Project

The main part of the Project involves material from about analytic functions and the GROUP BY clause. You need to use Common Table Expressions (CTEs) to manage complexity of some SQL statements in this part. You should focus on basic query formulation skills with selection of tables and join conditions as well as specific material.

The main part of the Project uses the tables in the CPI Card Group case study. You will need to use many tables in the case study database for the main part.

Part 1: Queries involving Grouping and Analytic Functions

For the main part of this Project, you should understand business intelligence needs and relate the needs to the data warehouse tables. For business intelligence needs, you should read the associated document providing background about analytical query formulation. After you are clear about the database representation for a problem, you should then begin writing SQL statements. To help structure your query formulations, you should create basic statements without the analytic function elements. After your basic statements execute correctly, you can then revise them for analytic function elements.

Base Queries

The base queries (Table 1) involve two broad areas, revenues/costs and quality control. To facilitate reuse of base queries with analytic function requirements, you should use CTEs as indicated. For BQ2, BQ3, BQ5, and BQ6, you should have WITH statements with a CTE and a SELECT statement to retrieve the result of the CTE. The CTEs in these problems should be reusable in analytic queries. In some problems, a WITH statement may contain multiple CTEs.

Table 1: Base Query (BQ) Requirements

Query (Area)	**Result Columns**	**Comments**

BQ1: Location/Sales class summary for job quantity and amount (revenue/costs)	Location id, location name, sales class id, sales class description, year of job contract date, month of job contract date, base price of sales class, sum of quantity ordered, sum of job amount	Job amount defined as quantity ordered times unit price; combine with time dimension table on contract date FK
BQ2: Location invoice revenue summary (revenue/costs)	Job id, location id, location name, job unit price, job order quantity, year of contract date, month of contract date, sum of invoice amount, sum of invoice quantity	Need to combine some fact tables; Be careful about cross product operations from missing a join condition among fact tables; Contract year/month used to match revenues and costs; WITH statement with CTE and query.
BQ3: Location subjob cost summary (revenue/costs)	Job id, location id, location name, year of contract date, month of contract date, sum of labor cost, sum of material cost, sum of machine cost, sum of overhead costs, sum of total costs, sum of quantity produced, unit cost	Need to combine some fact tables; machine costs computed as machine hours times rate per hour for machine; Contract year/month used to match revenues and costs; Unit cost defined as sum of total costs / sum of quantity produced. WITH statement with CTE and query.
BQ4: Returns by location and sales class (quality control)	Location id, location name, sales class id, sales class description, year of invoice sent date, month of invoice sent date sum of quantity returned, sum of dollar amount of returns	Return quantity defined as quantity shipped minus quantity invoiced; condition for positive return quantity; use columns in the invoice fact table to calculate the unit price
BQ5: Last shipment delays involving date promised (quality control)	Job id, location id, location name, sales class id, sales class description, time id of the date promised for the job, time id of the last shipment date, order quantity in the job, sum of shipped quantity after job promised date, difference in business days between last shipment date and promised date	Condition for last shipment date later than job promised date; use computing difference in business days; WITH statement with CTE and query. See SELECT statement on page 4
BQ6: First shipment delays involving shipped by date (quality control)	Job id, Location id, location name, sales class id, sales class description, time id of the shipped by date of the job, time id of the first shipment date, difference in business days between first shipment date and contractual shipped by date	Condition for first shipment date later than job shipped by date; use computing difference in business days; Use WITH statement with CTE and query. See SELECT statement on page 4

A complexity in some base queries is the need to combine fact tables. Typically, data warehouse queries involve a fact table combined with many dimension tables. Because some base queries compare quantities associated with different business processes (such as job and invoice details), combining multiple fact tables is necessary. Although the data warehouse design does not enforce relationships among fact tables, you can still use these columns (*W_Sub_Job_F.Job_Id, W_Job_Shipment_F.Sub_Job_Id, W_Job_Shipment_F.Invoice_Id*, and *W_Lead_F.Job_Id*) to navigate among fact tables. If you combine fact tables, you must be careful not to omit join conditions among fact tables. Otherwise, you will have a cross product operation resulting in much excessive processing (think about the product $10{,}000^2$ rows).

The base queries for contractual delays (last shipment delays (BQ5) and first shipment delays (BQ6)) are complex. To help with formulation, you can create CTEs and use these CTEs in the FROM clauses. In the subquery for BQ5, the condition on Actual_Ship_Date and Date_Promised is highly restrictive as most jobs have all shipments completed before the promised date. The subquery for BQ6 does not have condition on Actual_Ship_Date and Ship_By_Date because many jobs have an actual ship date after the ship by date. Even if a job has a shipment date after the actual ship by date, it still can have other shipment dates before the ship by date.

To calculate the difference in business days in the base queries for contractual delays (BQ5 and BQ6), you can use the *getBusDaysDiff* function. You can use this function to calculate the difference in business days between two-time identifier values. After compiling this function, you can use it in a SELECT statement just like you use PostgreSQL built-in functions. As an alternative to this function, you can use CTEs to retrieve workdays used in the calculation of business day differences. You can use the SELECT statement as a CTE and then reference the CTE to calculate the difference in business days.

Queries involving Analytic Functions

The base queries can be used in many queries involving analytic functions. This Project involves a small subset of possible analytic queries using the base queries. In some analytic queries, you should extend base queries with analytic functions. In other analytic queries, you should use a CTE in the FROM clause instead of directly extending base queries.

To assist with formulation of analytic queries, you should create CTEs for base queries involving revenue summary, cost summary, date promised delays, and shipped by date delays. You can use these CTEs just like base tables in queries for analytic functions.

Analytic queries involving customer revenue trends

You should write two analytic queries extending the base query for job revenue trends as summarized in Table 2. Both analytic queries involve window comparisons.

Table 2: Analytic Query Requirements for Location Quantity Trends

Analytic Query	Analytic Functions	Other Columns	Notes
AQ1: Cumulative amount for locations	Cumulative sum of amount ordered; Restart cumulative sum by location name and year; Use contract month as the ordering criteria; cumulative sum involves current row and all preceding rows	Location name, contract year, contract month, sum of job amount ordered	Extends location/sales class summary (BQ1)
AQ2: Moving average of average amount ordered for locations	Moving average of average amount ordered; Restart moving average by location name; Use combination of contract year and month as the ordering criteria; moving average of current row and 11 preceding rows	Location name, contract year, contract month, average of job amount ordered	Extends location/sales class summary (BQ1)

Analytic queries involving revenue and cost summaries

These analytic queries (Table 3) involve computed columns using the results of base queries for revenues/costs. Profit involves sum of revenues from invoices minus sum of total costs from subjobs. Total costs are the sum of labor, material, machine, and overhead. The base query for cost summary calculates the sum of each cost component so total costs is the sum of the component costs. Profit margin is sum of profit divided by sum of revenues. Profit margin is a widely used business measure for comparing business performance across different kinds of products.

Table 3: Analytic Query Requirements for Location Revenue Trends

Analytic Query	**Analytic Functions**	**Other Columns**	**Notes**
AQ3: Rank locations by descending sum of annual profit	Rank in descending order of annual sum of profit; Restart ranks for each year	Location name, contract year, sum of profit	Use CTEs for BQ2 and BQ3 in the FROM clause; Sum of profit calculated as sum of invoice amount minus sum of total costs (labor, material, machine, and overhead)
AQ4: Rank locations by descending annual profit margin	Rank in descending order of annual profit margin; Restart ranks for each year	Location name, contract year, profit margin	Use CTEs for BQ2 and BQ3 in the FROM clause; Profit margin calculated as annual sum of profit divided by annual sum of invoice amount; See AQ3 for calculation of sum of profit
AQ5: Percent rank of job profit margins for locations	Percent rank of profit margin for jobs; Use all jobs so no restarting of percent ranks	Job id, location name, contract year, contract month, profit margin	Use CTEs for BQ2 and BQ3 in the FROM clause; See AQ4 for calculation of profit margin
AQ6: Top performers of percent rank of job profit margins for locations	Percent rank of profit margin for jobs; Use all jobs so no restarting of percent ranks	Job id, location name, contract year, contract month, profit margin	Refinement of AQ5 to show only top 5% of job profit margins; Use AQ5 in the FROM clause or make a CTE for AQ5 and use the new CTE in the FROM clause

These analytic queries are difficult because they involve two base queries. To simplify formulation, you should use CTEs in the FROM clause, not the base queries.

Analytic queries involving returns

The analytic queries (Table 4) for returns extend the base queries for returns. The focus on both queries is sales class, not location.

Table 4: Analytic Query Requirements for Return Summaries

Analytic Query	Analytic Functions	Other Columns	Notes
AQ7: Rank sales class by return quantities for each year	Rank in descending order of sum of return quantity; Restart ranks for each year of the date sent year	Sales class description, year of date sent, sum of return quantity	Extends return summary (BQ4)
AQ8: Ratio to report of return quantities for sales classes by year Remember, RATIO_TO_REPORT not supported in PostgreSQL but easy work around	Ratio to report of sum of return quantity; Restart ratios for each date sent year	Sales class description, year of date sent, sum of return quantity	Extends return summary (BQ4); order by year and sum of return quantity

For AQ8, PostgreSQL does not support the RATIO_TO_REPORT analytic function. To compensate for the lack of the RATIO_TO_REPORT function, you should use the query pattern presented in the class notes and textbook.

Analytic queries involving contractual delays

The analytic queries for contractual delays (Table 5) rank locations by descending values of delay measures. The promised date in a job indicates the date by which all shipments should occur. The shipped by date in a job indicates the date by which the first shipment should occur. To simplify formulation, you should use CTEs in the FROM clause, not the base queries. In AQ10, the on time rate is calculated as SUM(Quantity_Ordered - SumDelayShipQty) / SUM(Quantity_Ordered) assuming that the CTE for BQ5 contains the computed column SumDelayShipQty.

Table 5: Analytic Query Requirements for Contractual Delay Summaries

Analytic Query	**Analytic Functions**	**Other Columns**	**Notes**
AQ9: Rank locations by sum of business days delayed for the job shipped by date (first shipment)	Rank in descending order of sum of business days delayed; Use both ranking functions; Restart rankings on year of shipped by date	Location name, year of date shipped by, sum of difference in business days	Use CTE for first shipped delays (BQ6); CTE should only contain delayed jobs
AQ10: Rank locations by delay percentage for jobs delayed on the last shipment date	Rank in descending order of the on-time rate; Restart rankings on year of date promised	Location name, year of date promised, count of delayed jobs, sum of difference in business days	Use CTE for last shipment date delays (BQ5); CTE should only contain delayed jobs