Data Analysis and Integration

Project

Introduction

In this project, you will be working with the Sakila and sequelmovie databases. The Sakila database[1] was originally installed with MySQL for demonstration purposes. The database contains data about film rentals, stores, customers as well as data about films (e.g., actors, category, release year, etc). The sequelmovie database[2] contains data about movies, actors, directors, soundtracks among others.

In this project, you are asked to perform two main tasks:

1. Create an integrated view of the films data that exists in the Sakila database and in the sequelmovie database.
2. Create a data warehouse from the Sakila database, and write queries for analysis and reporting purposes.

The following sections describe in more detail what you must do in each of these tasks.

The Sakila and sequelmovie databases

The script **Sakila.sql** contains the SQL instructions needed to create and load the Sakila database in MySQL. The database schema is quite large, but in this project, you will be using only a subset of it. The file **Sakila.png** provides a detailed view of the database schema.

The script **sequelmovie.sql** contains the SQL instructions needed to create and load the sequelmovie database in MySQL. Analogously to the Sakila database, the sequelmovie schema is also large but you will be using a subset of it as well. The file **sequelmovie.png** contains the sequelmovie database schema.

In this project, we will be focusing mainly on data about **customers, rental payments, films, categories, actors** from the Sakila database and data about **movies, person, person role** from the sequelmovie database.

Execute the two SQL scripts to create and load the two databases in your MySQL instance.

Integrating the Sakila database and sequelmovie database

In the Sakila and sequelmovie databases, you will find data about films (movies in sequelmovie) and actors (person with an actor role in sequelmovie). Consider the following mediated schema:

```
all_films(film_id, title, description, length, rating, category)
all_actors(actor_id, first_name, last_name)
all_film_act(film_id, actor_id)
    film_id: FK(all_films)
    actor_id: FK(all_actors)
```

1) Present the schema matching between the sequelmovie database and this mediated schema. The schema matching can be presented in the form of a table or diagram (as in the slides for this course).

[1] https://dev.mysql.com/doc/sakila/en/
[2] https://github.com/ciaranRoche/mySQL-movie-db

2) Present the SQL views that define the schema mapping between the two data sources and the mediated schema. Define the SQL views using the base tables and execute the corresponding SQL instructions.

3) Present the SQL query over the mediated schema that returns the film titles and actor names of films that are scientific fiction (usually abbreviated to Sci-Fi), and whose length is higher than or equal to 60 minutes. Run the SQL query and present the results obtained. Taking into account the values of all_films.category, what can you say about the underlying data heterogeneity? Justify.

Profiling the source databases

4) Use the tool *datacleaner* that you have used in the lab classes to answer the following questions:
   a) What is the value distribution of the last name values of actors in the Sakila database?
   b) What are the minimum and maximum film rental duration values, defined for each film, in the Sakila database?
   c) What is the movie title with the longest description in the sequelmovie database?
   d) What is the date range of the movies in the sequelmovie database?

Present the screenshots of the *datacleaner* canvas and the output obtained.

Creating the data warehouse

In this project, you should create a data warehouse from the Sakila database.

The data warehouse should have a star schema with the following dimensions:
- D_Film with title and category name.
- D_Customer with last name, city and country (slowly-changing dimension).
- D_Time with day, month and year.

Use rental payment amount as measure.

5) Present the SQL instructions needed to create the data warehouse tables. The data warehouse should be called sakila_dw. Ensure that all tables have appropriate primary and/or foreign keys. Present the SQL instructions in text format, but formatted and indented in a way that makes it easy to read (as in steelwheels_dw.sql from the lab guides).

6) Use PDI to develop a set of transformations and a job to implement an Extract-Transform-Load (ETL) process that populates the data warehouse with data coming from the sakila database. Take a screenshot of each transformation/job, followed by a screenshot of the configuration window for each step. You can use Alt+PrintScreen to capture these windows.

Exploring the data warehouse with SQL/OLAP

7) Write the following queries in SQL:
   a) What is the total rental payment amount.
   b) What is the total rental payment amount by category name.
   c) What is the total rental payment amount by category name and by city.

8) Write a single SQL/OLAP query (SQL with OLAP extensions) whose result is the same as the union of the results of the previous queries. The query should run on MySQL 5.7.[3]

Exploring the data warehouse with Saiku and MDX

9) Use PSW to create an XML definition of the data cube. The XML file should be called sakila_dw.xml. Add the .xml file to the zip file that you must submit.

10) Use Saiku to visualize the result of the following queries:
    a) Rental amount by country.
    b) Rental amount by country and film category.
    c) Rental amount by country and film category and year.

    Present a screenshot of the results for each query.

11) Write the following query in MDX:

    The rental amount growth per month (include previous month) for each South America country existing in the database (i.e., Argentina, Bolivia, Brazil, Chile, Colombia, Ecuador, French Guiana, Paraguay, Peru, and Venezuela).

    Present also a screenshot of the result.

12) Use PRD to create a report based on the following query:

    Rental amount by film category, sorted by rental amount in decreasing order.

    The report should contain a list and a bar chart.
    Take two screenshots of the report in PRD: one in *Design* mode and another in *Preview* mode.

Submitting the project

After you complete the tasks above, use a word processor (e.g. LibreOffice or MS Word) to prepare a pdf document with the answers to the questions above. Add the xml file for question 9 and create a zip file.

At the top of the first page of the pdf file, you must write the number of your group, your name(s) and student number(s).

Submit the zip file in Fénix until December 6, 2019.

[3] See the SQL syntax supported by MySQL in: https://dev.mysql.com/doc/refman/5.7/en/