**PROJECT-2\_INFO-5709**

**PART-1:**

For this project, we took a movie dataset from kaggle (<https://www.kaggle.com/datasets/victorsoeiro/netflix-tv-shows-and-movies?resource=download> ). The datasets are credits and titles.

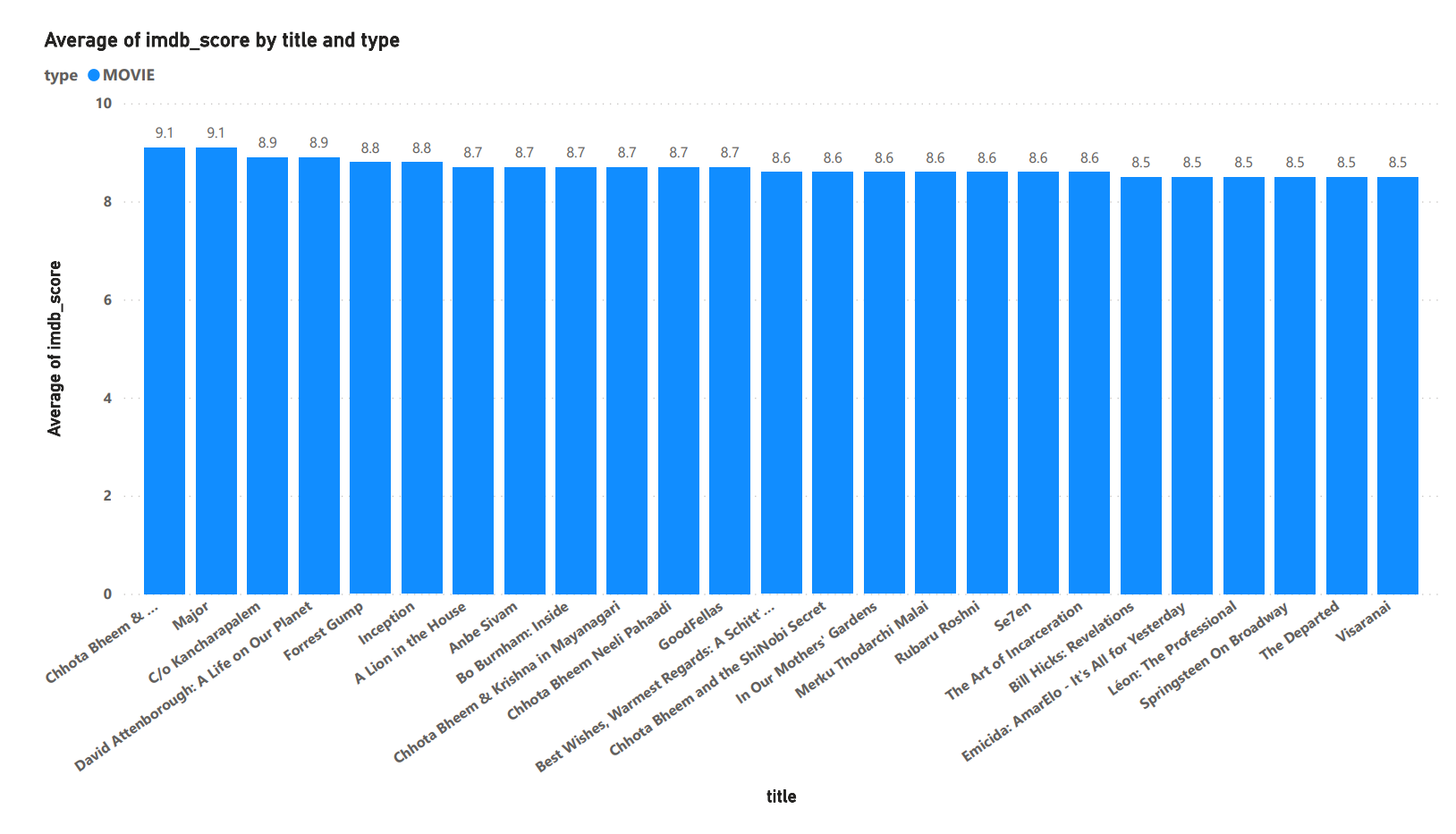
Nominal(N), Quantitative(Q), Ordinal(O).

The Credit dataset contains person\_id(Q), id(N), name(N), character(N), role(N).

Titles dataset contains id(N), title(N), type(N), description(N), release\_yr(Q), age\_certifications(O), runtime(Q), genre(N), production\_countries(N), seasons(Q), imdb\_score(Q).

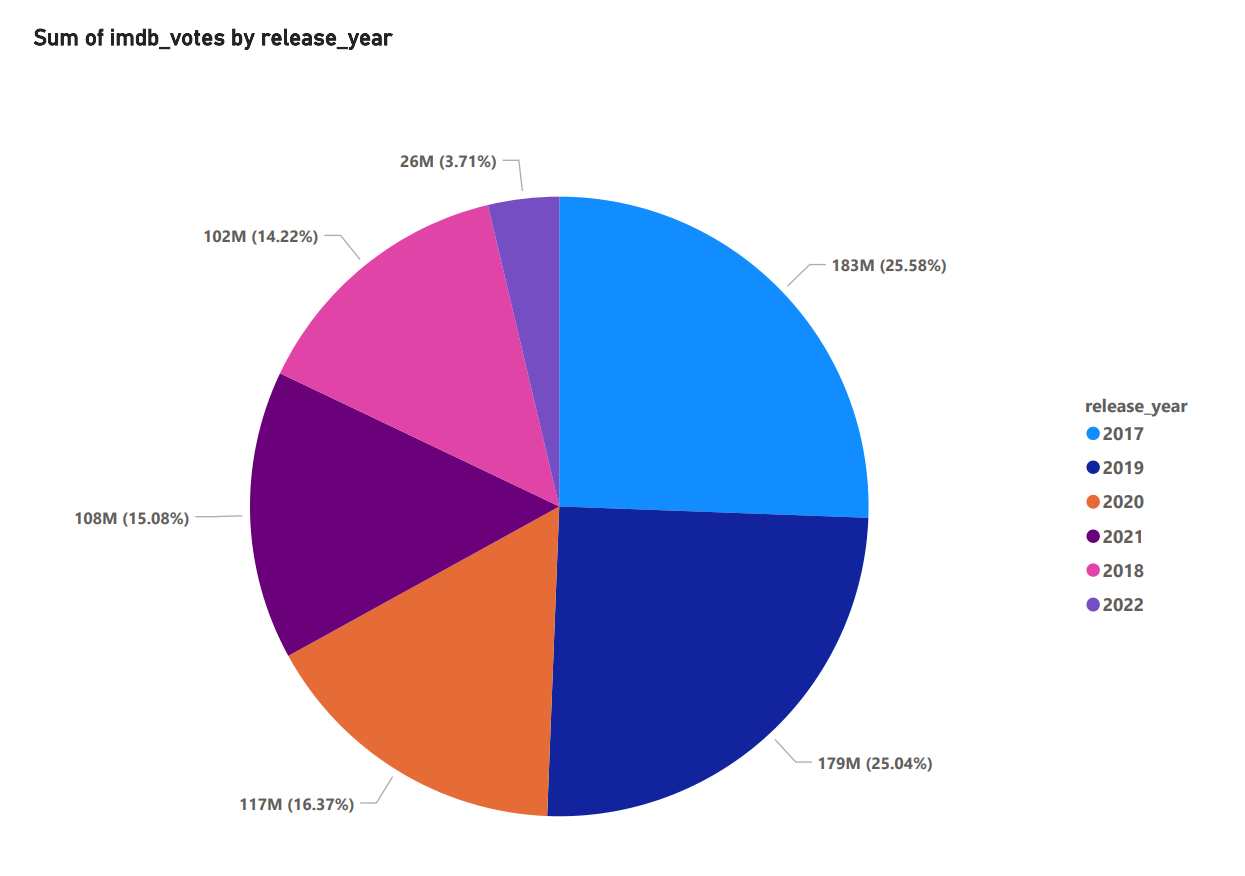
The three questions for the data, that I have analyzed are

1. Listing the movie titles of rating having greater than 8 and finding the best movie over them.

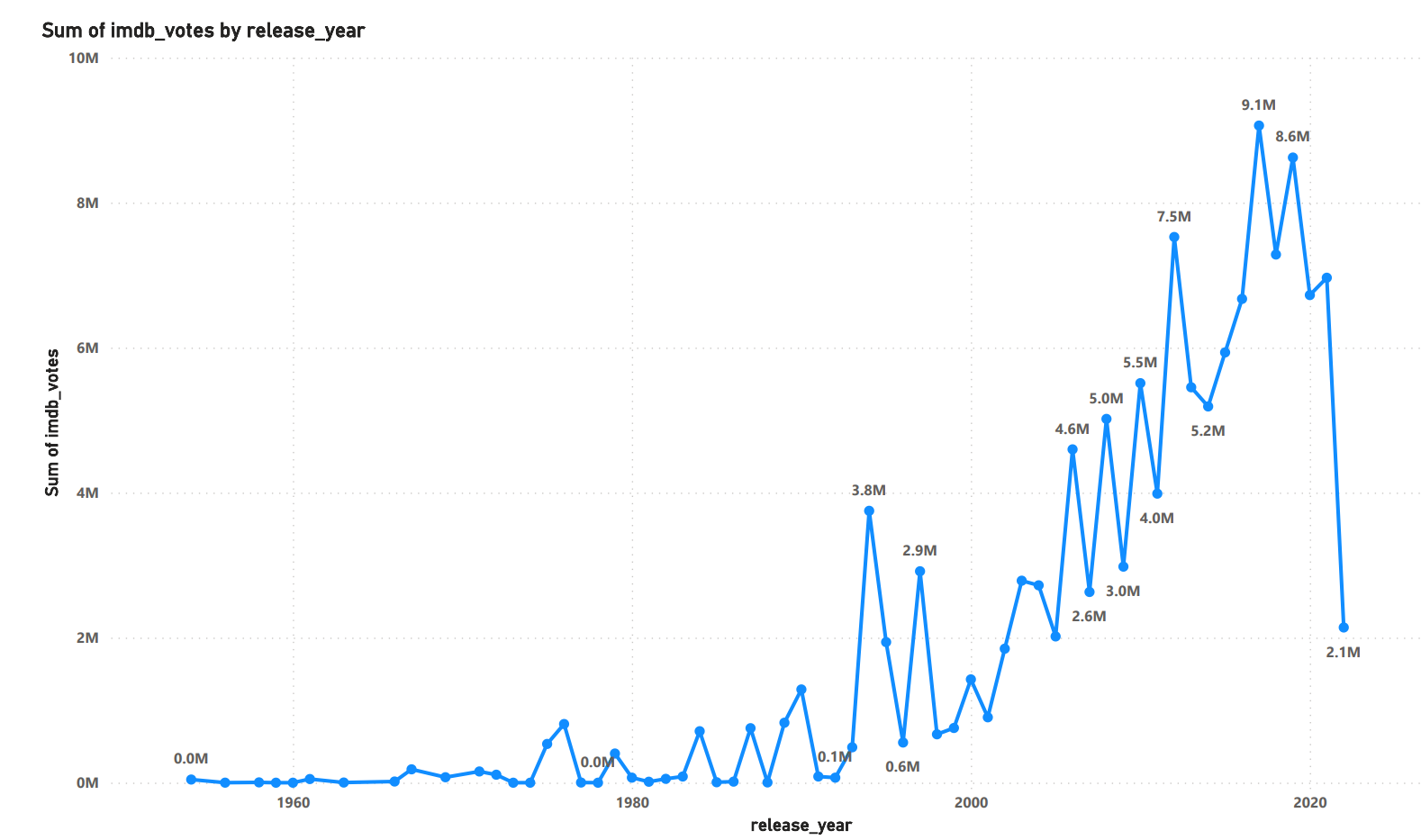


Here to know the answer, I have taken the average of my imdb score and rating greater than 8.I used stacked column chart on the scale of x-axis with movie titles and y-axis with average of imdb\_score on scale of 0 to 10 and filtering out only with the ratings greater than 8. The size of the image is 649x1164. The color of each stacked column used is blue. There are a total of 25 movies with ratings greater than 8. In which, the highest is Chhota Bheem with 9.1 rating.

1. To know, what % of votes have been submitted in the last 5 years.

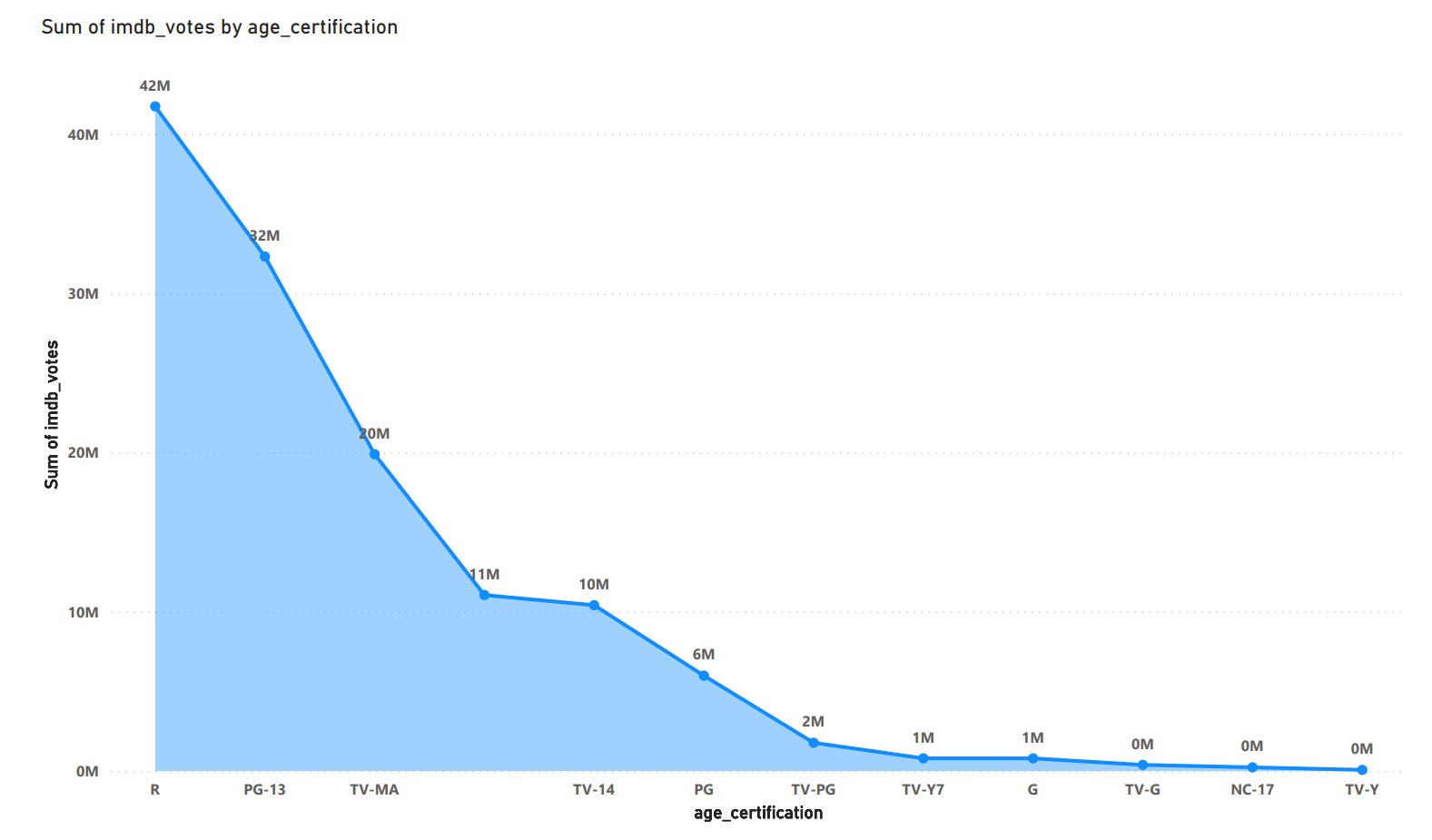


To know the answer for this, I have used pie chart visualization with release year as legend variable, Here the size of image is 695x887. Here, sky blue represents 2017(25.58%), dark blue is 2019(25.04%), orange is 2020 (16.37%), purple is 2021 (15.08%), pink is 2018(14.22%) and light purple is 2022(3.71%). The highest number of votes that were upcasted in 2017 over the last 5 years.



Here, for the same Question, I used another visual approach. Which is a Line chart. Where on x-axis it is year wise on a scale of 1960 to 2020, and on y-axis it is the sum of imdb votes on a scale of 0 to 10 million. It has a size of 889x912. As we can see, in 2016-2019, there are massive amounts of votes that have been upvoted.

1. Based on the age\_certification category, At what rate, the imdb voting have been upvoted?



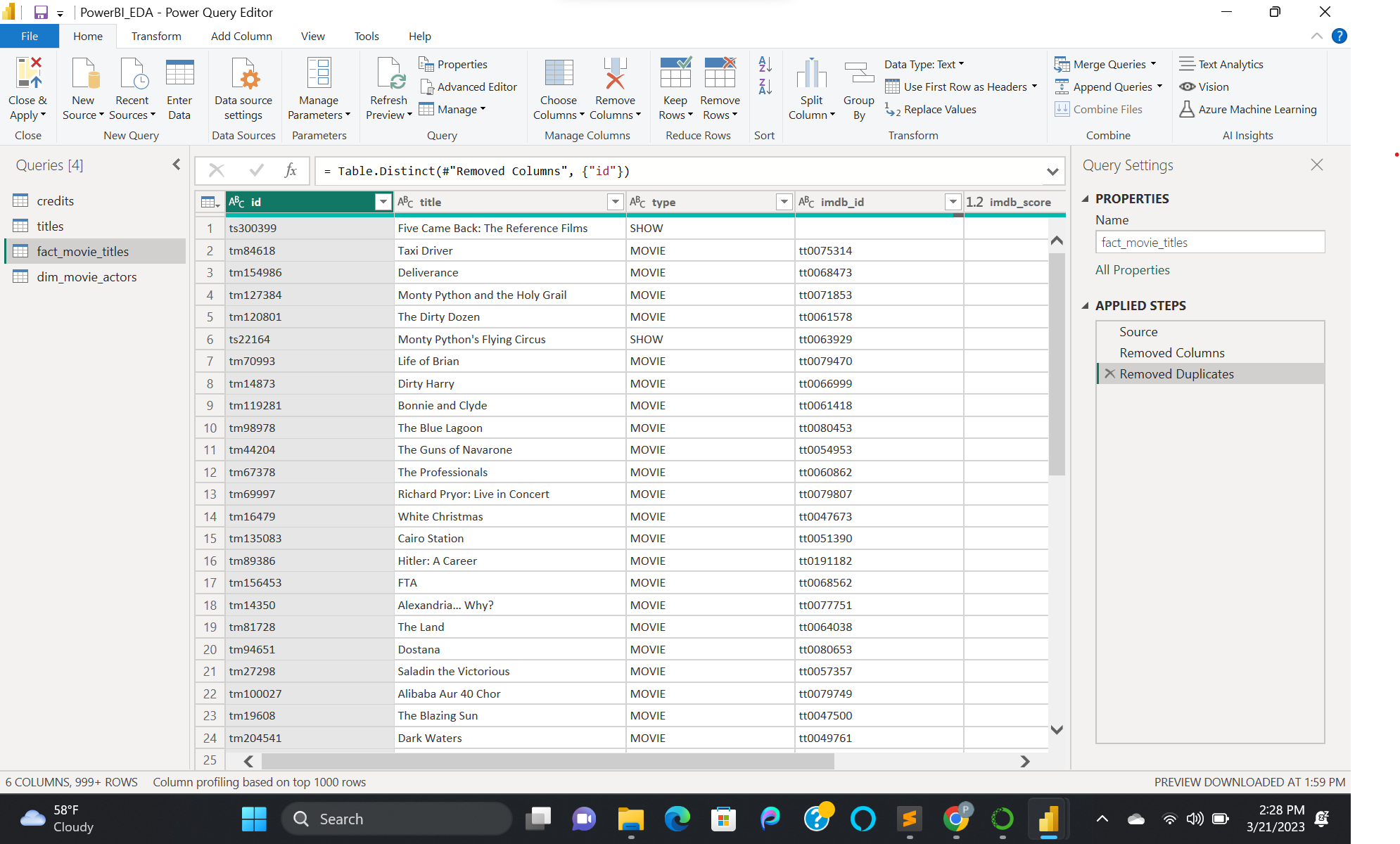
Here, I used the area chart, with age certification of different labels on x-axis and sum of imdb votings for each age\_cert on y-axis on a scale of 0 to 40M. The size of the image is 694x791. As we can see, R age certificated movies have got the most imdb votes.

**PART-2:**

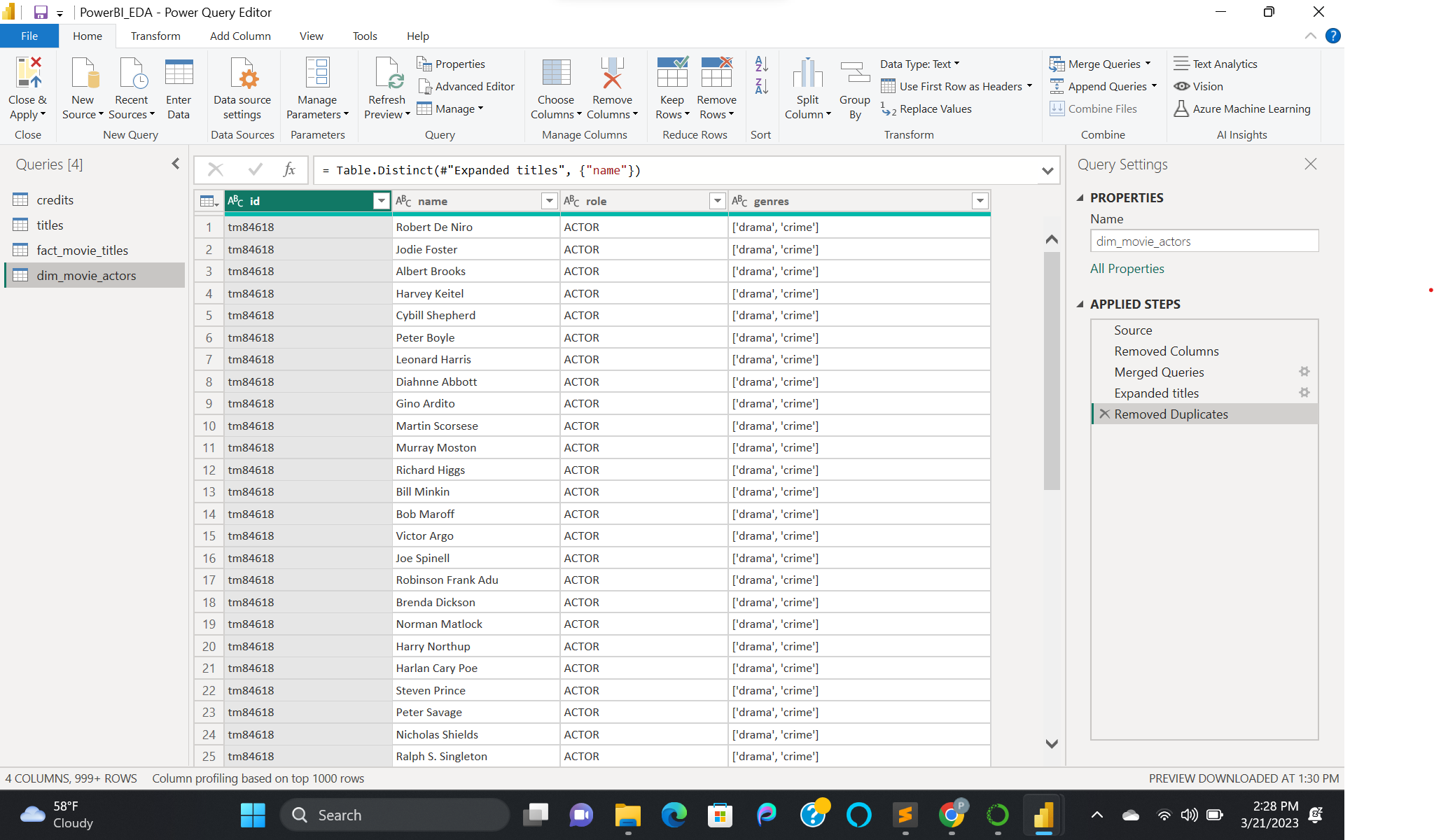
So, to display the visualizations for data understanding purposes, I have used PowerBI. It is interactive software used for data visualizations, which is developed by Microsoft by focusing on business intelligence. This tool is used to display the various kinds of visualizations, which will be helpful for decision making. Also helps to reveal patterns and relations between data points and to highlight interesting details. Also there are some issues related to this particular software. As it does not provide data cleaning solutions and might have some performance issues, limited sharing of the data and complex in nature.

**EXTRA CREDIT:**

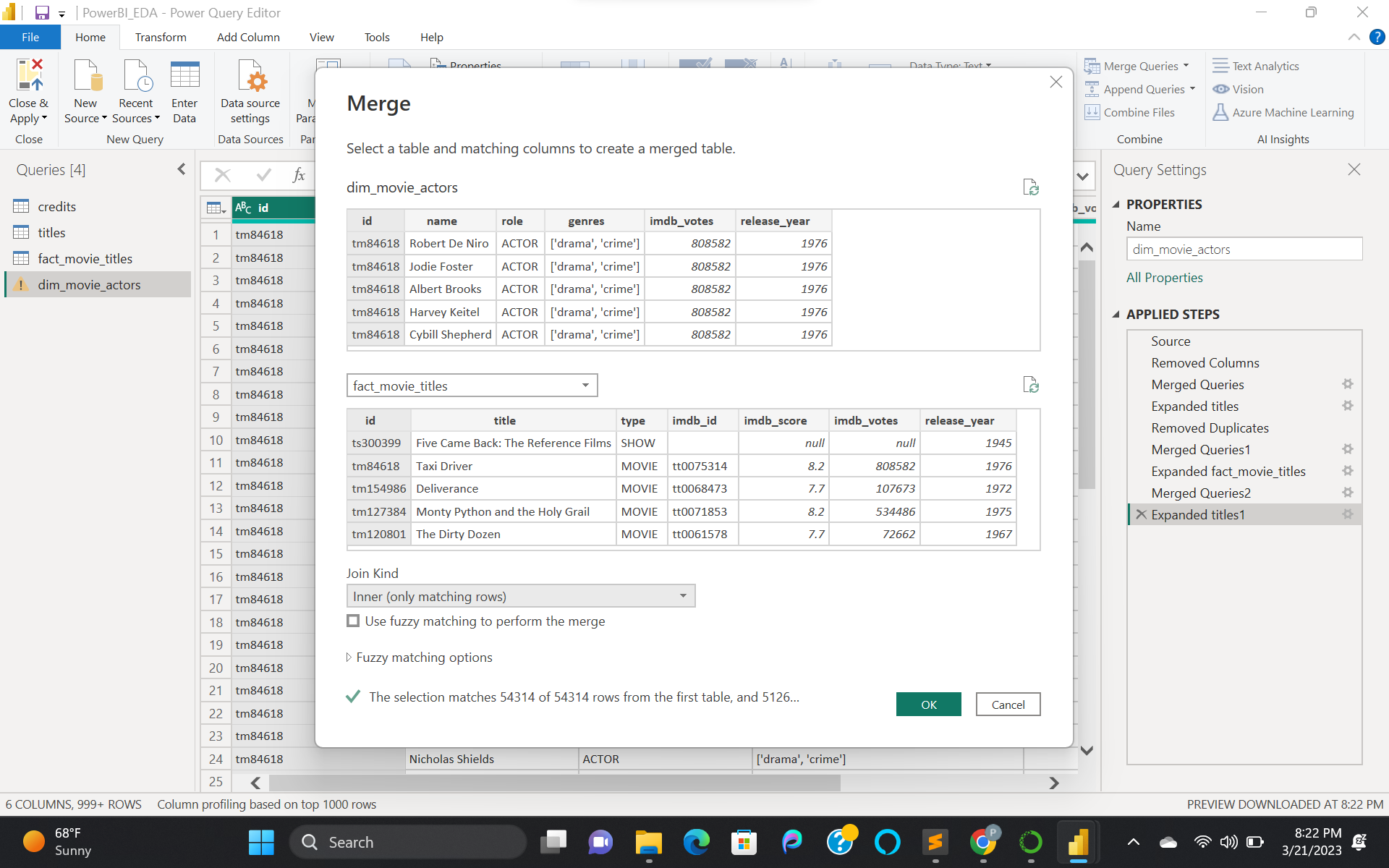
Here we have built the ETL process to transform the source data into star schema.



Here first, I went to transform data and then took reference of two tables and created new tables and then renamed them.



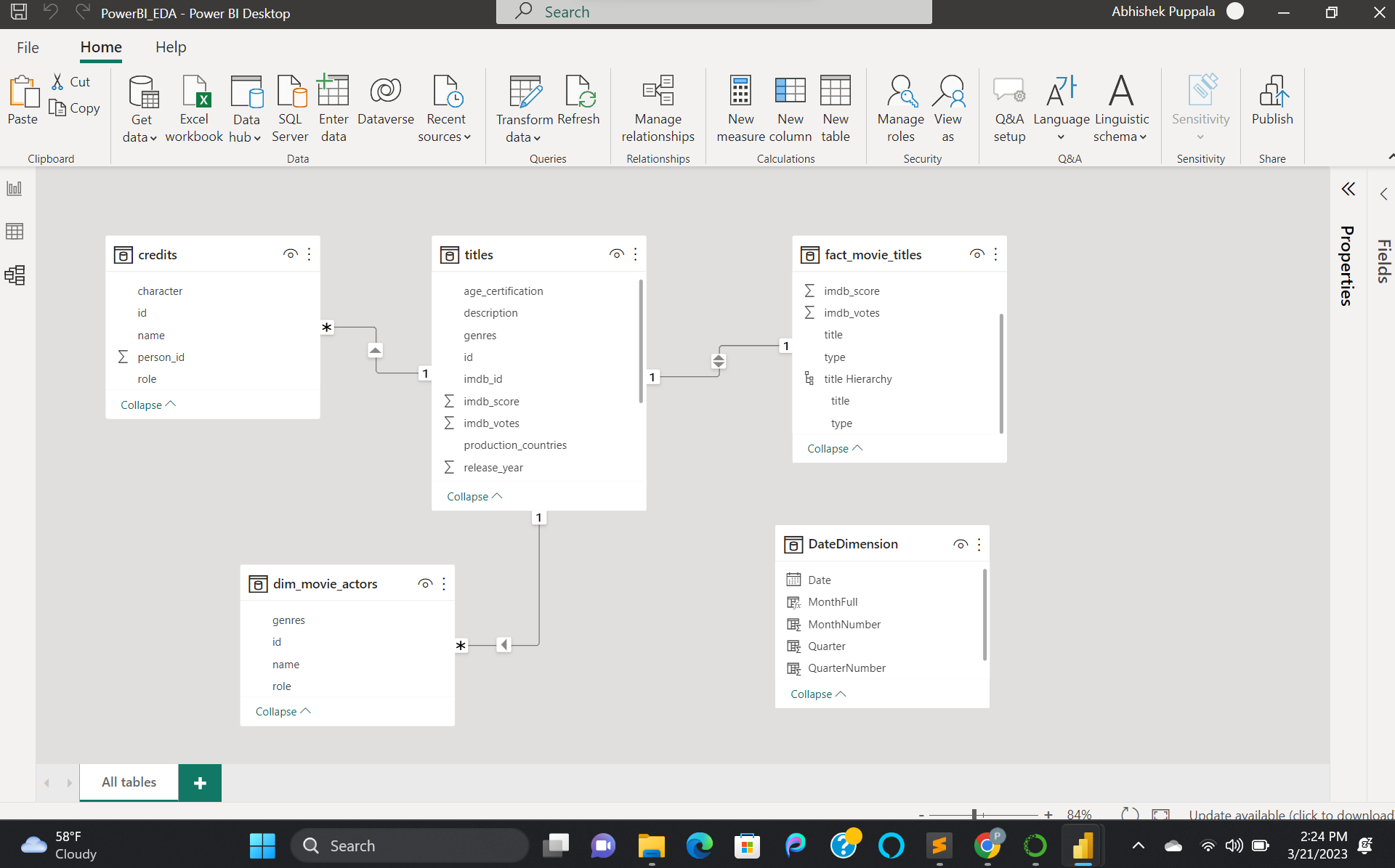
After renaming them, I have performed some of the data processing operations such as merging, filtering, and removing duplicates.



Here I have merged the titles data to dim\_credits using a merge query based on id and selecting titles data with join kind as inner\_join and then applied.

After applying the above operation, then I have filtered or selected the only needed data columns from titles table data. Then I have removed all duplicate rows from the table data. And also removed unnecessary columns.

We can see all the various operations performed under applied steps on the right side. After completing all the operations, then I have clicked on close and apply.



After all the preprocessing, the above star schema has been generated, which has dim\_movie\_actors and fact\_movie\_titles.

**ETL source codes** :

**Fact\_movie\_titles**:

removed cols = Table.RemoveColumns(Source,{"release\_year", "age\_certification", "runtime", "genres", "production\_countries", "seasons", "tmdb\_popularity", "tmdb\_score", "description"})

Removed\_dups = Table.Distinct(#"Removed Columns", {"id"})

**Dim\_movie\_actors**:

removed cols = Table.RemoveColumns(Source,{"person\_id", "character"})

Join inner = Table.NestedJoin(#"Removed Columns", {"id"}, titles, {"id"}, "titles", JoinKind.Inner)

expanded title = Table.ExpandTableColumn(#"Merged Queries", "titles", {"genres"}, {"genres"})

removed dups = Table.Distinct(#"Expanded titles", {"name"})