```
Start coding or generate with AI.
```

1.Find the total number of unique movies.

```
import pandas as pd

# Load the datasets
movies = pd.read_csv('movies.csv')
ratings = pd.read_csv('ratings.csv')

print(movies['movieId'].nunique())
```

```
9742
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

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

```
ratings = pd.read_csv('ratings.csv')
print(ratings['userId'].nunique())
```

```
610
```

3.Find the average rating of all movies.

```
print(ratings['rating'].mean())
```

```
3.501556983616962
```

4.Find the movie with the highest average rating (minimum 50 ratings).

```
movie_avg = ratings.groupby('movieId').agg({'rating': ['mean', 'count']})
movie_avg.columns = ['avg_rating', 'rating_count']
filtered = movie_avg[movie_avg['rating_count'] >= 50]
top_movie_id = filtered['avg_rating'].idxmax()
top_movie_title = movies[movies['movieId'] == top_movie_id]['title'].values[0]
print(top_movie_title)
```

```
Shawshank Redemption, The (1994)
```

5.Find the number of movies belonging to each genre.

```
# Split genres
movies['genres'] = movies['genres'].str.split('|')
all_genres = pd.Series([g for sublist in movies['genres'] for g in sublist])
print(all_genres.value_counts())
```

```
Drama                4361
Comedy               3756
Thriller             1894
Action               1828
Romance              1596
Adventure            1263
Crime                1199
Sci-Fi                980
Horror                978
Fantasy               779
Children              664
Animation             611
Mystery               573
Documentary           440
War                   382
Musical               334
Western               167
IMAX                  158
Film-Noir              87
(no genres listed)     34
Name: count, dtype: int64
```

6.List top 5 most rated movies.

```
top_rated = ratings['movieId'].value_counts().head(5)
print(movies[movies['movieId'].isin(top_rated.index)])
```

```
      movieId                              title  \
257       296               Pulp Fiction (1994)
277       318  Shawshank Redemption, The (1994)
314       356               Forrest Gump (1994)
510       593  Silence of the Lambs, The (1991)
1939     2571                Matrix, The (1999)

                                genres
257   [Comedy, Crime, Drama, Thriller]
277                     [Crime, Drama]
314      [Comedy, Drama, Romance, War]
510          [Crime, Horror, Thriller]
1939        [Action, Sci-Fi, Thriller]
```

7.Find movies with no ratings.

```
rated_movies = ratings['movieId'].unique()
unrated_movies = movies[~movies['movieId'].isin(rated_movies)]
print(unrated_movies)
```

```
      movieId                                       title  \
816      1076                        Innocents, The (1961)
2211     2939                               Niagara (1953)
2499     3338                       For All Mankind (1989)
2587     3456  Color of Paradise, The (Rang-e khoda) (1999)
3118     4194               I Know Where I'm Going! (1945)
4037     5721                           Chosen, The (1981)
4506     6668   Road Home, The (Wo de fu qin mu qin) (1999)
4598     6849                               Scrooge (1970)
4704     7020                                 Proof (1991)
5020     7792                    Parallax View, The (1974)
5293     8765                     This Gun for Hire (1942)
5421    25855                  Roaring Twenties, The (1939)
5452    26085                   Mutiny on the Bounty (1962)
5749    30892            In the Realms of the Unreal (2004)
5824    32160                     Twentieth Century (1934)
5837    32371                    Call Northside 777 (1948)
5957    34482                 Browning Version, The (1951)
7565    85565                           Chalet Girl (2011)

                          genres
816      [Drama, Horror, Thriller]
2211             [Drama, Thriller]
2499                 [Documentary]
2587                       [Drama]
3118         [Drama, Romance, War]
4037                       [Drama]
4506              [Drama, Romance]
4598     [Drama, Fantasy, Musical]
4704      [Comedy, Drama, Romance]
5020                     [Thriller]
5293  [Crime, Film-Noir, Thriller]
5421      [Crime, Drama, Thriller]
5452   [Adventure, Drama, Romance]
5749      [Animation, Documentary]
5824                       [Comedy]
5837     [Crime, Drama, Film-Noir]
5957                       [Drama]
7565             [Comedy, Romance]
```

8.Find the user who rated the most number of movies.

```
top_user = ratings['userId'].value_counts().idxmax()
print(top_user)
```

```
414
```

9.Calculate the standard deviation of movie ratings.

```
print(ratings['rating'].std())
```

```
1.0425292390605359
```

10.Find the earliest and latest rating timestamp (convert timestamp to datetime).

```
ratings['timestamp'] = pd.to_datetime(ratings['timestamp'], unit='s')
print(ratings['timestamp'].min(), ratings['timestamp'].max())
```

```
1996-03-29 18:36:55 2018-09-24 14:27:30
```

11.Find the top 3 genres with highest average ratings.

```
# Expand genres
movies_expanded = movies.explode('genres')
merged = ratings.merge(movies_expanded, on='movieId')
genre_avg = merged.groupby('genres')['rating'].mean().sort_values(ascending=False).head(3)
print(genre_avg)
```

```
genres
Film-Noir      3.920115
War            3.808294
Documentary    3.797785
Name: rating, dtype: float64
```

12.Find how many movies were released each year.

```
# Extract year from title
import re
movies['year'] = movies['title'].str.extract(r'\((\d{4})\)')
print(movies['year'].value_counts().sort_index())
```

```
year
1902      1
1903      1
1908      1
1915      1
1916      4
        ...
2014    278
2015    274
2016    218
2017    147
2018     41
Name: count, Length: 106, dtype: int64
```

13.Find the number of users who have given rating 5.0.

```
users_5star = ratings[ratings['rating'] == 5.0]['userId'].nunique()
print(users_5star)
```

```
573
```

14.Find the movie with the most number of 5-star ratings

```
top_5_movie = ratings[ratings['rating'] == 5.0]['movieId'].value_counts().idxmax()
top_5_title = movies[movies['movieId'] == top_5_movie]['title'].values[0]
print(top_5_title)
```

```
Shawshank Redemption, The (1994)
```

15.Find the average rating given by each user.

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

```
userId
1      4.366379
2      3.948276
3      2.435897
4      3.555556
5      3.636364
```

```
          ...
606    3.657399
607    3.786096
608    3.134176
609    3.270270
610    3.688556
Name: rating, Length: 610, dtype: float64
```

16.Find users who have rated more than 1000 movies.

```
heavy_users = ratings['userId'].value_counts()
print(heavy_users[heavy_users > 1000].index.tolist())
```

```
[414, 599, 474, 448, 274, 610, 68, 380, 606, 288, 249, 387]
```

17.Find the most common rating given.

```
print(ratings['rating'].mode()[0])
```

```
4.0
```

18.Find the correlation between number of ratings and average rating for movies.

```
movie_stats = ratings.groupby('movieId').agg({'rating': ['mean', 'count']})
movie_stats.columns = ['avg_rating', 'rating_count']
print(movie_stats['avg_rating'].corr(movie_stats['rating_count']))
```

```
0.12725857359560638
```

19.Find the movies that are of "Comedy" genre and have an average rating greater than 4.0.

```
comedy_movies = movies[movies['genres'].apply(lambda x: 'Comedy' in x)]
merged = ratings.merge(comedy_movies, on='movieId')
comedy_avg = merged.groupby('title')['rating'].mean()
print(comedy_avg[comedy_avg > 4.0])
```

```
title
00 Schneider - Jagd auf Nihil Baxter (1994)             4.50
12 Chairs (1971)                                        4.50
12 Chairs (1976)                                        5.00
3 Idiots (2009)                                         4.75
A Dog's Purpose (2017)                                  4.50
                                                        ...
Wings, Legs and Tails (1986)                            5.00
Woman Is a Woman, A (femme est une femme, Une) (1961)   5.00
World of Glory (1991)                                   5.00
World of Tomorrow (2015)                                4.50
Wow! A Talking Fish! (1983)                             5.00
Name: rating, Length: 446, dtype: float64
```

20.For each year, find the highest-rated movie (minimum 10 ratings).

```
movies['year'] = movies['title'].str.extract(r'\((\d{4})\)')
movies_expanded = movies.explode('genres')
data = ratings.merge(movies_expanded, on='movieId')
yearly = data.groupby(['year', 'title']).agg({'rating': ['mean', 'count']})
yearly.columns = ['avg_rating', 'rating_count']
yearly = yearly[yearly['rating_count'] >= 10]
result = yearly.reset_index().sort_values(['year', 'avg_rating'], ascending=[True, False]).drop_duplicates('year')
print(result[['year', 'title', 'avg_rating']])
```

```
      year                                          title  avg_rating
0     1902  Trip to the Moon, A (Voyage dans la lune, Le) ...    3.500000
1     1920  Cabinet of Dr. Caligari, The (Cabinet des Dr. ...    3.857143
2     1921                                 Kid, The (1921)    4.100000
3     1922  Nosferatu (Nosferatu, eine Symphonie des Graue...    3.531250
5     1925                           Gold Rush, The (1925)    4.071429
...    ...                                             ...         ...
3424  2014          Black Mirror: White Christmas (2014)    4.750000
3512  2015                             Spotlight (2015)    4.157895
3543  2016                Kubo and the Two Strings (2016)    4.000000
3589  2017   Three Billboards Outside Ebbing, Missouri (2017)    4.750000
```

```
      3598  2018             Avengers: Infinity War - Part I (2018)    4.000000

      [97 rows x 3 columns]
```