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

CS561 – Programming Assignment 2

Due Dates: 5/1/2020 (Fri.) for both Sections (A & B)

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 *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 2007 and January of 2008 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* and (3) the customer's average sale for the given state, but for *all of the other products*.
  4. 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:

```
PRODUCT  MONTH  SALES_COUNT_BETWEEN_AVGS
=======  =====  =======================
Cookies      1                  <NULL>
Yogurt       3                      19

. . . .
```

Report #2:

CUSTOMER	PRODUCT	MONTH	BEFORE_AVG	DURING_AVG	AFTER_AVG
Bloom	Coke	1	<NULL>	1539	2434
Sam	Eggs	3	254	539	325

. . . .

Report #3:

CUSTOMER	PRODUCT	STATE	PROD_AVG	OTHER_CUST_AVG	OTHER_PROD_AVG
Helen	Bread	NY	243	268	1493
Emily	Milk	NJ	1426	478	926

. . . .

Report #4:

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 TAs (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.