of the aggregates were computed for the rows within the groups); (2) some of the aggregates in the new reports will be computed based on other aggregates of the same reports – they are known as "dependent aggregates".

Description:

- Generate reports based on the following queries:

  1. For each *customer, product* and *month,* count the number of sales transactions that were between the *previous* and the *following* month's average sales quantities. For January and December, display <NULL> or 0.

  2. For *customer* and *product*, show the average sales *before, during* and *after* each *month* (e.g., for February, show average sales of January and March. For "before" January and "after" December, display <NULL>. The "YEAR" attribute is not considered for this query – for example, both January of 2017 and January of 2018 are considered January regardless of the year.

  3. For each *customer*, *product* and *state* combination, compute (1) the product's average sale of this customer for the state (i.e., the simple AVG for the group-by attributes – this is the easy part), (2) the average sale of the product and the state but for *all of the other customers*, (3) the customer's average sale for the given state, but for *all of the other products*, and (4) the customer's average sale for the given product, but for *all of the other states*.

  4. Find the *customers* with the top 3 highest quantities purchased in New Jersey (NJ). Show the customer's name, the quantity and product purchased, and the date they purchased it. If there are ties, show all – refer to the sample output below.

  5. For *customer* and *product*, find the *month* 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 (as in problem #2 above) is to pretend all 500 rows of sales data are from the same year.

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).

You are only allowed to standard SQL syntax covered in class – do not use any other functions other than the 5 aggregate functions (sum, count, avg, max & min); and use only simple syntax of 'agg(x)' – i.e., do not use features such as CASE statement inside (such features hide implicit JOINs).

Report #1:

```
CUSTOMER  PRODUCT  MONTH  SALES_COUNT_BETWEEN_AVGS
========  =======  =====  ========================
Claire    Apple      1                      <NULL>
Dan       Banana     3                          19

. . . .
```

Report #2:

```
CUSTOMER  PRODUCT   MONTH   BEFORE_AVG   DURING_AVG   AFTER_AVG
========  =======   =====   ==========   ==========   =========
Boo       Fish      1         <NULL>       284          434
Sam       Eggs      3         254          539          325

. . . .
```

Report #3:

```
CUSTOMER  PRODUCT  STATE  PROD_AVG  OTHER_CUST_AVG  OTHER_PROD_AVG  OTHER_STATE_AVG
========  =======  =====  ========  ==============  ==============  ================
Helen     Bread    NY       243          268             493              287
Emily     Milk     NJ       426          478             926              194

. . . .
```

Report #4:

```
CUSTOMER   QUANTITY   PRODUCT   DATE
========   ========   =======   ==========
Boo        982        Banana    2019-03-01
Boo        762        Fish      2016-12-23
Boo        679        Eggs      2017-02-27
Boo        679        Ice       2017-01-15
Chae       999        Butter    2016-11-11
Chae       990        Grapes    2017-08-07
Chae       901        Jellies   2017-07-31

. . . .
```

Report #5:

```
CUSTOMER  PRODUCT   1/3 PURCHASED BY MONTH
========  =======   ======================
Emily     Butter    2
Bloom     Soap      3

. . . .
```

Grading: **NOTE: A query with syntax errors will lose 50% of the points for the query.**

Submission: Submit all of the queries in a single TXT file – do NOT submit separate files for the queries or a ZIP file.

Please don't forget to include your name and CWID, and include a "README" section in the file if any special instructions are required.

I encourage you to discuss the "ideas" with your CA (rather than your classmates, esp, if you have any specific questions), but the final queries must be your own work. If I determine that your queries are copies of someone else's, both you and that someone else will be disciplined (you will receive 0 for the entire assignment) and possibly receive additional penalties for the course.