Data Analytics

What makes a board game successful?

Michael Elbaz

Febuary, 2025

Table of content

Introduction

1. **Business Use Case**

   Understanding what are the main factors that may influence the chances of success when releasing a new board game on the market.

   Qualified and funny board games satisfy customers and may enhance the maximization of profit for a publishing company or for an independent game creator.

2. **Goal**

   The goal of my project is to:
   - Analyze the various criteria that belong to the board-gaming industry
   - Identify the key factors of a good game rating, and create a prediction model

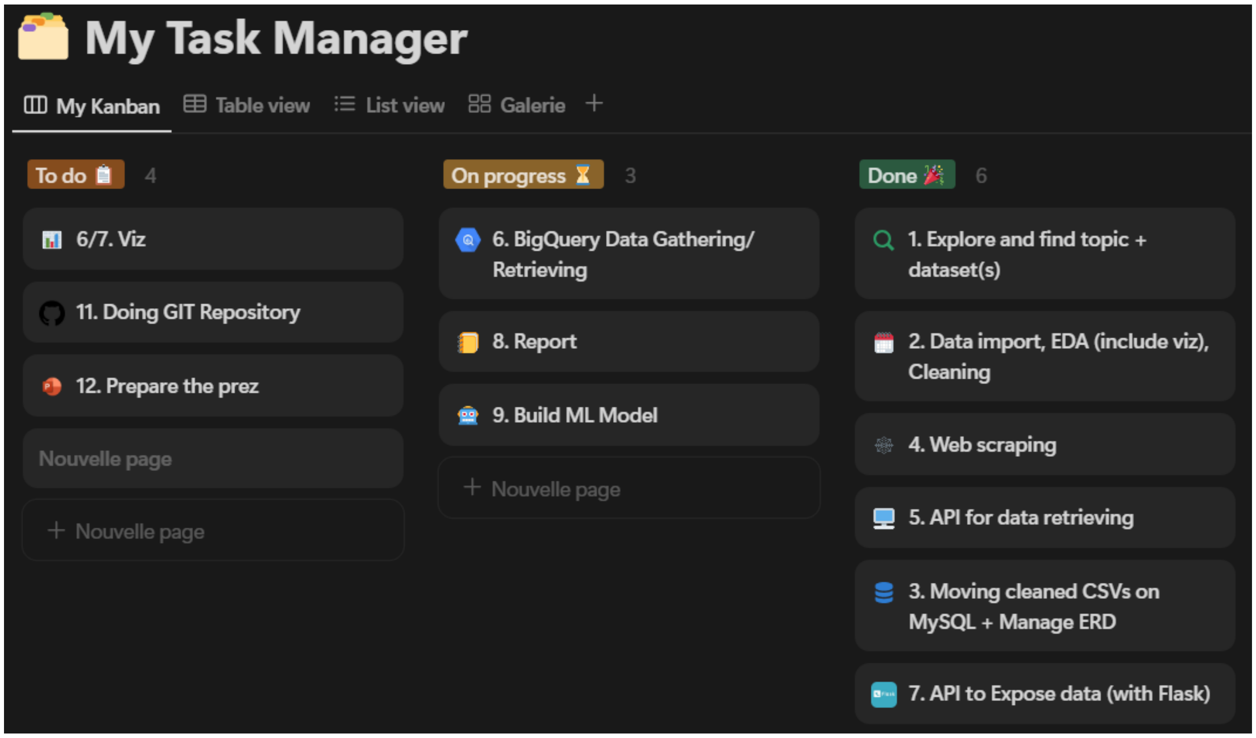
3. **Plan**

   - Research about project topic
   - Data collection
   - Project Planning on Notion
   - Selection and creation of a database using MySQL
   - Adding data to databases and creating an Entity Relationship Diagram
   - Data manipulation through SQL use
   - Exploratory Data Analysis with Python (wrangling, normalizing, cleaning, removing outliers, visualizations)
   - Exposing data via API

Project Management

Overview of my Notion Board : structure & management of daily tasks

Link: [Here]()



My Task Manager

To do 4	On progress ⌛ 3	Done 🎉 6
6/7. Viz	6. BigQuery Data Gathering/ Retrieving	1. Explore and find topic + dataset(s)
11. Doing GIT Repository	8. Report	2. Data import, EDA (include viz), Cleaning
12. Prepare the prez	9. Build ML Model	4. Web scraping
Nouvelle page	+ Nouvelle page	5. API for data retrieving
+ Nouvelle page		3. Moving cleaned CSVs on MySQL + Manage ERD
		7. API to Expose data (with Flask)

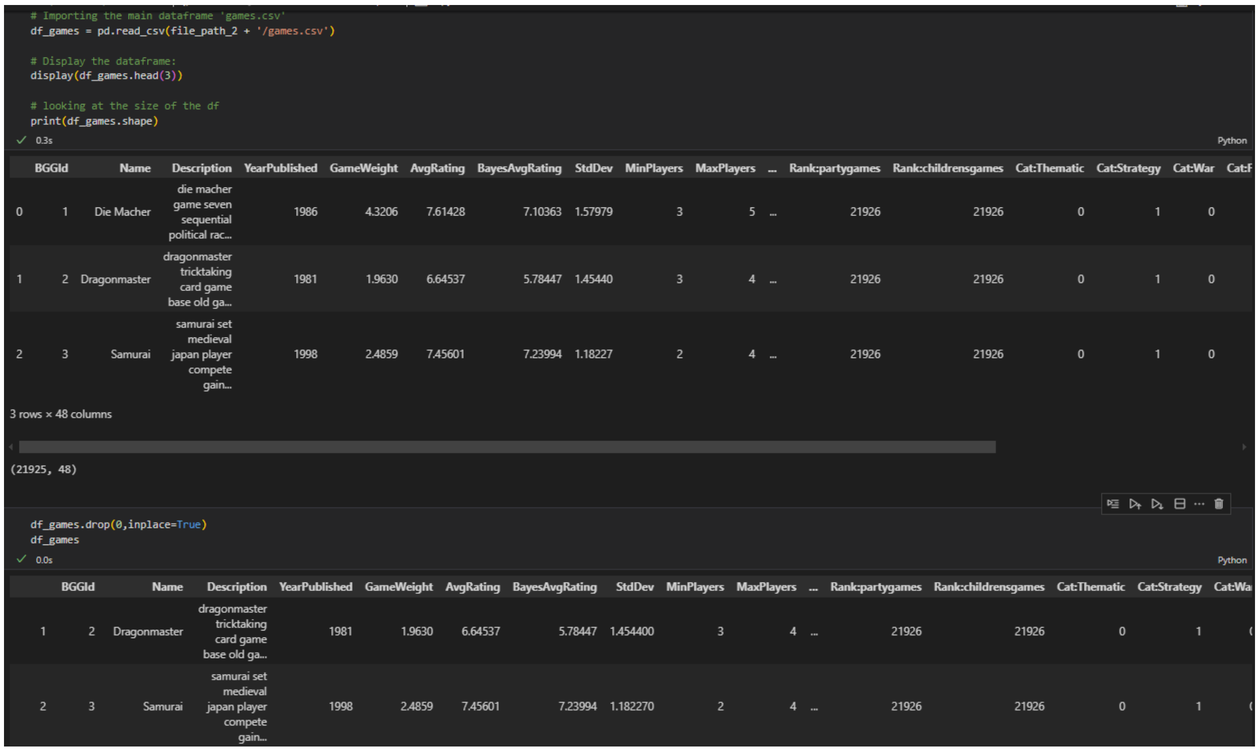
Data and data sources

For this project, I gathered information from various places to create my datasets. The main data was collected from Kaggle.

In addition to this, different methods were used such as APIs, BigQuery, and Web scrapping.

1. **Flat data**

After a long time of research and comparing various Kaggle folders, my choice went to https://www.kaggle.com/datasets/threnjen/board-games-database-from-boardgamegeek as it contains many information and relates important criteria such as them, category, mechanism, artists and publishers in addition to the central KPI I'm focusing at: the average rating of a board game. It also provides the number of times a board game has been rated but also info regarding playtime or number of players.



```python
# Importing the main dataframe 'games.csv'
df_games = pd.read_csv(file_path_2 + '/games.csv')

# Display the dataframe:
display(df_games.head(3))

# looking at the size of the df
print(df_games.shape)
```

✓ 0.3s Python

	BGGId	Name	Description	YearPublished	GameWeight	AvgRating	BayesAvgRating	StdDev	MinPlayers	MaxPlayers	...	Rank:partygames	Rank:childrensgames	Cat:Thematic	Cat:Strategy	Cat:War	Cat:F
0	1	Die Macher	die macher game seven sequential political rac...	1986	4.3206	7.61428	7.10363	1.57979	3	5	...	21926	21926	0	1	0	
1	2	Dragonmaster	dragonmaster tricktaking card game base old ga...	1981	1.9630	6.64537	5.78447	1.45440	3	4	...	21926	21926	0	1	0	
2	3	Samurai	samurai set medieval japan player compete gain...	1998	2.4859	7.45601	7.23994	1.18227	2	4	...	21926	21926	0	1	0	

3 rows × 48 columns

```
(21925, 48)
```

```python
df_games.drop(0,inplace=True)
df_games
```

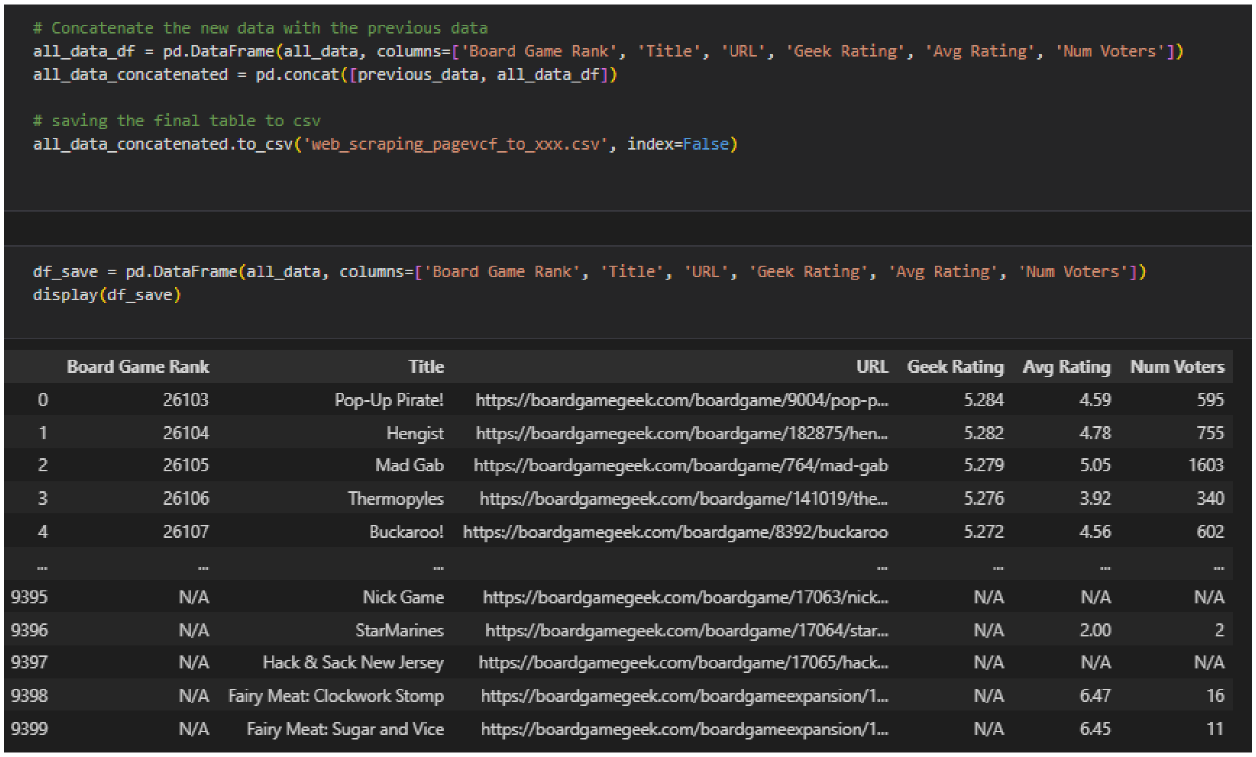
✓ 0.0s Python

	BGGId	Name	Description	YearPublished	GameWeight	AvgRating	BayesAvgRating	StdDev	MinPlayers	MaxPlayers	...	Rank:partygames	Rank:childrensgames	Cat:Thematic	Cat:Strategy	Cat:Wa
1	2	Dragonmaster	dragonmaster tricktaking card game base old ga...	1981	1.9630	6.64537	5.78447	1.454400	3	4	...	21926	21926	0	1	(
2	3	Samurai	samurai set medieval japan player compete gain...	1998	2.4859	7.45601	7.23994	1.182270	2	4	...	21926	21926	0	1	(

2. **Web scrapping**

Since the flat files already explore board games being listed on board game geek database, I had a look at various places and went to the conclusion that the most convenient thing would be to scrap their browsing area from their website, as the URL structure is always the same.

<u>Source</u>:
https://boardgamegeek.com/browse/boardgame
or
https://boardgamegeek.com/browse/boardgame/page/1



```python
# Concatenate the new data with the previous data
all_data_df = pd.DataFrame(all_data, columns=['Board Game Rank', 'Title', 'URL', 'Geek Rating', 'Avg Rating', 'Num Voters'])
all_data_concatenated = pd.concat([previous_data, all_data_df])

# saving the final table to csv
all_data_concatenated.to_csv('web_scraping_pagevcf_to_xxx.csv', index=False)
```

```python
df_save = pd.DataFrame(all_data, columns=['Board Game Rank', 'Title', 'URL', 'Geek Rating', 'Avg Rating', 'Num Voters'])
display(df_save)
```

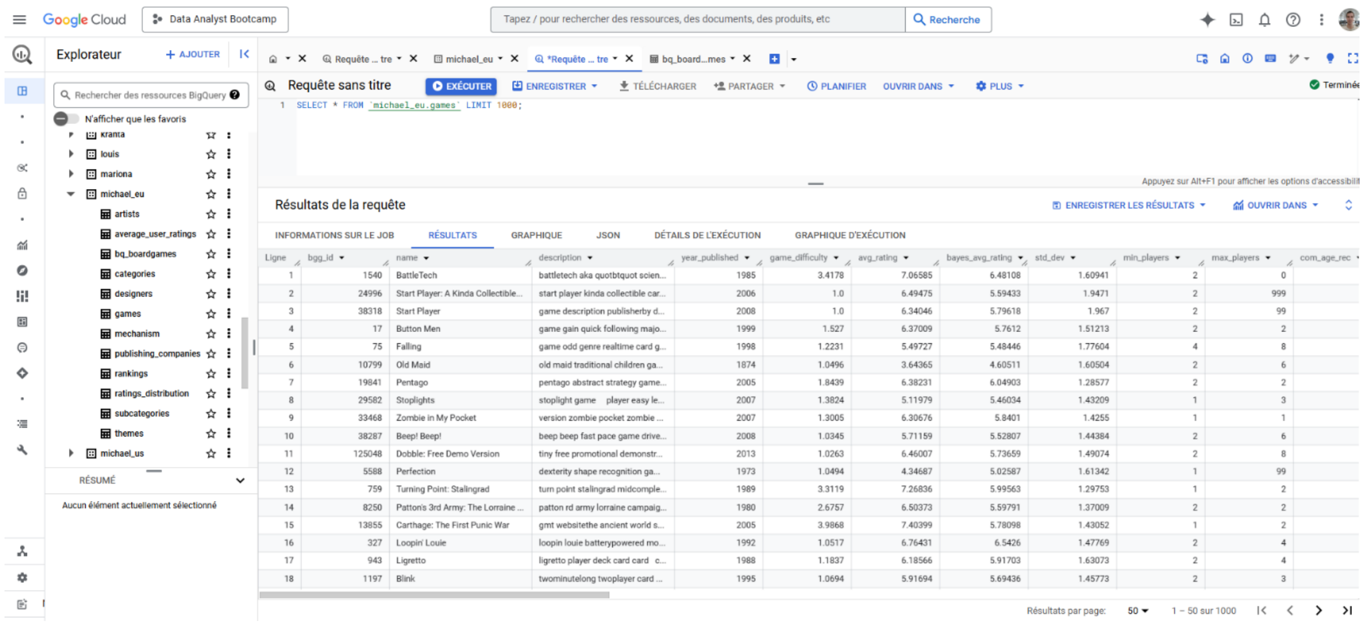
	Board Game Rank	Title	URL	Geek Rating	Avg Rating	Num Voters
0	26103	Pop-Up Pirate!	https://boardgamegeek.com/boardgame/9004/pop-p...	5.284	4.59	595
1	26104	Hengist	https://boardgamegeek.com/boardgame/182875/hen...	5.282	4.78	755
2	26105	Mad Gab	https://boardgamegeek.com/boardgame/764/mad-gab	5.279	5.05	1603
3	26106	Thermopyles	https://boardgamegeek.com/boardgame/141019/the...	5.276	3.92	340
4	26107	Buckaroo!	https://boardgamegeek.com/boardgame/8392/buckaroo	5.272	4.56	602
...	...	...	...	...	...	...
9395	N/A	Nick Game	https://boardgamegeek.com/boardgame/17063/nick...	N/A	N/A	N/A
9396	N/A	StarMarines	https://boardgamegeek.com/boardgame/17064/star...	N/A	2.00	2
9397	N/A	Hack & Sack New Jersey	https://boardgamegeek.com/boardgame/17065/hack...	N/A	N/A	N/A
9398	N/A	Fairy Meat: Clockwork Stomp	https://boardgamegeek.com/boardgameexpansion/1...	N/A	6.47	16
9399	N/A	Fairy Meat: Sugar and Vice	https://boardgamegeek.com/boardgameexpansion/1...	N/A	6.45	11

3. **API**

Here again, BGG be used through its API service, as an additional side-resource:
https://boardgamegeek.com/wiki/page/BGG_XML_API2
Since the Boardgame Id is present it can be used to complete the ones missing on my original dataset.

4. **Big Query (Google Cloud Platform)**

Tables from my schema /project (michael_eu) are available here:
https://console.cloud.google.com/bigquery?project=da-bootcamp-2023&ws=!1m4!1m3!3m2!1sda-bootcamp-2023!2smichael_eu&inv=1&invt=AbpTVA



```
SELECT * FROM `michael_eu.games` LIMIT 1000;
```

Ligne	bgg_id	name	description	year_published	game_difficulty	avg_rating	bayes_avg_rating	std_dev	min_players	max_players	com_age_rec
1	1540	BattleTech	battletech aka quotbtquot scien...	1985	3.4178	7.06585	6.48108	1.60941	2	0	
2	24996	Start Player: A Kinda Collectible...	start player kinda collectible car...	2006	1.0	6.49475	5.59433	1.9471	2	999	
3	38318	Start Player	game description publisherby d...	2008	1.0	6.34046	5.79618	1.967	2	99	
4	17	Button Men	game gain quick following majo...	1999	1.527	6.37009	5.7612	1.51213	2	2	
5	75	Falling	game odd genre realtime card g...	1998	1.2231	5.49727	5.48446	1.77604	4	8	
6	10799	Old Maid	old maid traditional children ga...	1874	1.0496	3.64365	4.60511	1.60504	2	6	
7	19841	Pentago	pentago abstract strategy game...	2005	1.8439	6.38231	6.04903	1.28577	2	2	
8	29582	Stoplights	stoplight game player easy le...	2007	1.3824	5.11979	5.46034	1.43209	1	3	
9	33468	Zombie in My Pocket	version zombie pocket zombie ...	2007	1.3005	6.30676	5.8401	1.4255	1	1	
10	38287	Beep! Beep!	beep beep fast pace game drive...	2008	1.0345	5.71159	5.52807	1.44384	2	6	
11	125048	Dobble: Free Demo Version	tiny free promotional demonstr...	2013	1.0263	6.46007	5.73659	1.49074	2	8	
12	5588	Perfection	dexterity shape recognition ga...	1973	1.0494	4.34687	5.02587	1.61342	1	99	
13	759	Turning Point: Stalingrad	turn point stalingrad midcomple...	1989	3.3119	7.26836	5.99563	1.29753	1	2	
14	8250	Patton's 3rd Army: The Lorraine ...	patton rd army lorraine campaig...	1980	2.6757	6.50373	5.59791	1.37009	2	2	
15	13855	Carthage: The First Punic War	gmt websitethe ancient world s...	2005	3.9868	7.40399	5.78098	1.43052	1	2	
16	327	Loopin' Louie	loopin louie batterypowered mo...	1992	1.0517	6.76431	6.5426	1.47769	2	4	
17	943	Ligretto	ligretto player deck card card c...	1988	1.1837	6.18566	5.91703	1.63073	2	4	
18	1197	Blink	twominutelong twoplayer card ...	1995	1.0694	5.91694	5.69436	1.45773	2	3	

Data cleaning and Exploratory data analysis

1. Main dataset overview

EDA and cleaning were realized with Python using mainly pandas library.
Multiple datasets (csv) were downloaded from Kaggle. The main and most important one (games.csv) relates all the key features associated with board games.

Main columns and key features:

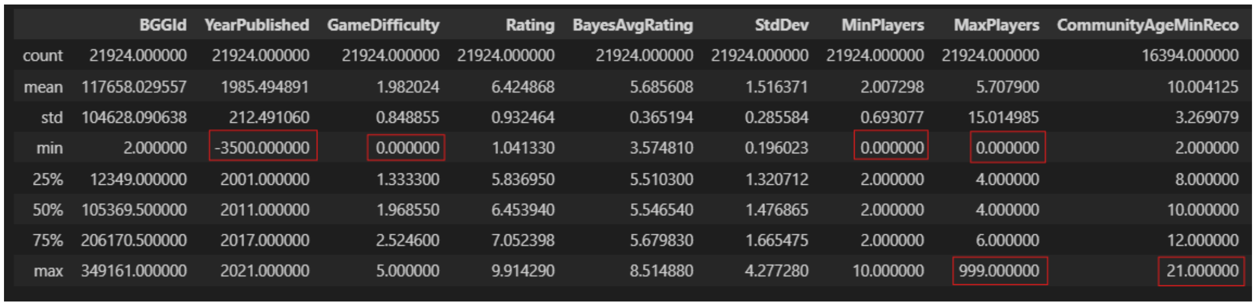
- BGG Id: The identifier for each board game – is a primary key
- Name: name of the board game
- Year published: Year of game publishing
- Game weight (scale: 1-5): Game difficulty / complexity
- Average rating (scale: 1- 10): Average rating for a game, the central kpi for my exploratory and predictive analysis – appears as well in two other datasets (average_rating_distribution and user_ratings csvs)
- Minimum players: Minimum of players required to play
- Maximum players: Maximum number of players allowed
- Community age recommendation: Minimum age recommended by the community
- Language ease: Language requirement (wasn't considered given its values)
- Manufacturer play time (minutes): Manufacturer stated play time
- Community minimum play time (minutes): Minimum play time stated by the community
- Community maximum play time (minutes): Maximum play time stated by the community
- Manufacturer age recommendation: Manufacturer age recommendation
- Kickstarted: binary format: 1 if the game creation was financed thanks to a Kickstarter crowdfunding campaign, 0 if not.

In addition to that, we also have other datasets (mechanics, themes, subcategories, artists, designers, publishers) where common columns being the BGG ID and the others being the categorical variables stated into binary format (1 or 0).

2. Data cleaning

- Since the games dataset contained specific binary columns about Categories, I split it to obtain another one. This one was reorganized so its columns headers had a clear name (Cat: Strategy' to 'Strategy' for instance). From my "games" data frame, I extracted specific columns to create 2 new data frames, one for categories and one for rankings.

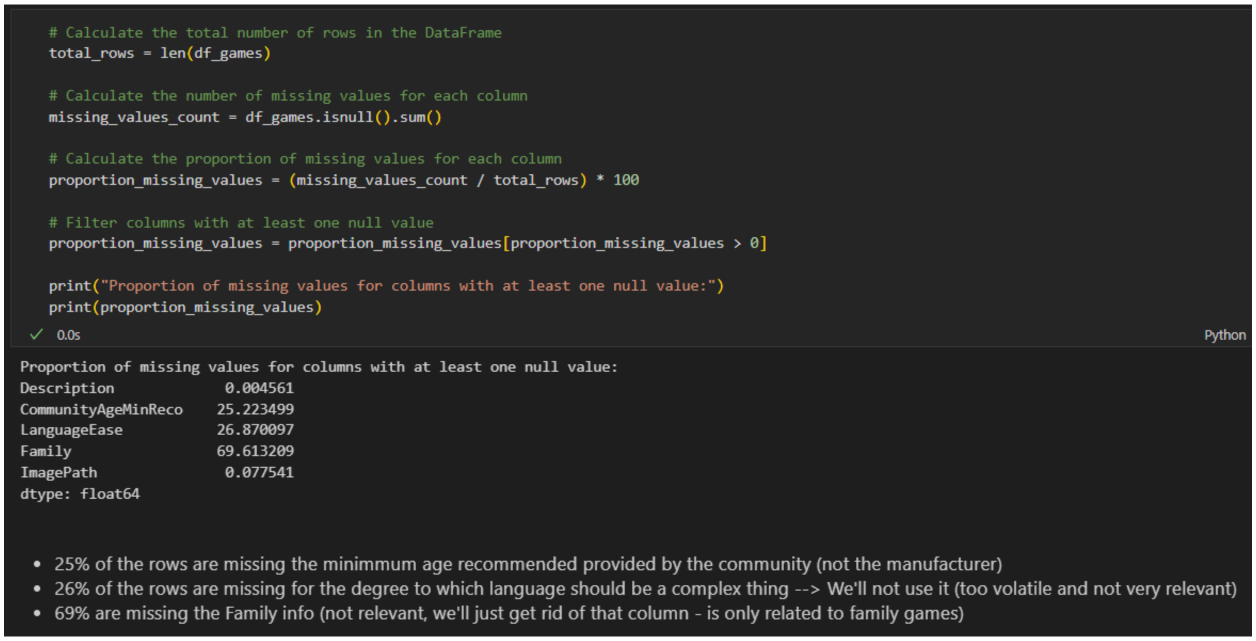
- For each dataset, I could observe the summary statistics and see for each variable what were the maximum values, minimum values, average, standard deviations and then have an idea of what variable could present some outliers



	BGGId	YearPublished	GameDifficulty	Rating	BayesAvgRating	StdDev	MinPlayers	MaxPlayers	CommunityAgeMinReco
count	21924.000000	21924.000000	21924.000000	21924.000000	21924.000000	21924.000000	21924.000000	21924.000000	16394.000000
mean	117658.029557	1985.494891	1.982024	6.424868	5.685608	1.516371	2.007298	5.707900	10.004125
std	104628.090638	212.491060	0.848855	0.932464	0.365194	0.285584	0.693077	15.014985	3.269079
min	2.000000	-3500.000000	0.000000	1.041330	3.574810	0.196023	0.000000	0.000000	2.000000
25%	12349.000000	2001.000000	1.333300	5.836950	5.510300	1.320712	2.000000	4.000000	8.000000
50%	105369.500000	2011.000000	1.968550	6.453940	5.546540	1.476865	2.000000	4.000000	10.000000
75%	206170.500000	2017.000000	2.524600	7.052398	5.679830	1.665475	2.000000	6.000000	12.000000
max	349161.000000	2021.000000	5.000000	9.914290	8.514880	4.277280	10.000000	999.000000	21.000000

Then, I could remove these outliers and keepi only the relavant board games.

- For each dataset, I also ensured that we had no duplicates and that the proportion of NaN and / or null values wouldn't be too important or wouldn't be on a key variable



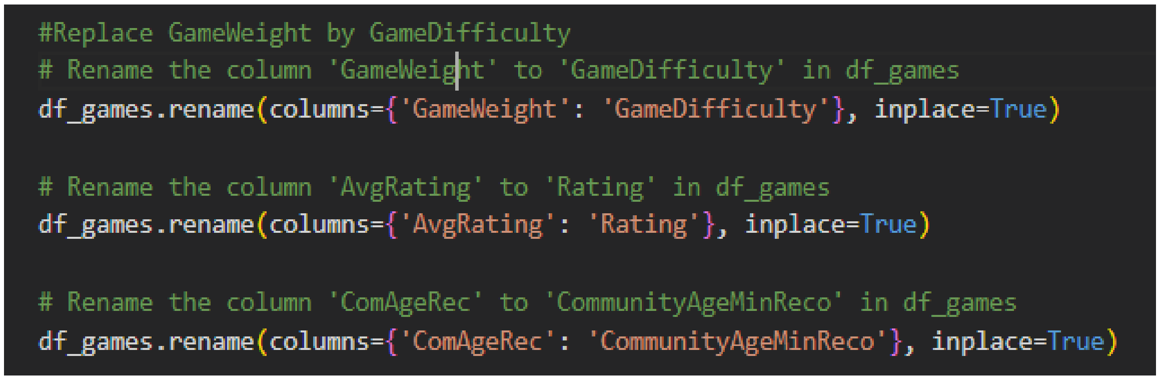
```
# Calculate the total number of rows in the DataFrame
total_rows = len(df_games)

# Calculate the number of missing values for each column
missing_values_count = df_games.isnull().sum()

# Calculate the proportion of missing values for each column
proportion_missing_values = (missing_values_count / total_rows) * 100

# Filter columns with at least one null value
proportion_missing_values = proportion_missing_values[proportion_missing_values > 0]

print("Proportion of missing values for columns with at least one null value:")
print(proportion_missing_values)
```

✓ 0.0s Python

```
Proportion of missing values for columns with at least one null value:
Description             0.004561
CommunityAgeMinReco    25.223499
LanguageEase           26.870097
Family                 69.613209
ImagePath               0.077541
dtype: float64
```

- 25% of the rows are missing the minimmum age recommended provided by the community (not the manufacturer)
- 26% of the rows are missing for the degree to which language should be a complex thing --> We'll not use it (too volatile and not very relevant)
- 69% are missing the Family info (not relevant, we'll just get rid of that column - is only related to family games)

- A renaming of certain columns was also established. This way for instance, Game weight' became 'Game difficulty'

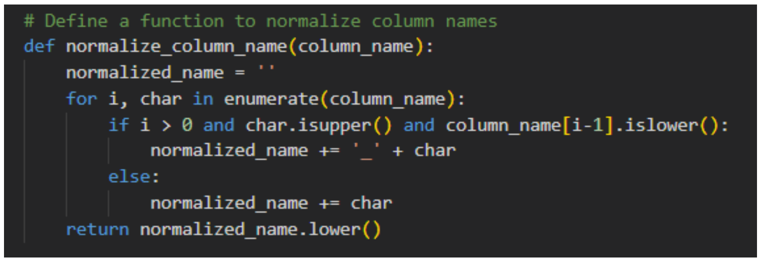


```
#Replace GameWeight by GameDifficulty
# Rename the column 'GameWeight' to 'GameDifficulty' in df_games
df_games.rename(columns={'GameWeight': 'GameDifficulty'}, inplace=True)

# Rename the column 'AvgRating' to 'Rating' in df_games
df_games.rename(columns={'AvgRating': 'Rating'}, inplace=True)

# Rename the column 'ComAgeRec' to 'CommunityAgeMinReco' in df_games
df_games.rename(columns={'ComAgeRec': 'CommunityAgeMinReco'}, inplace=True)
```

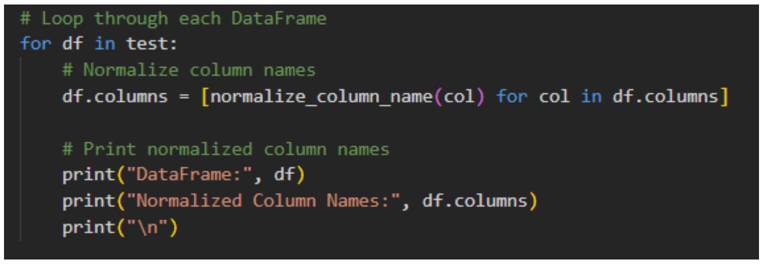
- Other observations and cleanings were realized as well, like observing the format, rounding numbers or passing them from float to integers

3. Normalizing

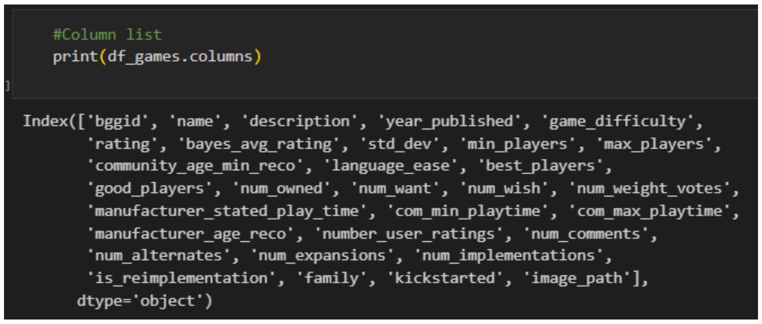
For each dataset, column names were normalized the same way:



```
# Define a function to normalize column names
def normalize_column_name(column_name):
    normalized_name = ''
    for i, char in enumerate(column_name):
        if i > 0 and char.isupper() and column_name[i-1].islower():
            normalized_name += '_' + char
        else:
            normalized_name += char
    return normalized_name.lower()
```



```
# Loop through each DataFrame
for df in test:
    # Normalize column names
    df.columns = [normalize_column_name(col) for col in df.columns]

    # Print normalized column names
    print("DataFrame:", df)
    print("Normalized Column Names:", df.columns)
    print("\n")
```

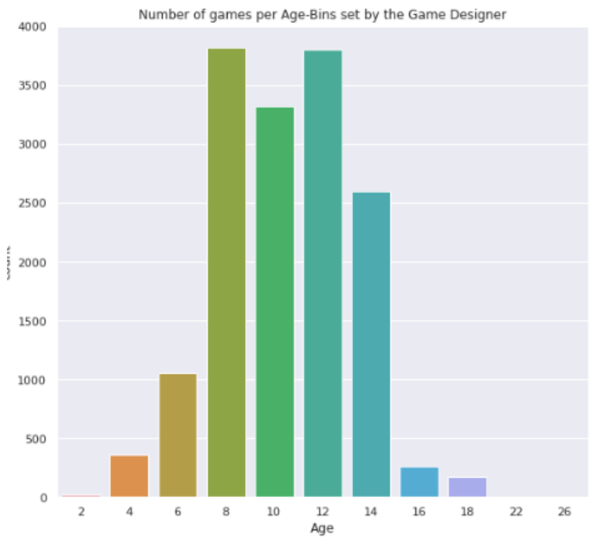


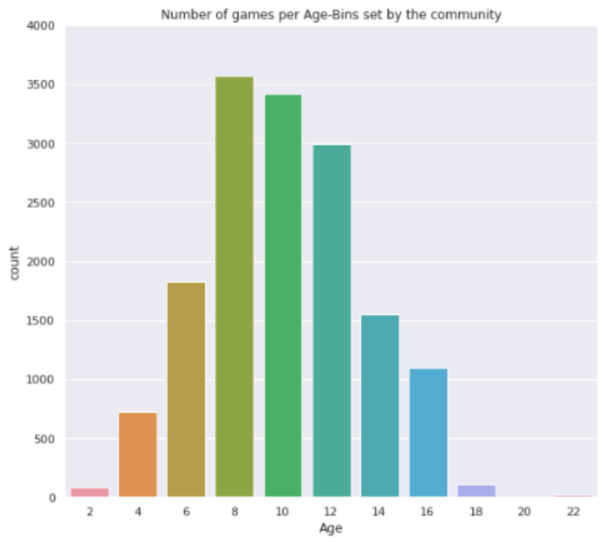
```
#Column list
print(df_games.columns)
```

```
Index(['bggid', 'name', 'description', 'year_published', 'game_difficulty',
       'rating', 'bayes_avg_rating', 'std_dev', 'min_players', 'max_players',
       'community_age_min_reco', 'language_ease', 'best_players',
       'good_players', 'num_owned', 'num_want', 'num_wish', 'num_weight_votes',
       'manufacturer_stated_play_time', 'com_min_playtime', 'com_max_playtime',
       'manufacturer_age_reco', 'number_user_ratings', 'num_comments',
       'num_alternates', 'num_expansions', 'num_implementations',
       'is_reimplementation', 'family', 'kickstarted', 'image_path'],
      dtype='object')
```

Visualizations

Minimum recommended age

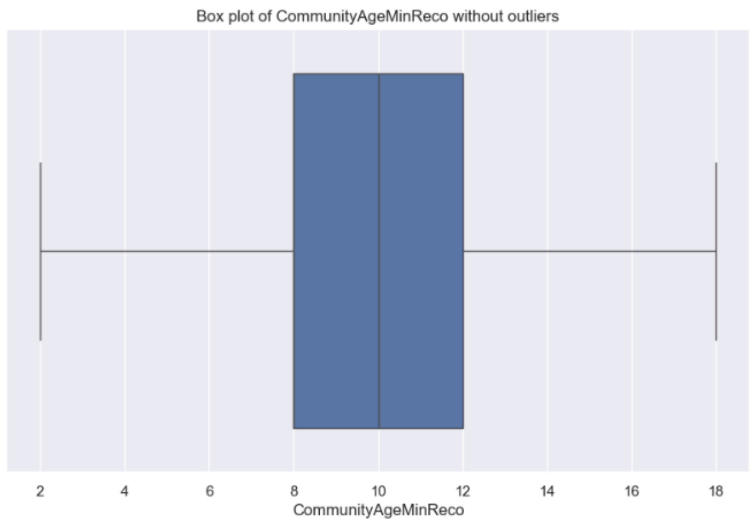




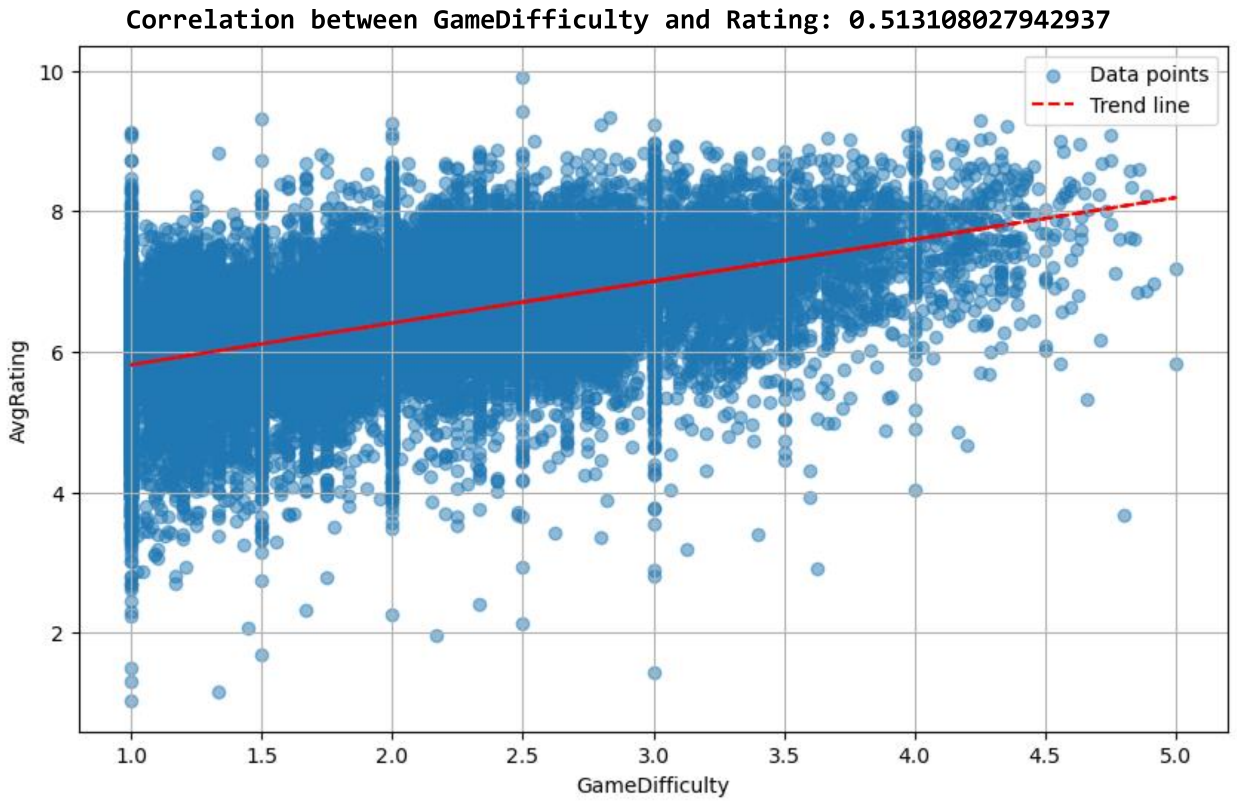
Left bar chart: Manufacturer minimum age recommendation
Right bar chart: Community minimum age recommendation

We can observe that:
- Most board games are between 8 to 12 years old
- The community (right graph) follows more likely a normal distribution
- Certain values are outliers

After outliers cleaning on community recommended minimum age variable, we keep board games where age goes from 2 to 18 years old



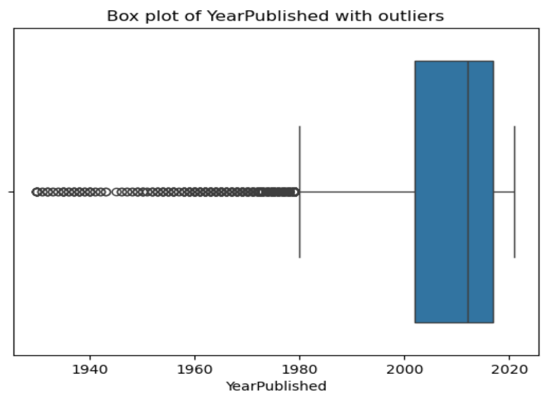
Game difficulty and rating

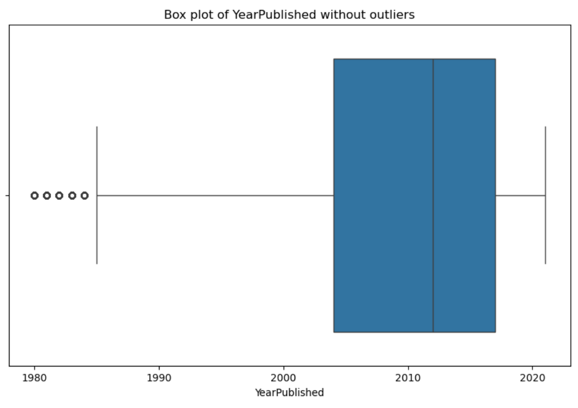


We can observe a clear correlation between the difficulty of a game and its average rating. Nevertheless, the more complicated a game is, the fewer data points we have.

Year of publishing emergence

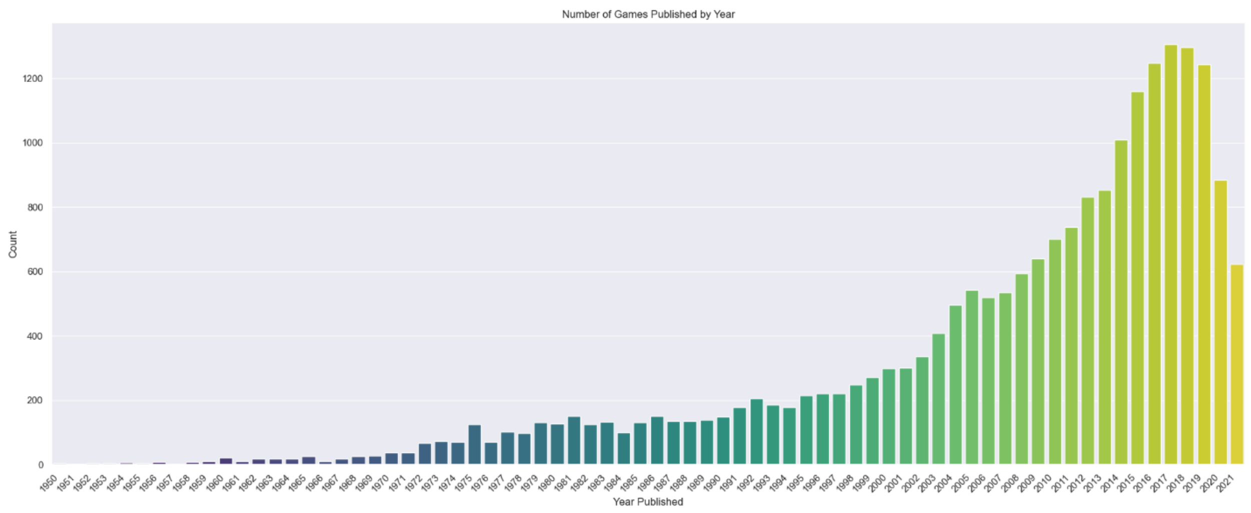
Comparing the years of board games publishing with and without





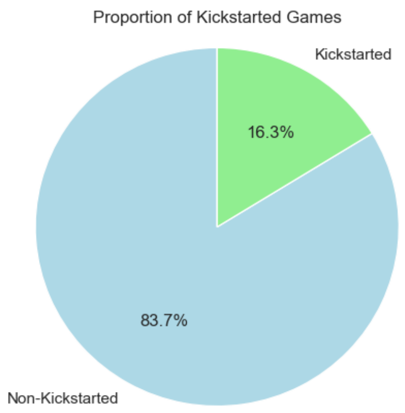
In our cleaned dataset, we go from 1979 up to 2021, the last year in which our dataset has values. It already appears obvious that board games before the 20th century were not relevant to us.

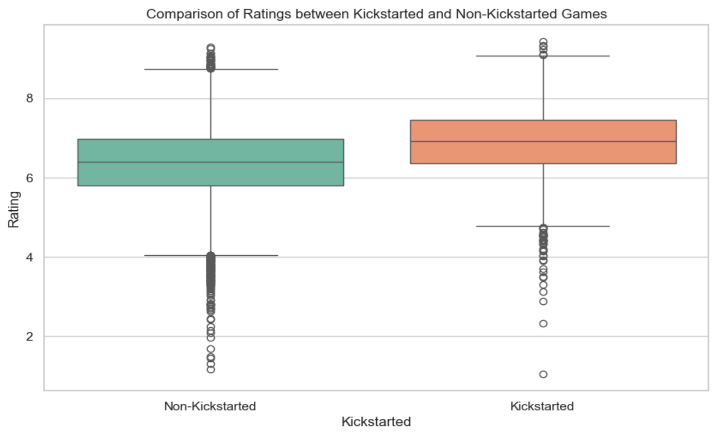
Here is a focus on board games emergency between 1950 and 2021



We can notice here a clear emergence of board games starting from the 80's
We reach a peak when approaching 2020

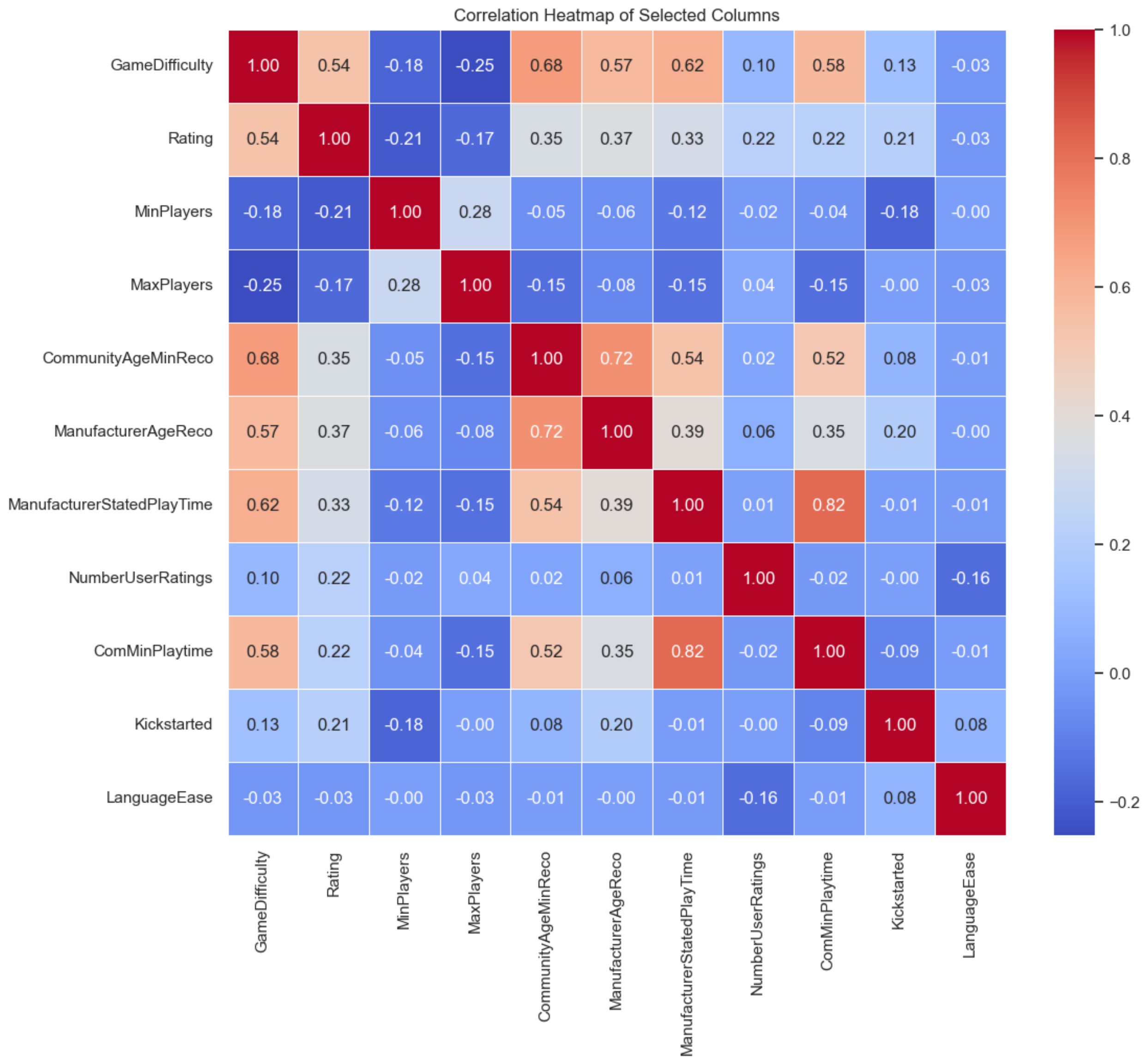
Kickstarter and its influence





Insight: A board game being launched on Kickstarter has bigger chances to obtain a better rating.

Heatmap: Correlation between board game rating and its factors



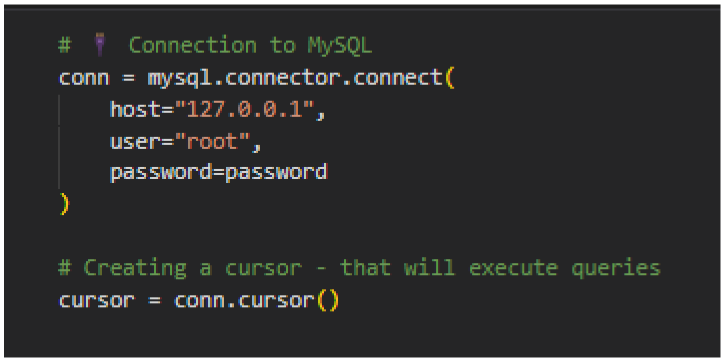
The variables being observed are numeric and correspond to our main “games” dataset

<u>Insights and observations:</u>

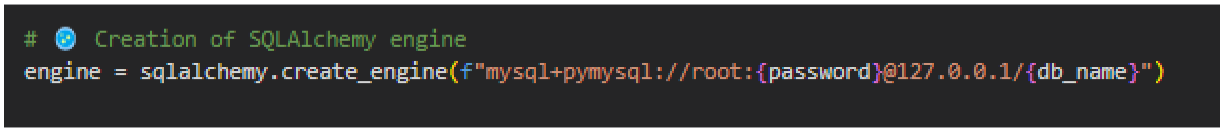
- Game difficulty foes up with minimum recommended age, which seems logical
- Game difficulty also arises with the average play time
- As we could observe before, we have a positive correlation between rating and game difficulty
- Negative correlation between the minimum number of players and the rating (the more a game requires players, the less the rating)
- The same phenomena apply with the maximum number of players

Database type selection

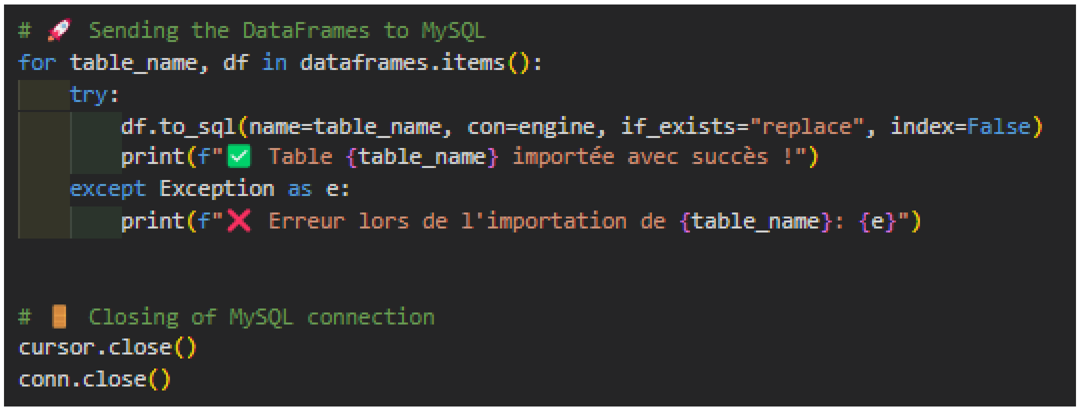
The database selected is MySQL. The organization of my data involved defining my tables along with their relationships and appropriate primary and foreign keys.
After cleaning the data in Python, the connection to MySQL was established to transfer my dataframes (tables) to the 'bgg_project' database I created.



```python
# 🔌 Connection to MySQL
conn = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password=password
)

# Creating a cursor - that will execute queries
cursor = conn.cursor()
```



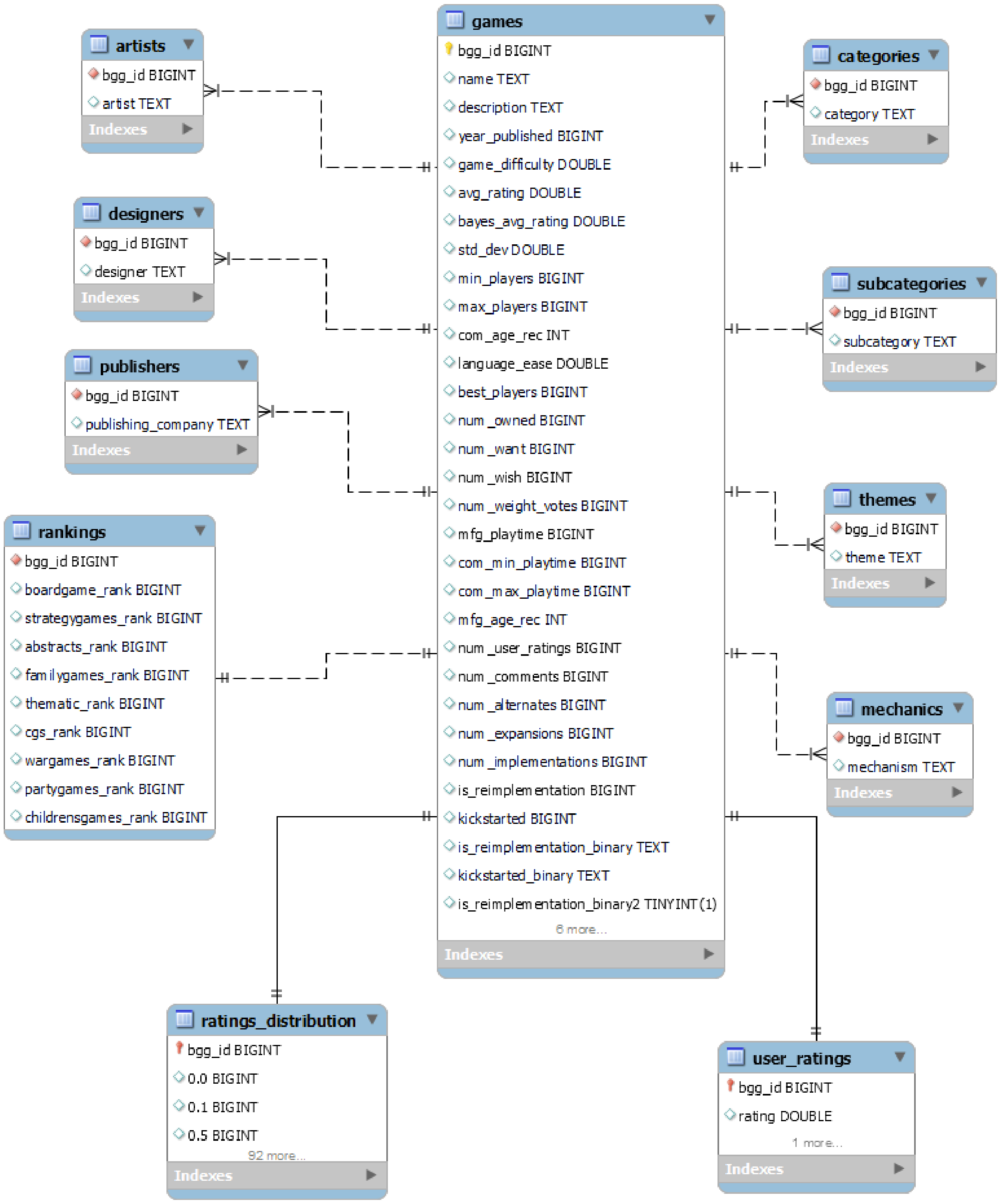
```python
# 🌐 Creation of SQLAlchemy engine
engine = sqlalchemy.create_engine(f"mysql+pymysql://root:{password}@127.0.0.1/{db_name}")
```



```python
# 🚀 Sending the DataFrames to MySQL
for table_name, df in dataframes.items():
    try:
        df.to_sql(name=table_name, con=engine, if_exists="replace", index=False)
        print(f"✅ Table {table_name} importée avec succès !")
    except Exception as e:
        print(f"❌ Erreur lors de l'importation de {table_name}: {e}")


# 🚪 Closing of MySQL connection
cursor.close()
conn.close()
```

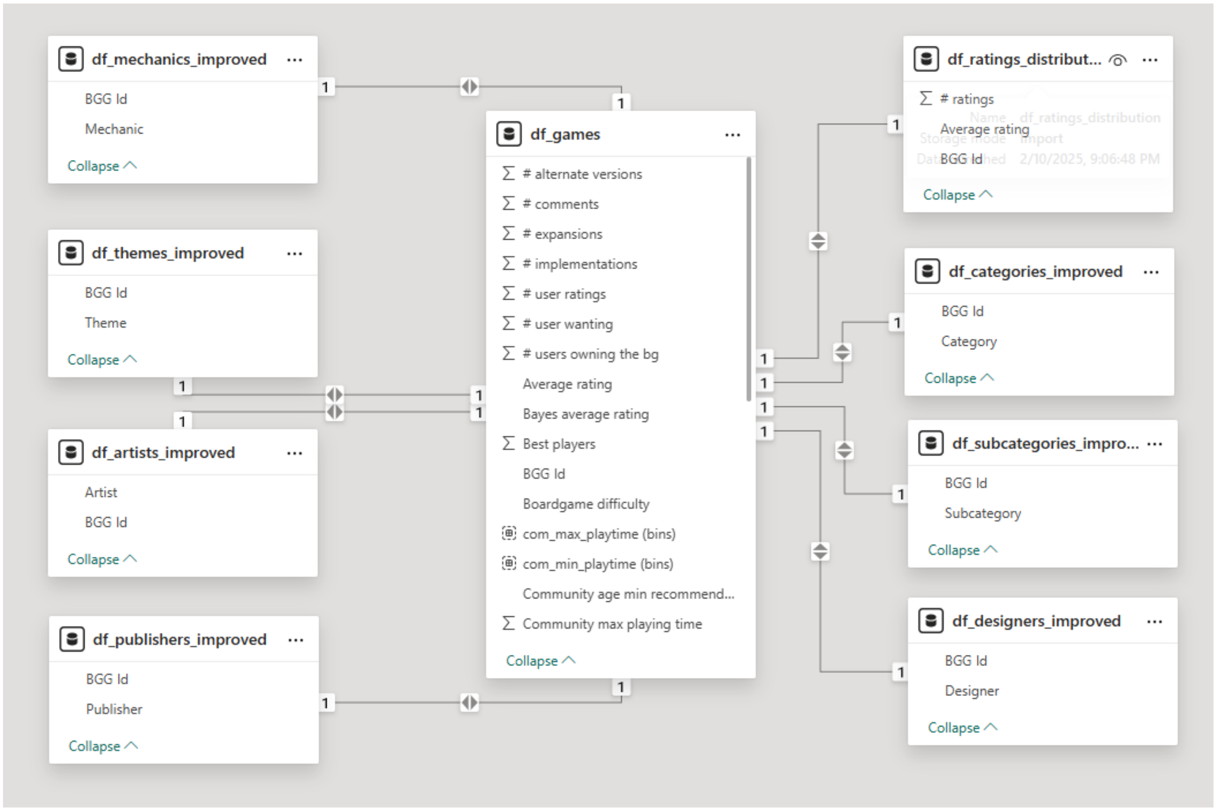
Entities, ERD

MySQL ERD



Power BI data model

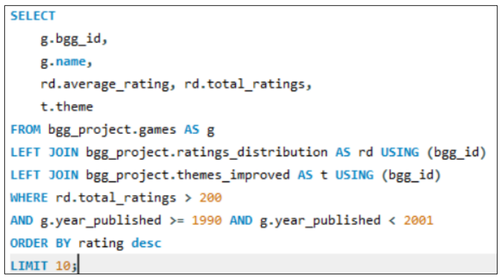
Here I didn't add user_ratings and rankings table (useless for me) but of course they could be added if necessary



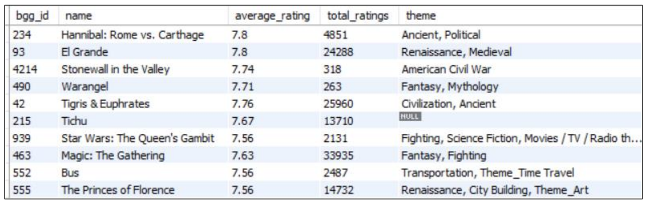
SQL queries

Examples of queries in MySQL (all queries available on the .sql file in the repository)

- Top 10 best board games in the 90's:

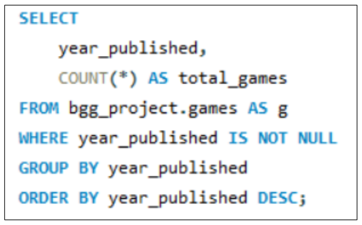


```sql
SELECT
    g.bgg_id,
    g.name,
    rd.average_rating, rd.total_ratings,
    t.theme
FROM bgg_project.games AS g
LEFT JOIN bgg_project.ratings_distribution AS rd USING (bgg_id)
LEFT JOIN bgg_project.themes_improved AS t USING (bgg_id)
WHERE rd.total_ratings > 200
AND g.year_published >= 1990 AND g.year_published < 2001
ORDER BY rating desc
LIMIT 10;
```

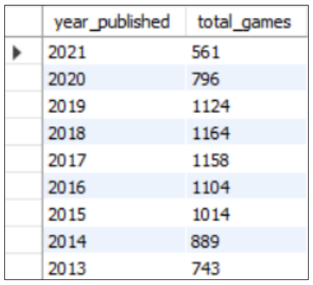


bgg_id	name	average_rating	total_ratings	theme
234	Hannibal: Rome vs. Carthage	7.8	4851	Ancient, Political
93	El Grande	7.8	24288	Renaissance, Medieval
4214	Stonewall in the Valley	7.74	318	American Civil War
490	Warangel	7.71	263	Fantasy, Mythology
42	Tigris & Euphrates	7.76	25960	Civilization, Ancient
215	Tichu	7.67	13710	NULL
939	Star Wars: The Queen's Gambit	7.56	2131	Fighting, Science Fiction, Movies / TV / Radio th...
463	Magic: The Gathering	7.63	33935	Fantasy, Fighting
552	Bus	7.56	2487	Transportation, Theme_Time Travel
555	The Princes of Florence	7.56	14732	Renaissance, City Building, Theme_Art

- Evolution of the number of board games published over the years

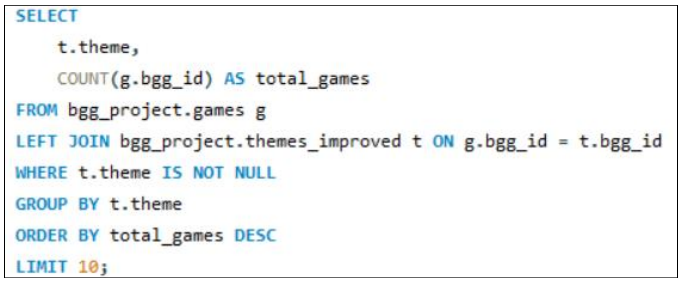


```sql
SELECT
    year_published,
    COUNT(*) AS total_games
FROM bgg_project.games AS g
WHERE year_published IS NOT NULL
GROUP BY year_published
ORDER BY year_published DESC;
```

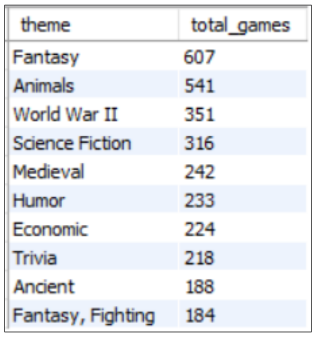


year_published	total_games
2021	561
2020	796
2019	1124
2018	1164
2017	1158
2016	1104
2015	1014
2014	889
2013	743

- The most popular themes

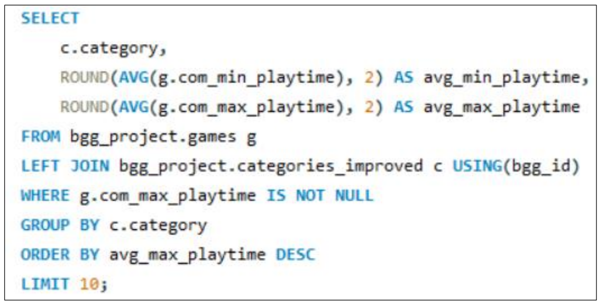


```sql
SELECT
    t.theme,
    COUNT(g.bgg_id) AS total_games
FROM bgg_project.games g
LEFT JOIN bgg_project.themes_improved t ON g.bgg_id = t.bgg_id
WHERE t.theme IS NOT NULL
GROUP BY t.theme
ORDER BY total_games DESC
LIMIT 10;
```

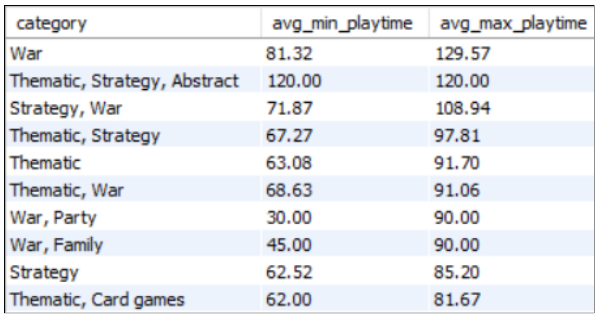


theme	total_games
Fantasy	607
Animals	541
World War II	351
Science Fiction	316
Medieval	242
Humor	233
Economic	224
Trivia	218
Ancient	188
Fantasy, Fighting	184

- Average playtime per board game category

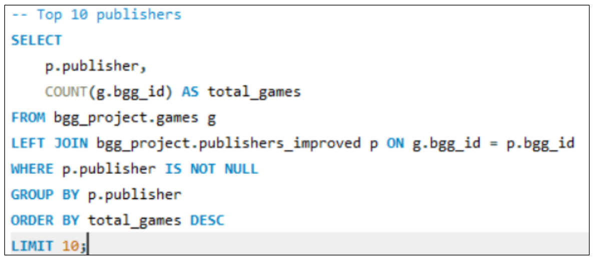


```sql
SELECT
    c.category,
    ROUND(AVG(g.com_min_playtime), 2) AS avg_min_playtime,
    ROUND(AVG(g.com_max_playtime), 2) AS avg_max_playtime
FROM bgg_project.games g
LEFT JOIN bgg_project.categories_improved c USING(bgg_id)
WHERE g.com_max_playtime IS NOT NULL
GROUP BY c.category
ORDER BY avg_max_playtime DESC
LIMIT 10;
```

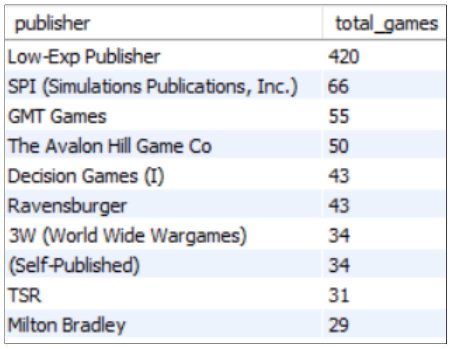


category	avg_min_playtime	avg_max_playtime
War	81.32	129.57
Thematic, Strategy, Abstract	120.00	120.00
Strategy, War	71.87	108.94
Thematic, Strategy	67.27	97.81
Thematic	63.08	91.70
Thematic, War	68.63	91.06
War, Party	30.00	90.00
War, Family	45.00	90.00
Strategy	62.52	85.20
Thematic, Card games	62.00	81.67

- Best publishers



```
-- Top 10 publishers
SELECT
    p.publisher,
    COUNT(g.bgg_id) AS total_games
FROM bgg_project.games g
LEFT JOIN bgg_project.publishers_improved p ON g.bgg_id = p.bgg_id
WHERE p.publisher IS NOT NULL
GROUP BY p.publisher
ORDER BY total_games DESC
LIMIT 10;
```

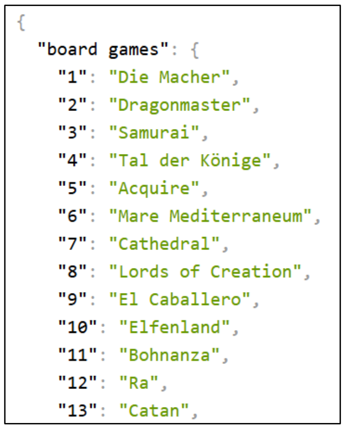


publisher	total_games
Low-Exp Publisher	420
SPI (Simulations Publications, Inc.)	66
GMT Games	55
The Avalon Hill Game Co	50
Decision Games (I)	43
Ravensburger	43
3W (World Wide Wargames)	34
(Self-Published)	34
TSR	31
Milton Bradley	29

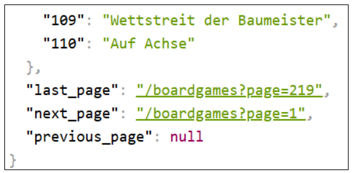
Create API

To expose a portion of the data from MySQL database, I created a local API with 2 different roots

Root 1: http://127.0.0.1:8080/boardgames
Provides the list of all the boardgames of my database by exposing their BGG Id and names

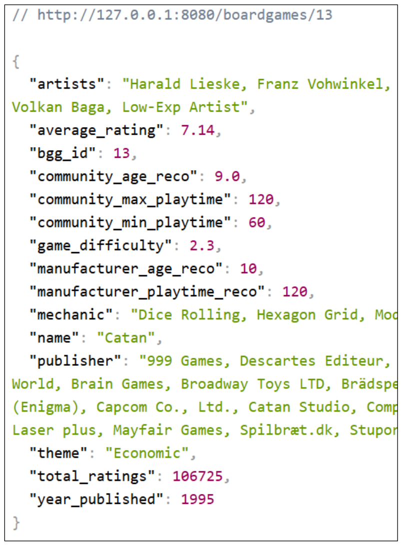


```
{
  "board games": {
    "1": "Die Macher",
    "2": "Dragonmaster",
    "3": "Samurai",
    "4": "Tal der Könige",
    "5": "Acquire",
    "6": "Mare Mediterraneum",
    "7": "Cathedral",
    "8": "Lords of Creation",
    "9": "El Caballero",
    "10": "Elfenland",
    "11": "Bohnanza",
    "12": "Ra",
    "13": "Catan",
```



```
    "109": "Wettstreit der Baumeister",
    "110": "Auf Achse"
  },
  "last_page": "/boardgames?page=219",
  "next_page": "/boardgames?page=1",
  "previous_page": null
}
```

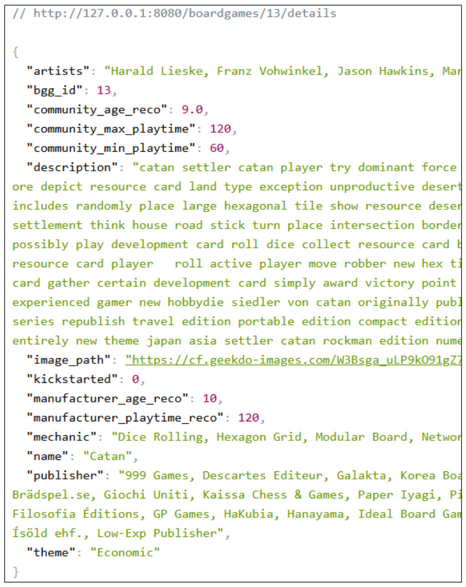
We can also retrieve some information about a given board game by adding its id number.
http://127.0.0.1:8080/_boardgames/<int:bgg_id>



```
// http://127.0.0.1:8080/boardgames/13

{
  "artists": "Harald Lieske, Franz Vohwinkel,
Volkan Baga, Low-Exp Artist",
  "average_rating": 7.14,
  "bgg_id": 13,
  "community_age_reco": 9.0,
  "community_max_playtime": 120,
  "community_min_playtime": 60,
  "game_difficulty": 2.3,
  "manufacturer_age_reco": 10,
  "manufacturer_playtime_reco": 120,
  "mechanic": "Dice Rolling, Hexagon Grid, Mod
  "name": "Catan",
  "publisher": "999 Games, Descartes Editeur,
World, Brain Games, Broadway Toys LTD, Brädspe
(Enigma), Capcom Co., Ltd., Catan Studio, Comp
Laser plus, Mayfair Games, Spilbræt.dk, Stupor
  "theme": "Economic",
  "total_ratings": 106725,
  "year_published": 1995
}
```

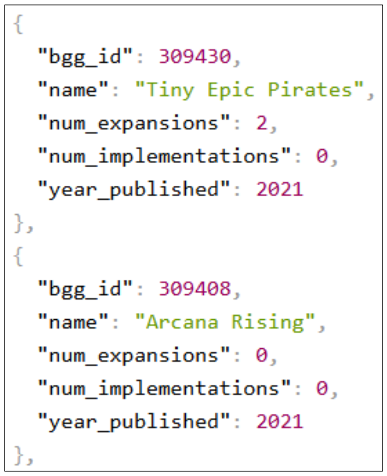
In addition to that, we can have more details such as game description, age recommendations, artists, mechanic or image link by adding /details to the url

http://127.0.0.1:8080/boardgames/<int:bgg_id>/details



```
// http://127.0.0.1:8080/boardgames/13/details

{
  "artists": "Harald Lieske, Franz Vohwinkel, Jason Hawkins, Mar
  "bgg_id": 13,
  "community_age_reco": 9.0,
  "community_max_playtime": 120,
  "community_min_playtime": 60,
  "description": "catan settler catan player try dominant force
ore depict resource card land type exception unproductive desert
includes randomly place large hexagonal tile show resource deser
settlement think house road stick turn place intersection border
possibly play development card roll dice collect resource card b
resource card player   roll active player move robber new hex ti
card gather certain development card simply award victory point
experienced gamer new hobbydie siedler von catan originally publ
series republish travel edition portable edition compact editio
entirely new theme japan asia settler catan rockman edition nume
  "image_path": "https://cf.geekdo-images.com/W3Bsga_uLP9kO91gZ7
  "kickstarted": 0,
  "manufacturer_age_reco": 10,
  "manufacturer_playtime_reco": 120,
  "mechanic": "Dice Rolling, Hexagon Grid, Modular Board, Networ
  "name": "Catan",
  "publisher": "999 Games, Descartes Editeur, Galakta, Korea Boa
Brädspel.se, Giochi Uniti, Kaissa Chess & Games, Paper Iyagi, Pi
Filosofia Éditions, GP Games, HaKubia, Hanayama, Ideal Board Gam
Ísöld ehf., Low-Exp Publisher",
  "theme": "Economic"
}
```

Root 2: http://127.0.0.1:8080/boardgames/kickstarted
Additional information for board games being kickstarted



```
{
  "bgg_id": 309430,
  "name": "Tiny Epic Pirates",
  "num_expansions": 2,
  "num_implementations": 0,
  "year_published": 2021
},
{
  "bgg_id": 309408,
  "name": "Arcana Rising",
  "num_expansions": 0,
  "num_implementations": 0,
  "year_published": 2021
},
```

Initial conclusions

Insights from the project tend to prove that certain factors may have an influence on the rating given to a board game.

- A balanced play time (not going above 90 minutes but doing at least 30 minutes)
- A board game being accessible from 8 years old, but with a good
- A good difficulty (being at least 3.5 on 5) enhances better chances of having a good rating
- A board game being released first on Kickstarter may have more chances to obtain a better rating

A theorical thought would be to state that big publishing companies or experienced artists or designers can have a major impact on board game notoriety and so on its rating and commercial success. We’ll try to see if Machine learning can assert or disprove it.

Another assumption is that certain themes or mechanics may also have an impact on the pleasure of playing and then on the rating given to a board game.

GDPR

Upon thorough examination of the data collected for this project, I confirm that no personal data was utilized throughout the project. All data sources used are publicly available at a country level, ensuring transparency and compliance with General Data Protection Regulation (GDPR) guidelines.

References

Flat Files:

- https://www.kaggle.com/datasets/threnjen/board-games-database-from-boardgamegeek

API:

- https://boardgamegeek.com/wiki/page/BGG_XML_API#

Web Scraping:

- https://boardgamegeek.com/browse/boardgame
- https://boardgamegeek.com/browse/boardgame/page/1

Notion board (task manager):

- https://micelbaz.notion.site/19d67c6c155b457aa625379f193f85cb?v=5a91fc24f6ba4bb38cbbe9f8a39b6727

GitHub repository (in progress):

- https://github.com/Mike578/board_games_project