Python project

By: Chirag Malhotra

MovieLens 100k Dataset

Data Source: Kaggle
MovieLens data sets were collected by the GroupLens Research Project
at the University of Minnesota.

This data set consists of:

- 100,000 ratings (1-5) from 943 users on 1682 movies.
- Each user has rated at least 20 movies.
- Simple demographic info for the users (age, gender, occupation, zip)

The data was collected through the MovieLens web site
(movielens.umn.edu) during the seven-month period from September 19th,
1997 through April 22nd, 1998.

Importing the data

In [15]:
```
import pandas as pd
import numpy as np
columns = ['User_id', 'Movie_id', 'Rating', 'timestamp']
file_path = r"D:\AIDTM\term 2\EDA\archive\ml-100k\u.data"
movie_data = pd.read_csv(file_path, sep='\t', names=columns)
movie_data
```

Out[15]:

	User_id	Movie_id	Rating	timestamp
0	196	242	3	881250949
1	186	302	3	891717742
2	22	377	1	878887116
3	244	51	2	880606923
4	166	346	1	886397596
...	...	...	...	...
99995	880	476	3	880175444
99996	716	204	5	879795543
99997	276	1090	1	874795795
99998	13	225	2	882399156
99999	12	203	3	879959583

100000 rows × 4 columns

Part 1 : Data Cleaning

1). Identify and handle missing values in the dataset.Are there any rows or columns you should remove?Justify your decisions.

In [31]:
```
1 movie_data.isnull().sum()
```

Out[31]:
```
User_id                0
Movie_id               0
Rating                 0
timestamp              0
day_of_week            0
hour_of_day            0
time_of_day_category   0
dtype: int64
```

Interpretation:

- There are no missing values in the dataset, so there is no need to remove the rows or columns in the dataset.
- If there are more no of values missing in the columns than it is always advised to remove the column, cause that may skew the result.
- Skewed results may give us wrong insights, which may result in bad business decisions.

2). Explore the "timestamp" column.Can you extract any useful information like day of the week or time of day for rentals?Clean and format the timestamp data accordingly.

In [32]:
```
1 import pandas as pd
2 movie_data['timestamp'] = pd.to_datetime(movie_data['timestamp'], unit
3 movie_data['rental_day_of_week'] = movie_data['timestamp'].dt.day_name
4 movie_data['rental_time_of_day'] = movie_data['timestamp'].dt.time
5
6 print(movie_data[['timestamp', 'rental_day_of_week', 'rental_time_of_d
```

```
                   timestamp rental_day_of_week rental_time_of_day
0 1970-01-01 00:00:00.881250949           Thursday    00:00:00.881250
1 1970-01-01 00:00:00.891717742           Thursday    00:00:00.891717
2 1970-01-01 00:00:00.878887116           Thursday    00:00:00.878887
3 1970-01-01 00:00:00.880606923           Thursday    00:00:00.880606
4 1970-01-01 00:00:00.886397596           Thursday    00:00:00.886397
5 1970-01-01 00:00:00.884182806           Thursday    00:00:00.884182
6 1970-01-01 00:00:00.881171488           Thursday    00:00:00.881171
7 1970-01-01 00:00:00.891628467           Thursday    00:00:00.891628
8 1970-01-01 00:00:00.886324817           Thursday    00:00:00.886324
9 1970-01-01 00:00:00.883603013           Thursday    00:00:00.883603
```

Interpretation:

- With this timestamp column we have got the day of the week and also the time of the day.

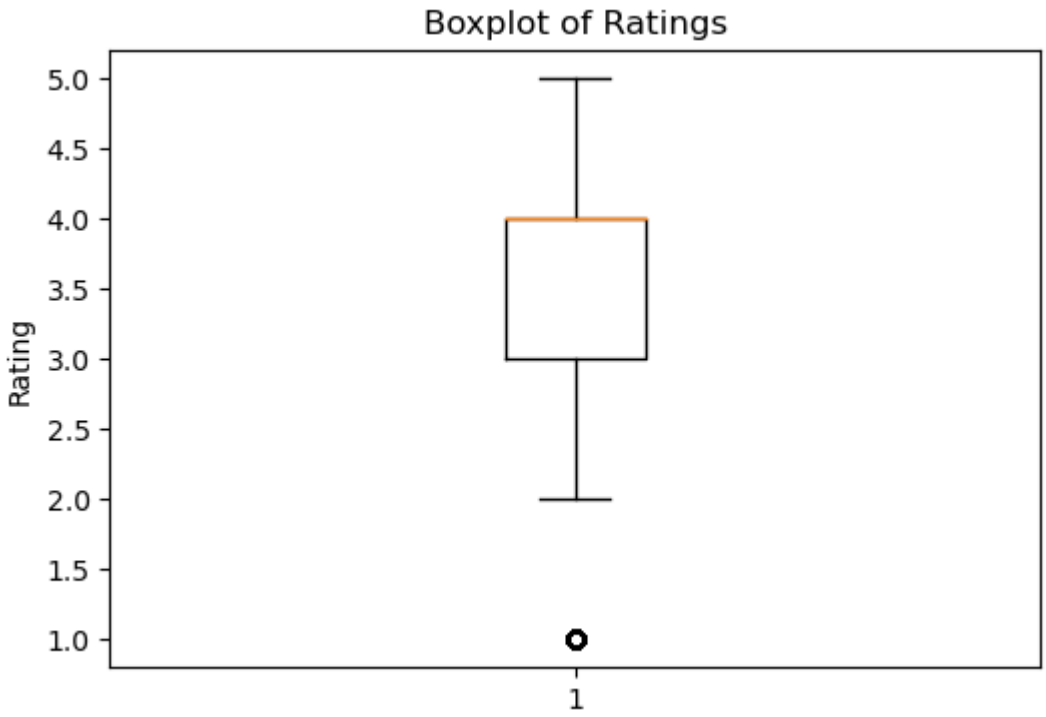
- By visualizing and analyzing the day of the week and time of day for rentals, we can know the trends that can be useful for decision-making, such as optimizing rental operations, identifying popular time slots, or adjusting pricing strategies based on demand during certain hours or days

3). Look for outliers and inconsistencies in the "rating" column.How would you handle outliers?Justify your chosen method.

In [34]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

print(movie_data['Rating'].describe())
plt.figure(figsize=(6, 4))
plt.boxplot(movie_data['Rating'])
plt.title('Boxplot of Ratings')
plt.ylabel('Rating')
plt.show()
```

```
count    100000.000000
mean          3.529860
std           1.125674
min           1.000000
25%           3.000000
50%           4.000000
75%           4.000000
max           5.000000
Name: Rating, dtype: float64
```



Interpretation:

- The ratings are skewed to the right, meaning that most of the ratings are higher than the median.
- This indicates that the majority of the reviewers were satisfied with the product or service being rated.

- The outlier at a rating of 1 is an extreme value that deviates significantly from the rest of the ratings.

4). Check for duplicate entries in the dataset.Remove duplicates if necessary.

In [36]:
```
duplicate_rows = movie_data.duplicated()
print("Number of duplicate rows:", duplicate_rows.sum())

duplicate_data = movie_data[duplicate_rows]
print("Duplicate rows:")
print(duplicate_data)
```

```
Number of duplicate rows: 0
Duplicate rows:
Empty DataFrame
Columns: [User_id, Movie_id, Rating, timestamp, day_of_week, hour_of_day,
time_of_day_category, rental_day_of_week, rental_time_of_day]
Index: []
```

Interpretation:

- There are no duplicate values int the data so there is nothing to remove.
- Duplicate values can increase the size and complexity of the data, making it harder to store, process, and analyze.
- Duplicate values can skew the distribution and statistics of the data, leading to inaccurate and misleading results.

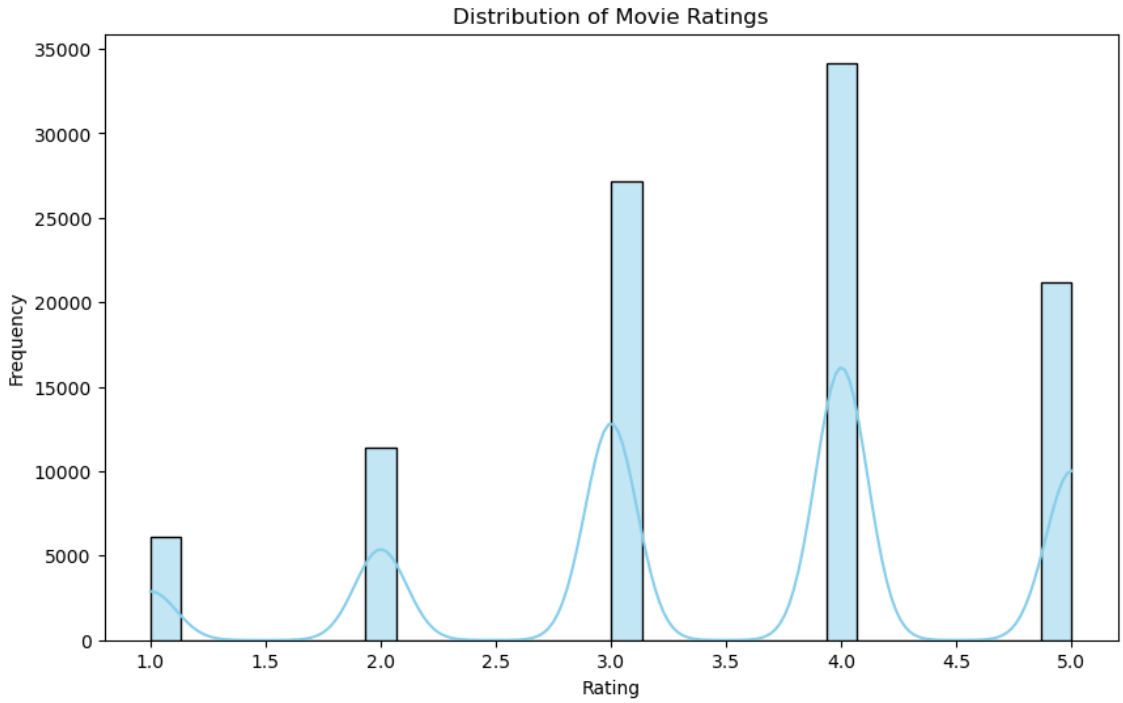
Part - 2 : Data Visualization

1). Create a histogram or boxplot to visualize the distribution of movie ratings.Are there any interesting trends?

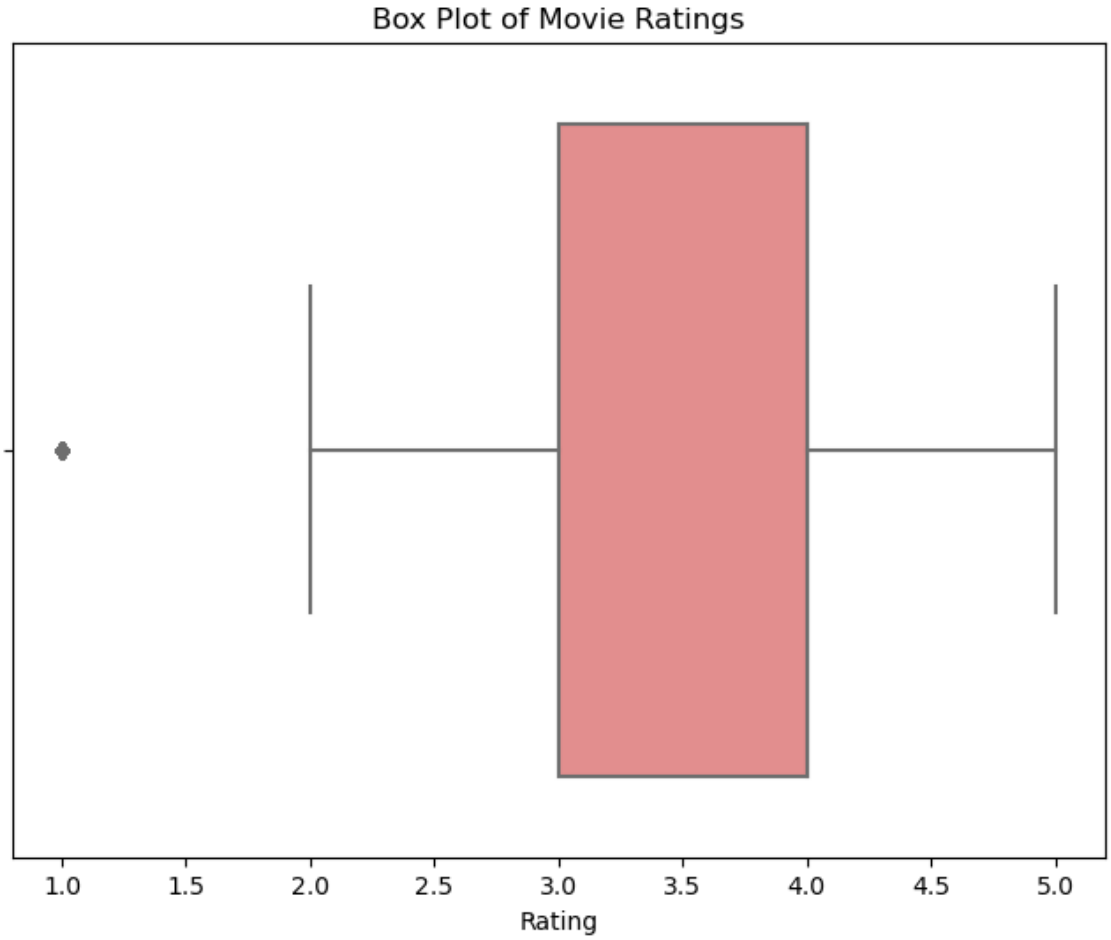
In [38]:
```
plt.figure(figsize=(10, 6))
sns.histplot(movie_data['Rating'], bins=30, kde=True, color='skyblue')
plt.title('Distribution of Movie Ratings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.show()

plt.figure(figsize=(8, 6))
sns.boxplot(x=movie_data['Rating'], color='lightcoral')
plt.title('Box Plot of Movie Ratings')
plt.xlabel('Rating')
plt.show()
```

```
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a fu
ture version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```




```
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```



Interpretation:

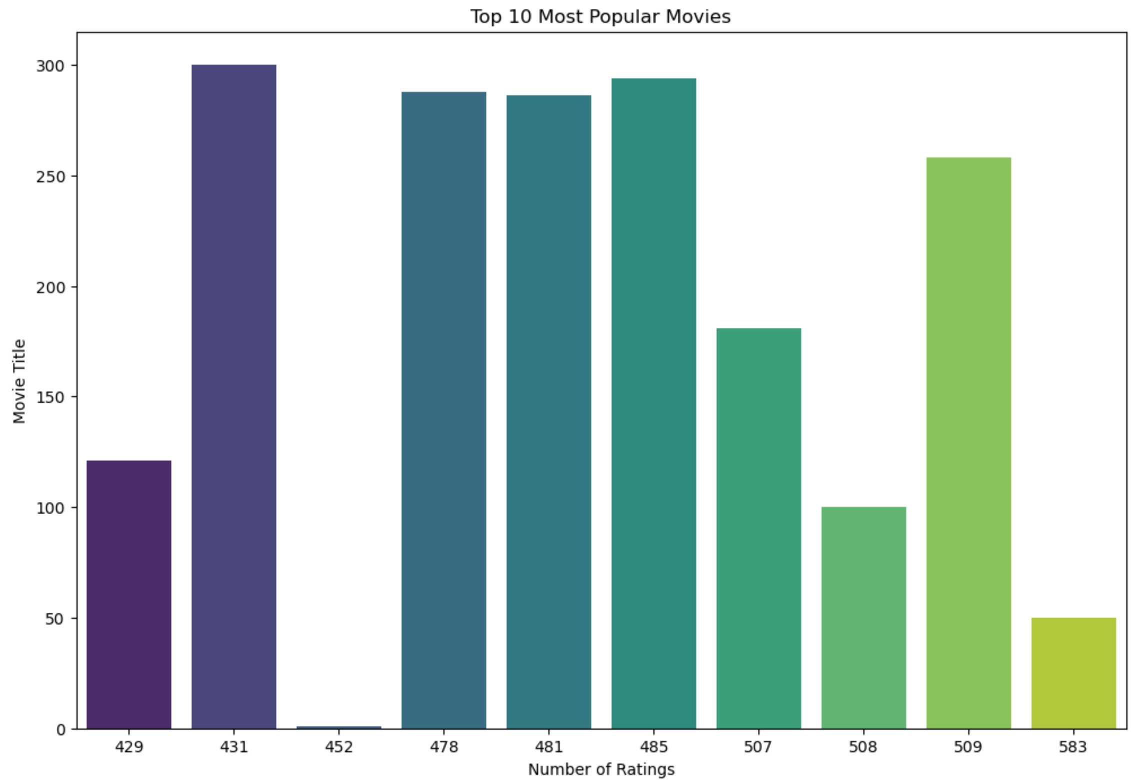
- In the histogram, you can observe the distribution of ratings, whether they are concentrated in a specific range or spread out across various values.
- The boxplot provides information about the median, quartiles, and potential outliers in the rating distribution.

3). Use a bar chart or heatmap to represent the most popular movies (top 10 or 20) among users.

In [48]:

```
ratings_count = movie_data.groupby('Movie_id')['Rating'].count()
top_movies = ratings_count.sort_values(ascending=False).head(10)
plt.figure(figsize=(12, 8))
sns.barplot(x=top_movies.values, y=top_movies.index, palette='viridis'
plt.title('Top 10 Most Popular Movies')
plt.xlabel('Number of Ratings')
plt.ylabel('Movie Title')
plt.show()
```

```
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```



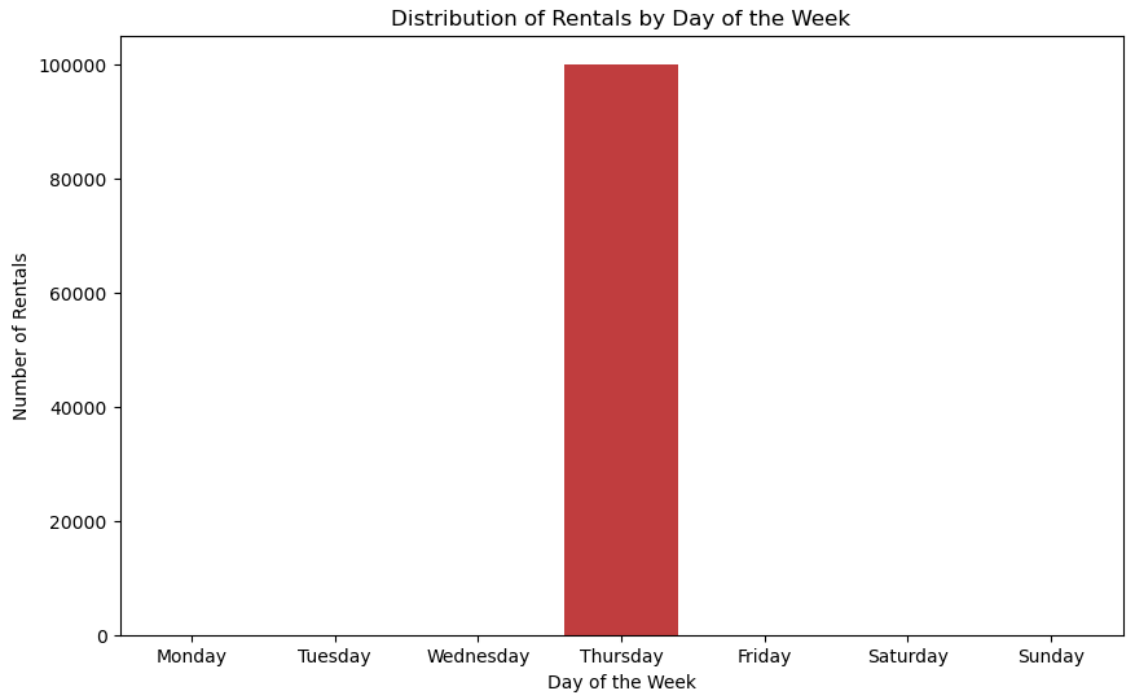
Interpretation:

- The resulting bar chart provides a visual representation of the popularity of movies based on the number of user ratings.
- Movies with higher bars have received more ratings, suggesting a higher level of user engagement.

4). Visualize the distribution of rentals by day of the week or time of day.Are there any preferred rental times?

In [52]:
```
movie_data['timestamp'] = pd.to_datetime(movie_data['timestamp'])
movie_data['day_of_week'] = movie_data['timestamp'].dt.day_name()
plt.figure(figsize=(10, 6))
sns.countplot(x='day_of_week', data=movie_data, order=['Monday', 'Tues
plt.title('Distribution of Rentals by Day of the Week')
plt.xlabel('Day of the Week')
plt.ylabel('Number of Rentals')
plt.show()
```

```
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\pf3zw\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1498: Futu
reWarning: is_categorical_dtype is deprecated and will be removed in a fu
ture version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```



Interpretation:

- The number of rentals is extremely high on Thursday, compared to the other days of the week.
- This could indicate that Thursday is a popular day for renting products or services, or that there is a special promotion or event on that day.

Part 3: Data Summarization

1). Calculate basic descriptive statistics for the "rating" column (mean,median,standard deviation).

In [54]:
```
rating_statistics = movie_data['Rating'].describe()
print(rating_statistics)
```

```
count    100000.000000
mean          3.529860
std           1.125674
min           1.000000
25%           3.000000
50%           4.000000
75%           4.000000
max           5.000000
Name: Rating, dtype: float64
```

Interpretation:

- The average rating is about 3.53, which means that most ratings are above the midpoint of 3.
- The standard deviation is about 1.13, which means that there is some variability in the ratings, but not too much. Most ratings are within one standard deviation of the mean, or between 2.4 and 4.6.
- The median (or 50th percentile) is 4, which means that half of the ratings are above 4 and half are below 4.

2). Summarize the number of unique users and movies in the dataset.

In [56]:
```
unique_users = movie_data['User_id'].nunique()
unique_movies = movie_data['Movie_id'].nunique()

print("Number of Unique Users:", unique_users)
print("Number of Unique Movies:", unique_movies)
```

```
Number of Unique Users: 943
Number of Unique Movies: 1682
```

Interpretation:

- Number of Unique Users: This represents the count of distinct user IDs in the dataset which is 943
- Number of Unique Movies: This represents the count of distinct movie IDs in the dataset which is 1682

3). Find the most active users (top 10 or 20) based on the number of movie rentals.

In [58]:
```
user_rental_counts = movie_data.groupby('User_id')['Movie_id'].count()
top_users = user_rental_counts.sort_values(ascending=False).head(10)
print("Top 10 Most Active Users:")
print(top_users)
```

```
Top 10 Most Active Users:
User_id
405    737
655    685
13     636
450    540
276    518
416    493
537    490
303    484
234    480
393    448
Name: Movie_id, dtype: int64
```

Interpretation:

- The resulting top_users Series contains user IDs as the index and the corresponding counts of movie rentals.
- This information helps identify the users who have rented the most movies, indicating their level of activity or engagement with the rental service.