Name : Shraddha Pravin Sonawane

Roll no : ME-33

PRN-202401090035

---

1. Find the total number of movies.

```
import pandas as pd
import numpy as np

movies = pd.read_csv('movies.csv')

total_movies = movies['movieId'].nunique()
print(f"Total number of movies: {total_movies}")
```

---

2. Find the total number of users who rated movies.

```
ratings = pd.read_csv('ratings.csv')

total_users = ratings['userId'].nunique()
print(f"Total users: {total_users}")
```

---

3. Find the average rating across all movies.

```
avg_rating = np.mean(ratings['rating'])
print(f"Average Rating: {avg_rating}")
```

---

4. Find the maximum rating given.

```
max_rating = np.max(ratings['rating'])
print(f"Maximum Rating: {max_rating}")
```

---

5. Find the minimum rating given.

```
min_rating = np.min(ratings['rating'])
print(f"Minimum Rating: {min_rating}")
```

---

6. Find the most rated movie (highest number of ratings).

```
most_rated = ratings['movieId'].value_counts().idxmax()
movie_title = movies.loc[movies['movieId'] == most_rated, 'title'].values[0]
print(f"Most rated movie: {movie_title}")
```

---

7. List top 5 movies with the highest average ratings (with minimum 50 ratings).

```
movie_rating_count = ratings.groupby('movieId').size()
popular_movies = movie_rating_count[movie_rating_count >= 50].index

avg_ratings =
ratings[ratings['movieId'].isin(popular_movies)].groupby('movieId')['rating']
.mean()
top5_movies = avg_ratings.sort_values(ascending=False).head(5)

top5_titles = movies.set_index('movieId').loc[top5_movies.index]['title']
print(top5_titles)
```

---

8. Find how many unique genres exist.

```
genres_list = movies['genres'].str.split('|').sum()
unique_genres = np.unique(genres_list)
print(f"Unique genres: {unique_genres}")
```

---

9. Find the number of movies belonging to the 'Comedy' genre.

```
comedy_movies = movies[movies['genres'].str.contains('Comedy')]
print(f"Number of Comedy movies: {len(comedy_movies)}")
```

---

10. Find the distribution of ratings (how many 0.5 stars, 1.0 stars, etc.).

```
rating_distribution = ratings['rating'].value_counts().sort_index()
print(rating_distribution)
```

---

11. Find the number of movies that have been tagged at least once.

```
tags = pd.read_csv('tags.csv')

tagged_movies = tags['movieId'].nunique()
print(f"Movies with at least one tag: {tagged_movies}")
```

---

12. Find the movie with the maximum number of different tags.

```
tags_per_movie = tags.groupby('movieId')['tag'].nunique()
max_tags_movie = tags_per_movie.idxmax()
max_tags_title = movies.loc[movies['movieId'] == max_tags_movie,
'title'].values[0]
print(f"Movie with most unique tags: {max_tags_title}")
```

---

13. Find the oldest movie based on its title (assume year is in title like (1995)).

```
movies['year'] = movies['title'].str.extract(r'□(\d{4})□')
oldest_year = np.min(pd.to_numeric(movies['year'], errors='coerce'))
oldest_movies = movies[movies['year'].astype(float) == oldest_year]
print(oldest_movies[['title', 'year']])
```

---

14. Find the number of movies released after 2010.

```
movies['year'] = pd.to_numeric(movies['year'], errors='coerce')
movies_after_2010 = np.sum(movies['year'] > 2010)
print(f"Movies released after 2010: {movies_after_2010}")
```

---

15. Calculate average number of ratings per user.

```
avg_ratings_per_user = ratings.groupby('userId')['rating'].count().mean()
print(f"Average number of ratings per user: {avg_ratings_per_user}")
```

---

16. Top 5 most active users (who rated the most movies).

```
top5_users = ratings['userId'].value_counts().head(5)
print(top5_users)
```

---

17. Find the average rating given by each user.

```
user_avg_rating = ratings.groupby('userId')['rating'].mean()
print(user_avg_rating.head())
```

---

18. Find the movie with highest average rating (at least 100 ratings).

```
ratings_count = ratings.groupby('movieId').size()
movies_with_100 = ratings_count[ratings_count >= 100].index

avg_rating_100 =
ratings[ratings['movieId'].isin(movies_with_100)].groupby('movieId')['rating'
].mean()
top_movie = avg_rating_100.idxmax()

top_movie_title = movies.loc[movies['movieId'] == top_movie,
'title'].values[0]
print(f"Top rated (100+ ratings) movie: {top_movie_title}")
```

---

19. Find percentage of movies that are 'Action' genre.

```
action_movies = movies[movies['genres'].str.contains('Action')]
percentage_action = (len(action_movies) / total_movies) * 100
print(f"Percentage of Action movies: {percentage_action:.2f}%")
```

---

20. Find the average length of movie titles.

```
title_lengths = movies['title'].apply(lambda x: len(x))
avg_title_length = np.mean(title_lengths)
print(f"Average title length: {avg_title_length}")
```
