096224: Distributed Database Management

Background

For the benefit of this project assignment, you are called to help the cinema company "Distributed Theaters".

"Distributed Theaters" is the 2^{nd} largest cinema company in Israel and it holds 70 cinemas across the country. However, despite their size they still aim to surpass the number 1 cinema company in the country. In order to surpass the best cinema company in Israel, "Distributed Theaters" aims to re-design it's database in order to maximize their efficiency and increase sales.

You are requested to help "Distributed Theaters" to re-design it's database.

Cinemas and Sites

"Distributed Theaters" holds 70 cinemas across the country by the following : Haifa - 15 cinemas, Tel Aviv - 17 cinemas, Jerusalem - 22 cinemas, Eilat - 12 cinemas, Tiberias - 4 cinemas.

In addition, each city of the above has a single component of computation, you can refer it as a "Site" in the terminology of our course, i.e. there are 5 sites.

All sites are equal in terms of computational ability and you can assume that each can storage infinite amount of data.

Furthermore, each site is connected to another in a two-way direction, i.e. data can move from one site to another and vice versa.

Clients

Clients are willing to buy tickets for a movie they like and have the best time of their life. In order to buy a movie ticket, clients often use the Distributed Theaters web search system, which can display many details about movies and showtimes. Each client search can be mapped into a query.

Clients can query the system from 5 different locations in the country: Haifa, Jerusalem, Tel Aviv, Eilat and Gesher (Gesher is a kibbutz in the Beit She'an Valley in northeastern Israel, some says it's the best place in the entire world).

Note that a query of a client, might output an empty set of movies. However, as new movies come, such query might result with movies later on.

Data Set

The data in this project is based on *The Movies Dataset* from Kaggle. However, it is heavily tweaked and went through many modifications.

Users

This source is provided within *users.csv*.

- **user_id** - User's unique identifier.
- **user_location** - User's location (city).

Movies

This source is provided within *movies.csv*.

- **movie_id** - Movie's unique identifier.
- **genres** - The genres of the movie.
- **overview** - An overview on the movie's plot.
- **production_companies** - The production companies that were involved in the making of the movie.
- **production_countries** - The countries that the movie was produced it.
- **release_date** - The movie release date.
- **revenue** - The movie's revenue.
- **spoken_languages** - Languages in which the movie is available in.
- **tagline** - The movie's tagline
- **title** - The title of the movie.
- **cities** - The cities the the movies is available in.

Credits

This source is provided within *credits.csv*.

- **cast** - The cast of the movie that corresponds to the id field.
- **crew** - The crew of the movie that corresponds to the id field.
- **id** - Movie's identifier (corresponding to 'movie_id' in movies.csv)

Queries

This source is provided within *queries.csv.*

- **user_id** - User's unique identifier.
- **genres** - List of genres the user is interested in, the movie genre's should be in at least one of them.
- **lang** - List of languages, the movie should be filmed in at least one of them.
- **actors** - Lists of actors, such all actors should be in the movie.
- **director** - Movie's director.
- **cities** - List of cities, movie should be played in one of.
- **country** - List of countries, movie should be produced in at least one of.
- **from_realese_date** - Minimal release date of a movie that the user is interested in.
- **production_company** - List of movies productions, movie should be produced in at least one of.

Tickets

This source is provided within *tickets.csv.*

- **user_id** - User's unique identifier.
- **movie_id** - Movie's unique identifier.
- **number_of_tickets** - Number of tickets the user had bought.
- **city** - City where the show takes place.
- **cinema_id** - Cinema's unique identifier within a city.

Assignment

The assignment as a whole includes a process of loading data, analyzing and understanding them, drawing conclusions, properly designing a distributed database and implementing the design using spark. We will break down the task into several parts.

Extract and Transform (15 points)

Load data, preprocess it, apply transformations. Use Spark.
This part is preliminary to the analysis part and the implementation part.

Consider principals discussed in class: data formats, hierarchical data, (un)structured data, etc.

What to submit?

1. Python file (.py) containing all the process you did. File name: project1_part1_[ID1]_[ID2].py
2. pdf file (.pdf) containing your data store and transformations description, please describe your considerations and decisions thoroughly. File name: project1_part1_[ID1]_[ID2].pdf

Data Analysis (25 points)

Analyze the dataset and derive at least 4 insights. Use spark.
Insights should refer to points that will help design the database on the next section.

You should mainly consider the users behaviour in terms of tickets buying and the nature of the queries, but feel free to include additional aspects. A great opportunity to let your creativity flourish.

As mentioned earlier, a client's query might output an empty set of movies. However, as new movies come, such query might result with movies later on. Therefore, in your analysis you should still consider such queries and not ignore them.

Use visualizations where applicable (:

What to submit?

1. Jupyter notebook (.ipynb) containing your analysis and explanations (use Markdown cells to write text). File name: project1_part2_[ID1]_[ID2].ipynb
2. html file (.html) of the notebook. File name: project1_part2_[ID1]_[ID2].html

Design (45 points)

Design a distributed database based on your analysis part's insights.

You may use algorithms from the lectures. If you decide to do this, code the algorithms yourself and document your code. Note that you can also modify the algorithms.

What to submit?

1. pdf file (.pdf) containing your design description, please describe your considerations and decisions thoroughly. File name: project1_part3_[ID1]_[ID2].pdf

Deploy via Spark (15 points)

Implement your design via spark in python (pyspark). Your implemantation should be divided into five directories (Haifa, Jerusalem, Tel Aviv, Eilat and Tiberias), one for each site. Each directory containing the fragments of each site according to your design.

What to submit?

1. Python file (.py) containing the design implementation. File name: project1_part4_[ID1]_[ID2].py

[ID1] and [ID2] are the id-s of the students doing the project.
If the project is done in a group of 3 replace [ID1]_[ID2] with [ID1]_[ID2]_[ID3]

General Guidelines

- Use *Spark 3* and above.
- Your final submission should be a zip file (.zip) named project1_[ID1]_[ID2].zip.
  The zip file should contain 4 folders: part1, .., part4. Each contains the matching part's files.
- Write clean code and document functions when necessary.
- Questions related to the project will be answered in the forum, via Moodle, exclusively.
- Only one of the team members need to submit.
- Submission is due to June 2nd, any delay in submission will result in a reduction of 20 points from the final score of this part.
- This assignment is not straight-forward and requires creativity, take it your way and enjoy.

Good luck,
Course staff.