**Movie Database Project by Group 2**

**Introduction**

This dataset provides information regarding the top 250 movies listed in IMDB, including title, year, genre(s), staff, plot, language(s), country, MetaScore, ImdbVote, ImdbID, DVD start date, and Box Office Sales. What can we derive from the top 250 movies of all time? In this project, we will learn valuable information pertaining to what makes a movie great, including key correlations and similarities that can be discovered to understand what trends make movies shine. Potential users of the application include investors, studio executives, artists, and actors.

Data Description:

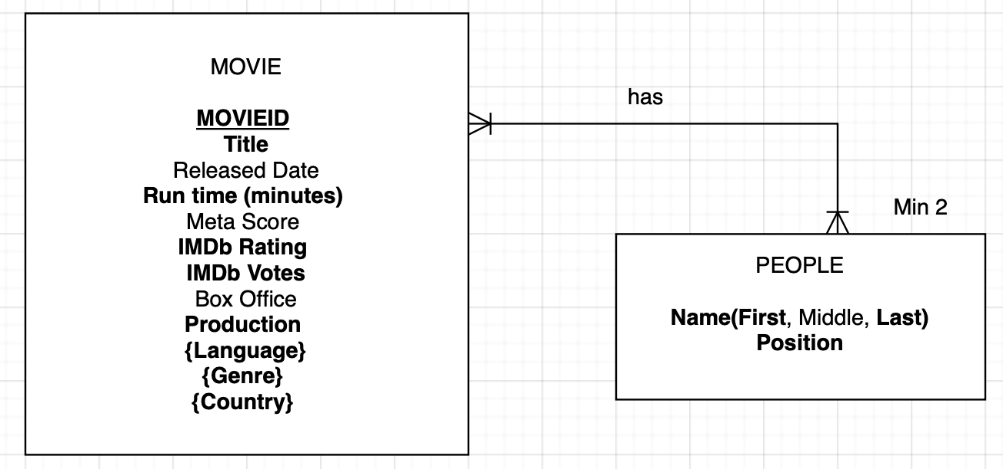
This project uses data updated in 2022 from Kaggle about the top 250 movies of all time according to ratings from IMDB (<https://www.kaggle.com/datasets/yehorkorzh/imdb-top-250-movies>). We have reduced the size of the data by reducing the columns that were deemed unnecessary for our proposed analysis such as type that had no differentiation in the data, year, as this was repetitive data, and awards, because the dataset does not combine nominations, wins, and other awards won correctly.

Data dictionary:

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| MovieID | Numeric | Unique Identifier for each individual movie |
| Title | Text | The name of the film produced |
| ReleasedDate | Numeric | The date the title was released |
| Runtime | Numeric | Length of Movie by minutes |
| Genre | Text | The categories/genres the title fits under |
| DirectorID | Numeric | Unique Identifier for each individual director |
| DirectorFirst | Text | The first name of the directors who created the movie |
| DirectorMiddle | Text | The middle name of a director if given |
| DirectorLast | Text | The last name of a director who created the movie |
| ActorFirst | Text | The first name of the actors who acted in the movie |
| ActorMiddle | Text | The middle name of the actors who acted in the movie |
| ActorLast | Text | The last name of the actors who acted in the movie |
| Suffix | Text | Letters after first name that help identify actor |
| ActorID | Text | Unique Identifier for each individual Actor |
| Language (1+) | Text | The languages that the movie was released in |
| Country (1+) | Text | The countries the movie was produced in |
| Meta Score | Numeric | The score provided by Meta critic about the overall rating of the movie |
| Imdb Rating | Numeric | The Movie rating out of 10 |
| imdbVote | Numeric | The number of people that ranked the movie |
| BoxOffice | Numeric | The Box office Revenue ($) if listed |
| Production | Text | The name of the production company |

The primary entity in the database is MOVIES, which is identified by Movie ID. Except for Released Date, Meta Score Box Office, the attributes are mandatory. The raw data contains many multivalued attributes that will have resolved as new tables and transposed. In the conceptual model, we have represented some of these attributes as weak entities because they have only partial identifiers. For example, many actors or directors can be in the cast of multiple movies. Or a production company will have filmed multiple movies.

**Figure 1: ERD**



*Figure 1: Entity Relationship Diagram*

Based on the data we have received from Kaggle, we have created an Entity Relationship Diagram with two entities. The primary entity in the database is MOVIES, with the unique identifier MovieID. Most of the attributes in MOVIE are required except for ReleasedDate, Meta Score, and Box Office which are indicated with no bolding. Additionally, the data contained many multivalued attributes which will be created into their own tables in the graphical relational schema. The MOVIE entity is connected to the PEOPLE entity which represents the actor’s and directors’ part of the data. This is a separate entity because a director and actor role could be the same person within a single movie but would need to be defined clearly by stating one or both positions within the movie. The two entities are connected with a mandatory one or many on both ends. However, there is a minimum of 2 on the PEOPLE entity as a movie because there must be at least one director and one actor, which can be the same person with two separate roles. Figure 1 displays the ERD for the data.

**Figure 2: Graphical Relational Schema**

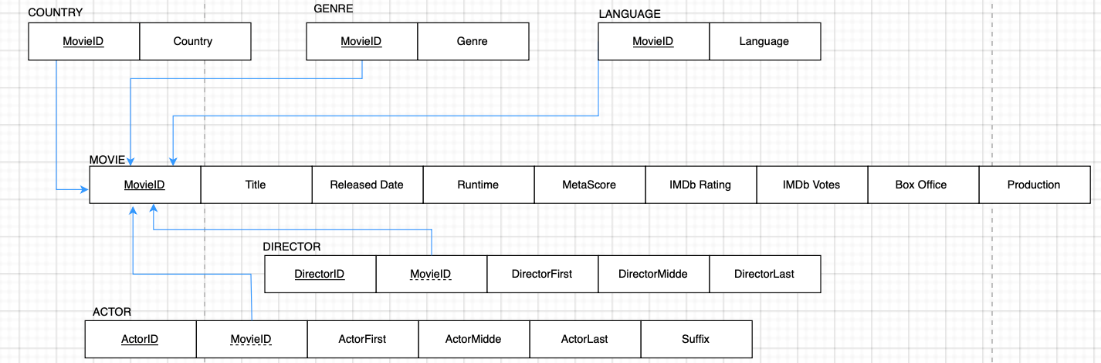
*Figure 2: Relational Schema Diagram*

Figure 2 displays the graphical relational schema of the database. There are 6 tables after resolving all the multivariable attributes. The MOVIES table acts as the parent, and each of the child tables connects to the MOVIES table via MovieID. MovieID may be either a primary key or a foreign key, depending on the table. The DIRECTOR and ACTOR tables have composite primary keys while the rest of the tables have a single primary key.

**Database Implementation**

To implement the database in APEX, we wrote CREATE TABLE commands for each table in the  
relational schema.

|  |
| --- |
| **MOVIE** As the parent table, MOVIEwas created and populated first  CREATE TABLE MOVIE (  MOVIEID VARCHAR(4) NOT NULL,  TITLE VARCHAR(50) NOT NULL,  YEAR NUMBER(4) NOT NULL,  RELEASED DATE NULL,  RUNTIME NUMBER(3) NOT NULL,  METASCORE NUMBER(3) NULL,  IMDBRATING NUMBER(1,1) NOT NULL,  IMDBVOTES NUMBER(7) NOT NULL,  IMDBID VARCHAR(15) NOT NULL,  DVD DATE NULL,  BOXOFFICE NUMBER(9) NULL,  PRODUCTION VARCHAR(50) NULL,  CONSTRAINT MOVIEID\_PK PRIMARY KEY (MOVIEID)  ); |
| **ACTORS**  CREATE TABLE ACTORS (  MOVIEID VARCHAR(4) NOT NULL,  ACTORFIRST VARCHAR(100) NOT NULL,  ACTORMIDDLE VARCHAR(100) NULL,  ACTORLAST VARCHAR(100) NOT NULL,  SUFFIX VARCHAR(100) NULL,  ACTORID VARCHAR(100) NOT NULL,  CONSTRAINT ACTORID\_PK PRIMARY KEY (ACTORID),  CONSTRAINT MOVIEID\_FK FOREIGN KEY (MOVIEID) REFERENCES MOVIE (MOVIEID)  ); |
| **DIRECTOR**  CREATE TABLE DIRECTOR (  DIRECTORID VARCHAR(4) NOT NULL,  MOVIEID VARCHAR2(4) NOT NULL,  DIRECTORFIRST VARCHAR(50) NOT NULL,  DIRECTORMIDDLE VARCHAR(50) NULL,  DIRECTORLAST VARCHAR(50) NOT NULL,  CONSTRAINT DIRECTOR\_PK PRIMARY KEY (DIRECTORID),  CONSTRAINT DIRECTOR\_FK FOREIGN KEY (MOVIEID) REFERENCES MOVIE (MOVIEID)  ); |
| **GENRE**  CREATE TABLE GENRE (  MOVIEID VARCHAR(4) NOT NULL,  GENRE VARCHAR(50) NOT NULL,  CONSTRAINT GENRE\_PK PRIMARY KEY (GENRE, MOVIEID),  CONSTRAINT GENRE\_FK FOREIGN KEY (MOVIEID) References MOVIE (MOVIEID)  ); |
| **LANGUAGE**  CREATE TABLE LANGUAGE (  MOVIEID VARCHAR(4) NOT NULL,  LANGUAGE VARCHAR2(50) NOT NULL,  CONSTRAINT LANGUAGE\_PK PRIMARY KEY (LANGUAGE, MOVIEID),  CONSTRAINT LANGUAGE\_FK FOREIGN KEY (MOVIEID) References MOVIES (MOVIEID)  ); |
| **COUNTRY**  CREATE TABLE COUNTRY (  MOVIEID VARCHAR(4) NOT NULL,  COUNTRY VARCHAR(25) NOT NULL,  CONSTRAINT COUNTRY\_PK PRIMARY KEY (COUNTRY, MOVIEID),  CONSTRAINT COUNTRY\_FK FOREIGN KEY (MOVIEID) References MOVIES (MOVIEID)  ); |

**Analysis:**

This analysis is intended to help investors, studio executives, actors, and anyone else with an interest in the movie industry determine what metrics result in the best movies.

**Q1: What are the most popular languages (besides English) used of the Top 500 IMDB movies?**

Our simple query will search for the languages used in the top 500 movies according to IMDB besides English. Our search has resulted in the top results in order of French, Spanish, German, Italian, and Latin. We decided to display these results as a pie chart for easy viewing.

|  |
| --- |
| SELECT COUNT(language) AS TOTALLANGUAGE, language  FROM LANGUAGE  WHERE language != 'English'  GROUP BY language  ORDER BY TOTALLANGUAGE DESC  FETCH FIRST 5 ROWS ONLY; |

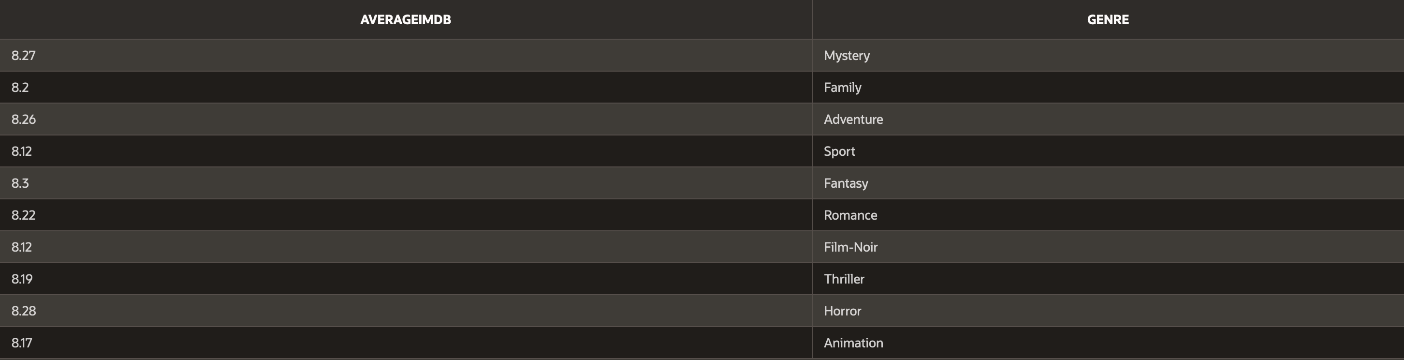
A black and grey striped background

Description automatically generated*Figure 3: Most Popular Language Table*

**Q2: What is the average IMDB rating per genre?**

Our join query will search for the average rating per genre within the top 500 movies. Of the 21 genres, they have very similar average ratings, so we ensured the numbers went to the hundred places. Of the 21 genres, they had average ratings of between 8.12 to 8.32. Lowest is both Film-Noir and Sport and Highest is both Crime + Music.

|  |
| --- |
| SELECT ROUND(AVG(MOVIE.IMDBRATING),2) AS AVERAGEIMDB, GENRE.GENRE  FROM MOVIE  JOIN GENRE ON GENRE.MOVIEID = MOVIE.MOVIEID  GROUP BY GENRE.GENRE; |
|  |

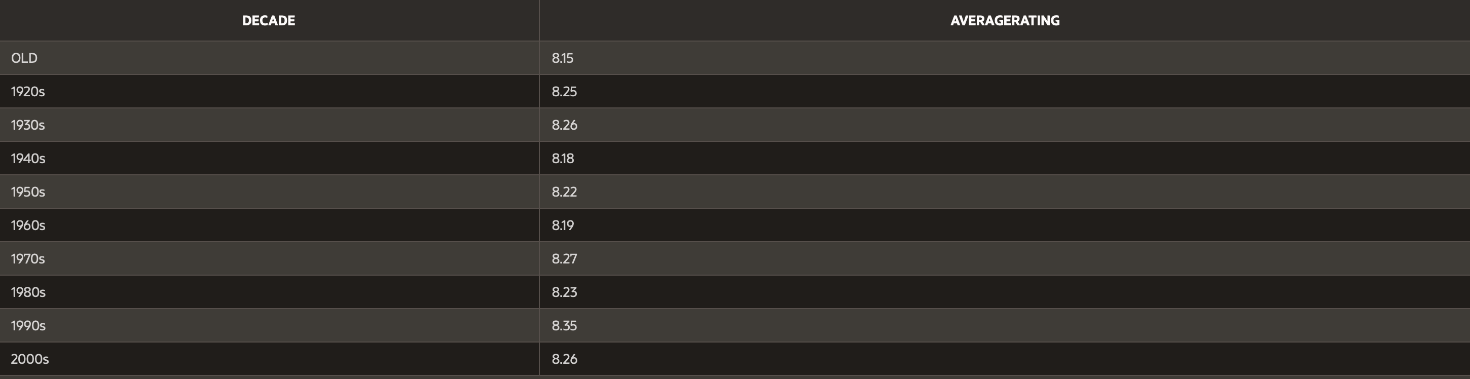


*Figure 4: Average IMDBrating per genre Table*

**Q3: What is the average IMDB rating per decade?**

Our advance query will search for the average rating per decade within the top 500 movies. Of the 21 decades, they have very similar average ratings, so we ensured the numbers went to the hundred places. Of the 7 decades, they had average ratings of between 8.18 to 8.35. The genre with the lowest average rating of 8.18 is before the 1910s and the genre with the highest average rating of 8.35 is the 1990s

|  |
| --- |
| END, DECADE;  SELECT  CASE  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 3 THEN '2020s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 13 THEN '2010s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 23 THEN '2000s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 33 THEN '1990s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 43 THEN '1980s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 53 THEN '1970s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 63 THEN '1960s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 73 THEN '1950s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 83 THEN '1940s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 93 THEN '1930s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 103 THEN '1920s'  ELSE 'OLD'  END AS DECADE,  ROUND(AVG(IMDBRATING),2) AS AVERAGERATING  FROM MOVIE  GROUP BY  CASE  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 3 THEN '2020s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 13 THEN '2010s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 23 THEN '2000s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 33 THEN '1990s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 43 THEN '1980s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 53 THEN '1970s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 63 THEN '1960s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 73 THEN '1950s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 83 THEN '1940s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 93 THEN '1930s'  WHEN EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM RELEASED) <= 103 THEN '1920s'  ELSE 'OLD'  END  ORDER BY  CASE  WHEN DECADE = 'OLD' THEN 0 -- Place 'OLD' first  ELSE 1  END, DECADE; |

*Figure 5: Average IMDB rating per decade Table*

**Q4: What is the most popular studio?**

Our query will search for the most popular studios among the top 500 IMDB Movies. Of the 500 movies there are 18 studios. By count the most popular is Warner Brothers with a movie count of 46 and the least popular is an 11-way tie with each of them producing 1 movie each. This was easily found with our interactive line chart.

|  |
| --- |
| SELECT PRODUCTION, COUNT(PRODUCTION) AS COUNTER  FROM MOVIE  GROUP BY PRODUCTION  ORDER BY COUNTER DESC; |

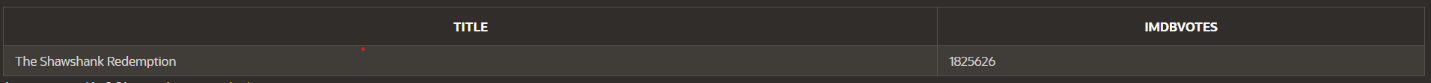
A black and grey striped background

Description automatically generated*Figure 6: Most popular Studio Table*

**Q5: Most voted on movie?**

Our query will search for the most voted on movie based on IMDBVotes. Our results have found that the most voted on movie is Shawshank Redemption with 1825626 votes.

|  |
| --- |
| SELECT title, IMDBVOTES  FROM MOVIE  WHERE IMDBVOTES = (SELECT MAX(IMDBVOTES) FROM MOVIE); |

*Figure 7: Most voted on movie Table*

**Web Design**

[Web Application:](https://apex.oracle.com/pls/apex/r/project312/movie-database-2023/home?session=112261482575628) This interactive link is an application to show you all our analysis of the given data and our research questions:

Home Page

The home page to the web application contains a brief description of the different categories of data that were included within the project. It also includes some examples of movies found within the data and a link to the original data pulled from Kaggle.

A screenshot of a computer

Description automatically generated*Figure 8: Movie Database 2023 Home Page*

Queries

The results of the queries are shown in the figures below with interactive charts for each question. Each page also includes a short description of what the chart is trying to represent from the data. The first chart showcases the most popular studios for movies within the top 250 with Warner Brothers coming out on top (Figure 9).

A screenshot of a computer

Description automatically generated

*Figure 9: Most Popular Studio Page*

Another question we wanted to answer was if there are certain genres that tend to have higher IMDB rating for movies within the top 250. This is shown through the bar chart below (Figure 10).

A screenshot of a graph

Description automatically generated*Figure 10: IMDb Rating Average by Genre*

Another curiosity we had is if there was a certain decade that excelled in having movies in the top 250. We did this by sorting the movies by the decade in which they were releases and taking the average IMDB rating for all the movies in that decade. This is shown with a line chart (Figure 11).

A graph on a screen

Description automatically generated*Figure 11: IMDb Rating Average by Decade*

One thing we were also looking into was the top languages in the Top 250 movies recorded. Out of this analysis, we found that French, German, Spanish, Italian and Latin were the leading languages with French being the most popular (Figure 12).

A screen shot of a computer

Description automatically generated

*Figure 12: Top Movie Languages*

A curiosity we had as a team was which movie was the most voted upon on the IMDB database, understanding what is one of the most popular movies to critics and consumers who choose to vote in the database. Through this we found that The Shawshank Redemption was the most voted movie with 1,825,626 votes (Figure 13).

A screenshot of a computer

Description automatically generated*Figure 13: Most Voted Movie*

Tables

A report was created for each database table with a search bar and filter option. Column headings along with table names so the data for each table is easily interpretable. Each table has a text box displaying a quick description of what data that table presents along with how many rows are in that table. (Figure 14).

A screenshot of a computer

Description automatically generated*Figure 14: Movie table*

A screenshot of a computer

Description automatically generated*Figure 15: Genre table* A screenshot of a computer

Description automatically generated*Figure 16: Actor Table*

A screenshot of a computer

Description automatically generated*Figure 17: Language table*

A screenshot of a computer

Description automatically generated*Figure 18: Country Table*

A screenshot of a computer

Description automatically generated*Figure 19: Director Table*