Import pandas as pd.

In [3]:
```
import pandas as pd
```

Read games.csv as a dataframe called games.

In [2]:
```
games = pd.read_csv("games.csv")
```

Check the head of the DataFrame.

In [3]:
```
games.head()
```

Out[3]:

	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage	users_rated	average_rating	bayes_average_rating	total_owner
0	12333	boardgame	Twilight Struggle	2005.0	2.0	2.0	180.0	180.0	180.0	13.0	20113	8.33774	8.22186	2664
1	120677	boardgame	Terra Mystica	2012.0	2.0	5.0	150.0	60.0	150.0	12.0	14383	8.28798	8.14232	1651
2	102794	boardgame	Caverna: The Cave Farmers	2013.0	1.0	7.0	210.0	30.0	210.0	12.0	9262	8.28994	8.06886	1223
3	25613	boardgame	Through the Ages: A Story of Civilization	2006.0	2.0	4.0	240.0	240.0	240.0	12.0	13294	8.20407	8.05904	1434
4	3076	boardgame	Puerto Rico	2002.0	2.0	5.0	150.0	90.0	150.0	12.0	39883	8.14261	8.04524	4436

Use .info() method to find out total number of entries in dataset

In [4]:
```
games.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 81312 entries, 0 to 81311
Data columns (total 20 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            81312 non-null  int64
 1   type          81312 non-null  object
 2   name          81271 non-null  object
 3   yearpublished 81309 non-null  float64
 4   minplayers    81309 non-null  float64
 5   maxplayers    81309 non-null  float64
 6   playingtime   81309 non-null  float64
 7   minplaytime   81309 non-null  float64
 8   maxplaytime   81309 non-null  float64
 9   minage        81309 non-null  float64
 10  users_rated   81312 non-null  int64
 11  average_rating  81312 non-null  float64
 12  bayes_average_rating  81312 non-null  float64
 13  total_owners  81312 non-null  int64
 14  total_traders 81312 non-null  int64
 15  total_wishers 81312 non-null  int64
 16  total_weathers 81312 non-null  int64
 17  total_comments 81312 non-null  int64
 18  total_weights 81312 non-null  int64
 19  average_weight  81312 non-null  float64
dtypes: float64(10), int64(8), object(2)
memory usage: 12.4+ MB
```

What is the mean playin time for all games put together ?

In [5]:
```
games['playingtime'].mean()
```

Out[5]:
```
51.634788672082052
```

What is the highest number of comments received for a game?

In [6]:
```
games['total_comments'].max()
```

Out[6]:
```
11798
```

What is the name of the game with id 1500?

In [7]:
```
games[games['id']==1500]['name']
```

Out[7]:
```
10592    Zocken
Name: name, dtype: object
```

And which year was it published?

In [8]:
```
games[games['id']==1500]['yearpublished']
```

Out[8]:
```
10592    1999.0
Name: yearpublished, dtype: float64
```

Which game has received highest number of comments?

In [9]:
```
games[games['total_comments']==games['total_comments'].max()]
```

Out[9]:

	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage	users_rated	average_rating	bayes_average_rating	total_owners	to
165	13	boardgame	Catan	1995.0	3.0	4.0	90.0	90.0	90.0	10.0	53690	7.34303	7.21171	73188	
1965	13	boardgame	Catan	1995.0	3.0	4.0	90.0	90.0	90.0	10.0	53690	7.34303	7.21171	73188	

Which games have received least number of comments?

In [10]:
```
games[games['total_comments']==games['total_comments'].min()]
```

Out[10]:

	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage	users_rated	average_rating	bayes_average_rating
13048	318	boardgame	Looney Leo	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.
13054	468	boardgame	Dump	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2	5.5	0.
13068	579	boardgame	Field of Fire	2002.0	2.0	0.0	0.0	0.0	0.0	12.0	0	0.0	0.
13095	738	boardgame	Matheoro	2000.0	2.0	2.0	20.0	20.0	20.0	9.0	1	3.0	0.
13103	779	boardgame	Auction Madness: The Trivia Game for Any Colle...	2000.0	2.0	4.0	60.0	60.0	60.0	0.0	1	4.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...
81307	186441	boardgameexpansion	Micro Mechs: Anigurhon	2015.0	1.0	1.0	0.0	30.0	0.0	10.0	0	0.0	.
81308	186442	boardgame	Trivial Pursuit: Marvel Cinematic Universe Ou...	2013.0	2.0	0.0	0.0	0.0	0.0	12.0	0	0.0	0.
81309	186443	boardgame	BEARiania	2015.0	2.0	15.0	1.0	1.0	1.0	0.0	0	0.0	0.
81310	186449	boardgame	Freight	2015.0	2.0	4.0	60.0	30.0	60.0	8.0	0	0.0	0.
81311	186451	boardgame	Bingo Animal Kids	2010.0	1.0	6.0	10.0	10.0	10.0	2.0	0	0.0	0.

29001 rows × 20 columns

What was the average minage of all games per game *type*? (boardgame & boardgameexpansion)

In [11]:
```
games.groupby('type').mean()['minage']
```

Out[11]:
```
type
boardgame            6.724798
boardgameexpansion   8.723321
Name: minage, dtype: float64
```

How many unique games are there in the dataset?

In [12]:
```
games['id'].nunique()
```

Out[12]:
```
79463
```

How many boardgames and boardgameexpansions are there in the dataset?

In [13]:
```
games['type'].value_counts()
```

Out[13]:
```
boardgame            70820
boardgameexpansion   10492
Name: type, dtype: int64
```

Is there a correlation between playing time and total comments for the games? - Use the .corr() function

In [14]:
```
games[['playingtime','total_comments']].corr()
```

Out[15]:

	playingtime	total_comments
playingtime	1.000000	0.020645
total_comments	0.020645	1.000000

Data Visualization using Seaborn

Import the seaborn library and set color codes as true

In [18]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set(color_codes=True)
```

Drop na values for negating issues during visualization

In [20]:
```
games.dropna(inplace=True)
```

In [23]:
```
games.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 81268 entries, 0 to 81311
Data columns (total 20 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            81268 non-null  int64
 1   type          81268 non-null  object
 2   name          81268 non-null  object
 3   yearpublished 81268 non-null  float64
 4   minplayers    81268 non-null  float64
 5   maxplayers    81268 non-null  float64
 6   playingtime   81268 non-null  float64
 7   minplaytime   81268 non-null  float64
 8   maxplaytime   81268 non-null  float64
 9   minage        81268 non-null  float64
 10  users_rated   81268 non-null  int64
 11  average_rating  81268 non-null  float64
 12  bayes_average_rating  81268 non-null  float64
 13  total_owners  81268 non-null  int64
 14  total_traders 81268 non-null  int64
 15  total_wishers 81268 non-null  int64
 16  total_weathers 81268 non-null  int64
 17  total_comments 81268 non-null  int64
 18  total_weights 81268 non-null  int64
 19  average_weight  81268 non-null  float64
dtypes: float64(10), int64(8), object(2)
memory usage: 13.0+ MB
```

View the distance plot for minage

In [24]:
```
sns.distplot(games['average_rating'])
```

```
C:\Users\swara\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
<AxesSubplot:xlabel='average_rating', ylabel='Density'>
```

Out[24]:

Is there a linear relationship between Minage & average_rating?

In [25]:
```
sns.jointplot(games['minage'],games['average_rating'])
```

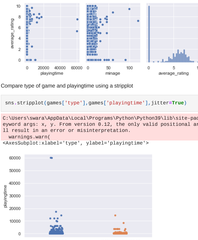
```
C:\Users\swara\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as k
eyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword wi
ll result in an error or misinterpretation.
  warnings.warn(
<seaborn.axisgrid.JointGrid at 0x2a4135bfab>
```

Out[25]:

Compare the relationship between playingtime , minage and average rating using pairplot

In [26]:
```
sns.pairplot(games[['playingtime','minage','average_rating']])
```

Out[26]:
```
<seaborn.axisgrid.PairGrid at 0x2a4155fbd30>
```

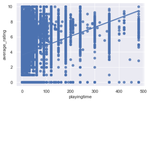


Compare type of game and playingtime using a stripplot

In [28]:
```
sns.stripplot(games['type'],games['playingtime'],jitter=True)
```

```
C:\Users\swara\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as k
eyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword wi
ll result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='type', ylabel='playingtime'>
```

Out[28]:



Analyze the linear trend between playing time(less than 500 mins) and average_rating received for the game

In [29]:
```
sns.regplot(x="playingtime",y="average_rating",data=games[games["playingtime"]<500])
```

Out[29]: