Data Administration in Information Systems

Project

This project is based on real-world data provided by E-REDES, the main operator of the electricity distribution network in Portugal. For more information about where the data came from, and their contents, please refer to the E-REDES Open Data Portal.

In particular, we use two datasets from E-REDES:

- Monthly consumption by municipality
- Number of active energy contracts by meter type

For this project, the two datasets were loaded into an SQL Server database, which can be restored from a backup file (**ProjectDB.bak**).

Tasks

In this project, you are asked to perform the following tasks:

1. Implement a partition scheme for the *MonthlyConsumption* table, so that the records for each year are stored in their own partition.
   As a result of this task, provide a SQL script or a step-by-step guide with the necessary SQL instructions to create the partitions and load the data into those partitions.
   At the end, include a query to show the number of records that each partition contains.

2. For this task, restore the database to its initial state.
   The following query has been written to analyze the evolution of energy consumption over the years, for each parish in the municipality of *Lisboa*, using the month of June as reference.

```
SELECT [Parish], [Year], SUM([ActiveEnergy]) AS [ActiveEnergy]
FROM [Energy].[MonthlyConsumption]
WHERE [Municipality] = 'Lisboa'
  AND [Month] = '06'
GROUP BY [Parish], [Year]
ORDER BY [Parish], [Year]
```

   Perform a study of the Estimated Subtree Cost for every possible index that might be useful to optimize this query. Present the execution plan and the Estimated Subtree Cost obtained with each index. Also, include the SQL instructions needed to create and to drop each index.
   If you create an index that the system does not use, force the use of that index through a query hint, in order to measure the Estimated Subtree Cost.

3. For this task, restore the database to its initial state.
   The following query has been written to analyze the total energy consumption by parish, the total number of contracts by parish, and (based on those two) the average energy consumption per contract, in each parish.

```
SELECT [Energy].[DistrictMunicipalityParishCode],
       [Energy].[District],
       [Energy].[Municipality],
       [Energy].[Parish],
       [Energy].[ActiveEnergy],
       [Contracts].[NumberContracts],
       [Energy].[ActiveEnergy] / [Contracts].[NumberContracts] AS EnergyPerContract
```

```
FROM (SELECT [DistrictMunicipalityParishCode],
             [District],
             [Municipality],
             [Parish],
             SUM([ActiveEnergy]) AS [ActiveEnergy]
      FROM [Energy].[MonthlyConsumption]
      GROUP BY [DistrictMunicipalityParishCode],
               [District],
               [Municipality],
               [Parish]) AS [Energy],
     (SELECT [DistrictMunicipalityParishCode],
             [District],
             [Municipality],
             [Parish],
             SUM([NumberContracts]) AS [NumberContracts]
      FROM [Energy].[ActiveContracts]
      GROUP BY [DistrictMunicipalityParishCode],
               [District],
               [Municipality],
               [Parish]) AS [Contracts]
WHERE [Energy].[DistrictMunicipalityParishCode] =
      [Contracts].[DistrictMunicipalityParishCode]
ORDER BY [Energy].[District],
         [Energy].[Municipality],
         [Energy].[Parish]
```

Perform a study of the Estimated Subtree Cost for every possible join algorithm and group-by algorithm that this query could use in SQL Server. Present the SQL instructions, the execution plan, and the Estimated Subtree Cost obtained for each join and group-by algorithm.

4. To speed up the previous query, create a materialized view for each nested subquery, and perform a study of the Estimated Subtree Cost for each possible join algorithm.
   Present the SQL instructions required to create the materialized views, and the new results in terms of execution plan and Estimated Subtree Cost for each join algorithm.

5. For this task, restore the database to its initial state.
   Using the queries of tasks 2 and 3 as workload, run Database Engine Tuning Advisor until it produces a set of recommendations that include, among other things, the two materialized views created in task 4, and at least one index to optimize the query in task 2.
   Provide step-by-step instructions to produce those recommendations from the initial database. Apply the recommendations and measure the improvement in terms of Estimated Subtree Cost.

Report

Prepare a report to present and describe all the SQL code, queries, execution plans and results for each task. There is no specific template, and any text processor will do, as long as it has the capability of generating an A4-sized PDF document for submission on Fénix.

As you present the results (for example, an Estimated Subtree Cost for a certain query), you are welcome to comment on why that result is better or worse than other options. Your comments should reveal a good knowledge of the topics addressed in this course.

Carrying out the tasks in this project is only part of the goal. Understanding what is going on is the ultimate goal. If needed, you are encouraged to do some research on your own. If you end up using external resources, please cite them as bibliography or references at the end of the report.