Final Project

Submission

Your submission should include three items:

- A minimum of 5 files as described below
- DO NOT submit a .zip file or .rar file.
- Separate files are required.
- MAKE SURE in all files it is clear which question – and which part of which question – is being answered.

Reminders

- Make sure you use good coding standards for all .SQL files including stored procedures and triggers.
- Make sure you full test all code
- Make sure you deal with error handling

Tasks

1 – (20%) - Building a data warehouse.

- I am providing you a logical model (basically the layout of a database in the form of a picture). You can find the logical model below.
- You need to create this database called **dba625**
- We need to ensure the following with your new database:
    - Good choices in **data types**
    - Good choices with when to use **NOT NULL**
    - Good choices with **CONSTRAINTS**
    - Good choices on **DEFAULT** values
    - A CALCULATED column for REVENUE
        - REVENUE = PRODUCT PRICE x QUANTITY
        - PROFIT = PRODUCT (PRICE – COST) x QUANTITY
    - **Primary Key / Foreign Key** connections for the following:
        - Stores
        - Customers
        - Employees
        - Products
- Ingest a **minimum** of the following into the database tables
    - 10 stores
    - 10 employees
    - 10 products
    - 10 customers
    - 100 sales records
- You can use any form of data ingestion you like (load, import, insert)

2 – (20%) – Data Integrity

- Create a TRIGGER called **saleschk** that ensures when a sales record is INSERT into the SALES table that the EMPLOYEE associated with that sales has a “Sales” job. If the employee is not a sales person then EMPLOYEE should be set to some value which indicates not a seller.
- Create a TRIGGER called **pricechk** and **costchk** that ensures that PRODUCT COST is less than PRODUCT PRICE. This trigger should fire when there is any INSERT or UPDATE which impacts either the PRODUCT COST or PRODUCT PRICE.
- Create a TRIGGER called **manchk** that ensures that the STORE MANAGER match with an EMPLOYEE

3 – (20%) – Performance Enhancements

- Create an INDEX on the SALES table to help make sure queries searching for products have strong performance
- Create a SUMMARY TABLE which keeps track of accumulating REVENUE and PROFIT by month and month to date.
- COMPRESS your SALES table

4 – (20%) - Security

- Create a userid called **dba625fa** who is a financial analyst
- CREATE a VIEW for this financial analyst which contains the following columns:
    - PRODUCT (Sales table)
    - STORE (Sales table)
    - STORE MANAGER (Store table)
    - REVENUE (Sales table)
    - PROFIT (Sales table)
- GRANT the appropriate access so that the dba625fa user has access to read data from this VIEW but no access to do anything else in the database (no inserts, updates, deletes – or see any data that is not included in the VIEW)

5 – (20%) - Application Logic & Queries

- Build a stored procedure called **QRev** as follows:
    - Input: Product, Quarter, Year
    - Output: Revenue for that product in the requested quarter
- Write the following SQL statements
    - A query that shows who the top sales person is for the year based on revenue
    - A query that shows the top 5 selling products for the year ordered by revenue in decreasing order
    - A query which lists the top 5 customers for the year based on revenue
    - A query which lists the top 2 stores for the year based on revenue

6 – (10%) – BONUS:

- Choose one of the “Other Topics” and implement it with your database. You can choose any of the topics. I would suggest doing one of the following:
    - Shifting your deployment to a containerized deployment
    - Create an external tables for one of your tables
    - Set up and run a federated query
    - Set up and deploy a workload management policy
- It’s up to you though which you try.
- You’ll need to show a successful deployment of whatever you choose to earn the bonus marks
- You can absolutely get 100% on the final project without doing Q6.
- This is an opportunity to improve your overall mark in the course by being able to achieve extra marks on this final project.

Here is the logical model for your database:

Sales Table:

- PRODUCT NUMBER
- STORE NUMBER
- EMPLOYEE NUMBER
- CUSTOMER NUMBER
- QUANTITY
- REVENUE
- PROFIT
- DATE

Store Table:

- STORE NUMBER
- MANAGER EMPLOYEE NUMBER
- ADDRESS
- PHONE

Employee Table:

- EMPLOYEE NUMBER
- LAST NAME
- FIRST NAME
- JOB

Customer Table:

- CUSTOMER NUMBER
- LAST NAME
- FIRST NAME
- PHONE

Product Table:

- PRODUCT NUMBER

- PRODUCT NAME
- PRODUCT DESCRIPTION
- PRODUCT COST
- PRODUCT PRICE

What to hand in:

Filenames –

For each question above it is clearly outlined what you need to hand in. I should receive:

(i) A **FINQ1, FINQ2, FINQ3, FINQ4, FINQ5** files which shows all the commands you are running, complete testing, their output and successful completion.

(ii) Remember to use good coding standards (indentation, comments, error handling) for all .SQL files and application code.

(iii) Remember to test all code including error handling.

(iv) Remember to show all output

In the end – I should receive at least 1 or more files per question. Just make sure everything is clear.