Final Exam

2659 – Data Curation for Business Analytics – T1 2324

October 19th, 2023

This 90-minutes exam consists of three parts (30% for 15 multiple choice questions in Python, 20% for 10 questions in SQL and 50% for pandas).
In Part I, you are expected to answer the Python and SQL questions on the Quiz module.
In Part II, you are expected to submit to the Assignment module your pandas solutions.
Points may be deducted if you do not comply with the following instructions:

- You only need to deliver this notebook file with solutions (notice that, a notebook file extension is filename.ipynb). Other types of submitted file will **NOT** be evaluated.
- Data files must **NOT** be submitted.
- The notebook should be delivered with the outputs already available (i.e., printed or visible).

Part I: SQL Task (Moodle Quiz)

Dataset Description

The Chinook dataset represents a digital media store, including tables for artists, albums, media tracks, invoices, and customers. As our old friend in the Data Curation course, it has helped us learned how to write SQL efficiently.
Now it is time to test our ability to apply SQL to extract various insightful information for business analytics again. In this exam, we mainly focus on these tables: **Invoice, InvoiceLine, Customer, Track, Genre**.
Please create your own Chinook database using DBeaver. If you have a problem with loading the database, try to delete the existing Chinook database in DBeaver, and create it again.
Please access the SQL questions and submit your solutions to the Final_Exam quiz module on Moodle.

Part II: Pandas Task (Moodle Assignment)

Dataset Description

"What movie should I watch this evening?" This perhaps is a question you would ask yourself very often. As for me—yes, and more than once. As such, from Netflix to Hulu, the need to build robust movie recommendation systems is extremely important given the huge demand for personalized content of modern consumers.

We are going to examine a dataset which provides non-commercial, personalized movie recommendations. This dataset describes 5-star rating from MovieLens. It contains ratings across movies created by users. More details about the contents and use of all these files follows.

> **Ratings Data File Structure (ratings_dc.csv)** All ratings are contained in the file ratings_dc.csv. Each line of this file after the header row represents one rating of one movie by one user, and has the following format: *userId, movieId, rating*.
>
> The lines within this file are ordered first by userId, then, within user, by movieId.
>
> Ratings are made on a 5-star scale, with half-star increments (0.5 stars - 5.0 stars).
>
> **Movies Data File Structure (movies_dc.csv)** Movie information is contained in the file movies_dc.csv. Each line of this file after the header row represents one movie, and has the following format: *movieId, title, genres, movie_description*

Questions 1 (10 points)

Load the movies_dc.csv data as a pandas dataframe. Fix the following problems:

- The "movieId" column is mistakenly encoded as "movieId_". Please revise the column name as movieId.
- The "movie_description" column has irrelevant values. Please delete this column.
- The "title" column contains each movie's release year. Please extract the year information from the "title" column and use it to generate a new column "year". (You can still keep the release year in the original "title" column.)
- Note that the data type of the new column "year" should be converted to **int**.

Question 2 (5 points)

Show the top 3 years with the highest number of movies.

Question 3 (5 points)

Load the **ratings_dc.csv** as a pandas dataframe. Show the mean, max, and min values of the "rating" column.

Question 4 (10 points)

- Use the rating dataframe to generate a dataframe that contains each movie's average rating and number of ratings.
- Then, you need to inner join the movie dataframe with the dataframe generated in the previous step to create a new dataframe, **movie_rating**, which has the following four column names: *movieId, title, genres, year, avg_rating, num_rating*. Each row presents information of one movie. "avg_rating" is the mean value of ratings of a movie; "num_rating" measures how many time a movie has been rated.

Question 5 (5 points)

Show the number of movies with the average ratings 4.0.

Question 6 (5 points)

Show the titles of movies with Top 2 number of ratings (NOT average ratings).

Question 7 (10 points)

In the movie_rating dataframe, with movie release years ranging from 1980 to 2018, please add a new column called "time_interval." Specifically, use year bins [1979, 1999, 2009, 2020] to categorize movies into three groups: "before 2000", "2000-2009", and "since 2010".

After creating this new column, display the count of movies in each time interval.

Question 8 (5 points, challenging and extra 5 bonus points)

Now you need to implement a recommender system using collaborative filtering method. This works simply as to recommend movies that "people who like this movie also like these movies".

For example, people who like to watch StarWars are very likely to watch Star Treks.

In order to do so, you need to find out users who like one movie (i.e., post a rating of 5), and count what are the movies these users also like, ranked by the number of likes.

Task: Show the recommended movie list with top 10 movies that users who like the *Forrest Gump (1994)* may also like.

Congratulations! You just build the first recommender system that worth 1 million dollars :D