STEVENS

Department of Computer Science

1 Castle Point Terrace
Hoboken, NJ 07030
stevens.edu

CS 442/561 – Programming Assignment 1

Objectives

In this assignment, you will *express* "complex" OLAP queries in SQL. The key point of the exercise is to observe the complexity of expressing the type of such queries despite the relatively simple ideas of the queries themselves. Your mission (in addition to writing the SQL queries) is to consider the reasons for the complexity of the *expression* of these queries.

Description

Generate 5 separate reports based on the following queries (one report for query #1, one for query #2, one for query #3, one for query #4 and another for query #5):

1. For each *customer*, compute the *minimum* and *maximum* sales quantities along with the *corresponding products* (purchased), *dates* (i.e., dates of those minimum and maximum sales quantities) and the *states* in which the sale transactions took place. If there are >1 occurrences of the min or max, display all.

   For the same *customer*, compute the *average* sales quantity.

2. For each *year*, find the *"busiest"* and the *"slowest" month* (those months with the most and the least total sales quantities of products sold) and the corresponding *total sales quantities (i.e., SUMs)*.

3. For each *product*, find the "*most popular*" *month* (the month that the product was purchased the most) and the "*least popular*" *month* (the month that the product was purchased the least).

4. For each *customer* and *product* combination, show the *average sales quantities for the four seasons*, Spring, Summer, Fall and Winter in four separate columns – Spring being the 3 months of March, April and May; and Summer the next 3 months (June, July and August); and so on – ignore the YEAR component of the dates (i.e., 10/25/2016 is considered the same date as 10/25/2017, etc.). Additionally, compute the *average for the "whole" year* (again ignoring the YEAR component, meaning simply compute AVG) along with the *total quantities* (SUM) and the *counts* (COUNT).

5. For each *product*, output the *maximum sales quantities for each quarter in 4 separate columns*. Like the first report, display the *corresponding dates* (i.e., dates of those corresponding maximum sales quantities). Ignore the YEAR component of the dates (i.e., 10/25/2016 is considered the same date as 10/25/2017, etc.).

The following is a sample output – quantities displayed are for illustration only (not the actual values).

Report #1:

CUSTOMER	MIN_Q	MIN_PROD	MIN_DATE	ST	MAX_Q	MAX_PROD	MAX_DATE	ST	AVG_Q
Claire	12	Pepsi	01/01/2016	NJ	93	Apple	09/25/2020	NY	435
Sam	1	Milk	02/15/2019	NJ	259	Banana	03/23/2017	CT	56
Emily	2	Bread	07/01/2020	NY	87	Milk	02/02/2020	NJ	512
. . . .									

Report #2:

YEAR	BUSIEST_MONTH	BUSIEST_TOTAL_Q	SLOWEST_MONTH	SLOWEST_TOTAL_Q
2016	2	912382	1	7555
2016	3	2058236	4	10708
2017	11	990382	12	15734
. . . .				
2019	12	1274101	3	7089
2020	12	2846023	5	7089
. . . .				

Department of Computer Science
1 Castle Point Terrace
Hoboken, NJ 07030
stevens.edu

Report #3:

PRODUCT	MOST_FAV_MO	LEAST_FAV_MO
Apple	1	8
Pepsi	6	4
Ice	7	12

. . . .

Report #4:

CUSTOMER	PRODUCT	SPRING_AVG	SUMMER_AVG	FALL_AVG	WINTER_AVG	AVERAGE	TOTAL	COUNT
Dan	Ice	149	998	383	25	488	6344	13
Wally	Fish	39	72	142	35	79	632	8
Helen	Butter	43	981	239	87	281	2529	9

. . . .

Report #5:

PRODUCT	Q1_MAX	DATE	Q2_MAX	DATE	Q3_MAX	DATE	Q4_MAX	DATE
Apple	879	01/11/2017	609	04/24/2018	523	07/03/2019	964	10/14/2019
Grapes	792	03/31/2020	982	05/14/2020	881	08/23/2017	369	12/31/2020
Cherry	345	02/22/2019	659	03/10/2015	288	09/11/2016	261	11/15/2015

. . . .

Grading

NOTE:

1. **A query with syntax errors will lose 50% of the points for the query**.
2. For this course, you are only allowed to **use the syntax covered in class** (**any query using such syntactic features will result in 0 point**) – e.g., do not use aggregate functions other than the 5 (sum, count, avg, max & min); do not use the keywords such as *coalesce*, *limit*, *row_number*, etc. and '*case*' statement inside aggregate functions. Additionally, do not use any *algorithmic features such as 'if then', 'while', etc*.

   If you're unsure, please ask before using any syntactic features that are not covered in class.

Submission

Submit **one file** containing all the 5 queries on Canvas – Please include your name and CWID in the file. The file type must be "TXT".

Please include a "README" file if any special instructions are required.

You can discuss the "ideas" with your classmates or your friends, 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.

PostgreSQL is the only product allowed for the course – please do NOT use any other products, such as SQLite, Oracle, SQL Server, etc.