Sistemas de Informação e Bases de Dados 2020/2021

Project Assignment - Part 3

The third project of the Databases discipline includes the development of SQL queries, integrity restrictions and the creation of a web application prototype and OLAP queries.
Use the supplied schemaPart3.sql file to create the database for this part.

1. Database Loading

Once created, the database should be consistently filled with the records required to ensure that all SQL queries, presented below, have a **non-empty result**. Record creation and the database loading can be carried out through whatever method you find more adequate (manually, Excel spreadsheet, SQL script, Python, or other).

2. Integrity Constraints

Integrity constraints related to primary keys and foreign keys should also be specified in each table.

Write the code to implement the following integrity constraints with the SQL procedural extensions (Stored Procedures and Triggers) in the table schema provided:

(IC1) For every transformer, **pv** must correspond to the voltage of the **busbar** identified by **pbbid**.

(IC2) For every transformer, **sv** must correspond to the voltage of the **busbar** identified by **sbbid**.

(IC5) For every analysis concerning a transformer, the **name, address** values cannot coincide with **sname, saddress** values of the substation where the transformer is located (i.e., **gpslat** and **gpslong** have the same values in transformer and substation).

The integrity constraints **that can be defined without resorting to procedural extensions** (Stored Procedures and Triggers), should they exist, must be implemented in a more appropriate way

3. View

Create a view to get the supervisors and the number of substations that each one of them supervises, without including supervisors that do not supervise any substation.

4. SQL

Present the most succinct SQL[1] query for each of the following questions. If appropriate, you can use the view created previously.

1. Who are the analysts that have analyzed every incident of element 'B-789'?
2. Who are the supervisors that do not supervise substations south of Rio Maior (Portugal) (Rio Maior coordinates: 39.336775, -8.936379 (cf. Google Maps)?
3. What are the elements with the smallest amount of incidents?
4. How many substations does each supervisor supervise? (include supervisors that do not supervise any at the moment)

5. Application Development

Create a web application using Python CGI scripts and HTML pages that allows users to:

a) Insert, list and remove bus bars, substations, transformers
b) Change the supervisor assigned to a substation
c) Register incidents for non-line elements and edit their description

The solution should prize security, preventing attacks by SQL INJECTION. Additionally, the **atomicity of related operations** in the database should be ensured.

6. Indexes

Present the SQL index(es) creating instructions to improve the querying times for each of the cases listed below.

Indicate, with proper justification, what type of index(es), over which attribute(s) and over which table(s) it would make sense to create, in order to speed up each query execution. Assume that the size of the tables exceeds the available memory by several orders of magnitude.

Assume that there are no indexes over the tables, aside from those implicit when declaring primary and foreign keys.

6.1 Return the number of transformers with a given primary voltage by locality:

```
SELECT locality, COUNT(*)
FROM transformer
   NATURAL JOIN substation
```

[1] You cannot use SQL instructions that are not part of the standard (such as the `LIMIT instruction`).

```
WHERE pv = <some_value>
GROUP BY locality
```

6.2 List all descriptions of line incidents that start with a given prefix within two points in time:

```
SELECT id, description
FROM incident
WHERE instant BETWEEN <ts1> AND <ts2>
AND description LIKE '<SOME_PATTERN>%'
```

7. Multidimensional Model

Create a star schema in the database with information concerning items and anomalies, with the following dimensions:

d_reporter(id_reporter, name, address)
d_time(id_time, day, week_day, week, month, trimester, year)
d_location(id_location, latitude, longitude, locality)
d_element(id_element, element_id, element_type)

f_incident(id_reporter, id_time, id_location, id_element, severity)

Write the required SQL instructions to define and load the star schema from the existing tables, loading the dimension tables first, followed by the fact table. Attributes **id_reporter**, **id_time**, **id_location** and **id_element** are surrogate keys.

8. Data Analytics Queries

Within the context of the star schema created in the previous question, write an SQL query that allows you to analyze the total number of anomalies reported by **severity**, **locality** and **week_day**. The submitted solution must use ROLLUP, CUBE, GROUPING SETS instructions, or the UNION of GROUP BY clauses.

Report

The project will be graded based on the report submitted and the discussion. The report should contain all answers to the items requested above. In the table below the points awarded to each portion of the work are listed.

Item	**Points**
SQL	5
Application	6
Integrity Constraints	2
Indexes	3
Multidimensional Model	2
ETL + Data analytics	2

The report should begin with a cover page with the title "**Database Project, Part 3**", the **name and number of the students, the contribution from each member in relative percentage, along with** the **effort (in hours)** that each member put into the project, the **group number**, the group **shift** and the lab teacher's name. Besides the cover page, the report should have, at most, **8 pages**.

Delivery

The submission in Fénix should be a **zip** file with the following structure:

reportGG.pdf (where GG is the group number)	The report in **pdf** where **GG** is the group number, containing an **explanation of the architecture of the web application with a link to a working version**, the **relations between the various files** and the **indexes**. You should not include the instructions used to populate the database. The report does not need to include the OLAP component. ⚠ Groups must make sure to have the application working online until after the discussion.
queries.sql	File with the SQL queries.
populate.sql	File to populate the database with the test data.
RI.sql	File to create the integrity constraints (triggers & stored procedures).

star_schema.sql	File to create the multidimensional schema.
etl.sql	File with the script to load the star schema.
olap.sql	File with OLAP queries
web/	Folder with the Python and HTML files.

The project must be delivered in ZIP formatted with the name `delivery-03-GG.zip`[2] (where **GG** is the group number), through Fénix until 23h59 of the delivery date.

Note: Penalties apply to groups that do not follow the delivery instructions. Evaluation elements that are not found in the prescribed as above **will not be taken into account for grading purposes**.

[2] ⚠ The file format must be exclusively ZIP or GZ. Other archive formats (such as RAR) will not be accepted.