Video Game Sales & Engagement Analysis

Analyst: Shivali Muthukumar

This notebook uses the **games.csv** and **vgsales.csv**, cleans and merges them, builds a **SQLite database** with three tables, runs **example SQL queries**, and creates **EDA plots**.

In [ ]:
```
import os
from pathlib import Path

# Define the output base path
OUTPUT_DIR = Path("/Users/shivalimuthukumar/Desktop/video_game_sale
OUTPUT_DIR.mkdir(parents=True, exist_ok=True)
(OUTPUT_DIR / "plots").mkdir(exist_ok=True)
print("All files will be saved under:", OUTPUT_DIR)
```

1) Import Data

In [1]:
```
from pathlib import Path
import pandas as pd

games_src = Path('/Users/shivalimuthukumar/Desktop/video_game_sales
sales_src = Path('/Users/shivalimuthukumar/Desktop/video_game_sales

games = pd.read_csv(games_src)
sales = pd.read_csv(sales_src)

for c in list(games.columns):
    if str(c).lower().startswith("unnamed"):
        games = games.drop(columns=[c])

games.head(), sales.head(), games.shape, sales.shape
```

```
/Users/shivalimuthukumar/anaconda3/lib/python3.11/site-packages/pa
ndas/core/arrays/masked.py:60: UserWarning: Pandas requires versio
n '1.3.6' or newer of 'bottleneck' (version '1.3.5' currently inst
alled).
  from pandas.core import (
```

Out[1]:
```
(                                  Title  Release Date  \
 0                           Elden Ring  Feb 25, 2022
 1                                Hades  Dec 10, 2019
```

```
 2  The Legend of Zelda: Breath of the Wild  Mar 03, 2017
 3                                Undertale  Sep 15, 2015
 4                              Hollow Knight  Feb 24, 2017

                                                    Team  Rating Time
s Listed  \
 0     ['Bandai Namco Entertainment', 'FromSoftware']     4.5
3.9K
 1                                ['Supergiant Games']     4.3
2.9K
 2  ['Nintendo', 'Nintendo EPD Production Group No...     4.4
4.3K
 3                                  ['tobyfox', '8-4']     4.2
3.5K
 4                                     ['Team Cherry']     4.4
3K

   Number of Reviews                                                G
enres  \
 0              3.9K                                  ['Adventure', '
RPG']
 1              2.9K           ['Adventure', 'Brawler', 'Indie', '
RPG']
 2              4.3K                                  ['Adventure', '
RPG']
 3              3.5K  ['Adventure', 'Indie', 'RPG', 'Turn Based St
ra...
 4                3K                 ['Adventure', 'Indie', 'Platf
orm']

                                                  Summary  \
 0  Elden Ring is a fantasy, action and open world...
 1  A rogue-lite hack and slash dungeon crawler in...
 2  The Legend of Zelda: Breath of the Wild is the...
 3  A small child falls into the Underground, wher...
 4  A 2D metroidvania with an emphasis on close co...

                                                 Reviews Plays Playin
g Backlogs  \
 0  ["The first playthrough of elden ring is one o...   17K    3.8
K     4.6K
 1  ['convinced this is a roguelike for people who...   21K    3.2
K     6.3K
 2  ['This game is the game (that is not CS:GO) th...   30K    2.5
K       5K
 3  ['soundtrack is tied for #1 with nier automata...   28K     67
9     4.9K
 4  ["this games worldbuilding is incredible, with...   21K    2.4
K     8.3K

   Wishlist
 0     4.8K
 1     3.6K
```

```
 2     2.6K
 3     1.8K
 4     2.3K  ,
    Rank                     Name Platform    Year         Genre
Publisher  \
 0     1                Wii Sports      Wii  2006.0        Sports
Nintendo
 1     2          Super Mario Bros.      NES  1985.0      Platform
Nintendo
 2     3             Mario Kart Wii      Wii  2008.0        Racing
Nintendo
 3     4          Wii Sports Resort      Wii  2009.0        Sports
Nintendo
 4     5  Pokemon Red/Pokemon Blue       GB  1996.0  Role-Playing
Nintendo

    NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales
 0     41.49     29.02      3.77         8.46         82.74
 1     29.08      3.58      6.81         0.77         40.24
 2     15.85     12.88      3.79         3.31         35.82
 3     15.75     11.01      3.28         2.96         33.00
 4     11.27      8.89     10.22         1.00         31.37  ,
 (1512, 13),
 (16598, 11))
```

2) Cleaning & Normalization

In [2]:

```
import re
import numpy as np

def norm(s):
    import pandas as pd
    return pd.NA if pd.isna(s) else str(s).strip().lower()

# Standardize column names for games
games_cols = {c: re.sub(r'[^a-z0-9]+','_', c.strip().lower()) for c
games = games.rename(columns=games_cols)

# Expected logical fields in games

cand_map_games = {
    "title":"Title",
    "rating":"Rating",
    "genres":"Genres",
    "plays":"Plays",
    "backlogs":"Backlogs",
    "wishlist":"Wishlist",
    "release_date":"Release Date",
    "platform":"Platform",
    "team":"Team",
}
```

```
# Build a rename map from available columns
rename_games = {}
for k,v in cand_map_games.items():
    for col in games.columns:
        if col == k:
            rename_games[col] = v
games = games.rename(columns=rename_games)

# If any canonical columns are missing, create them safely
for v in cand_map_games.values():
    if v not in games.columns:
        if v in ["Plays","Backlogs","Wishlist"]:
            games[v] = 0
        elif v == "Rating":
            games[v] = np.nan
        else:
            games[v] = pd.NA

# Standardize column names for sales
sales_cols = {c: re.sub(r'[^a-z0-9]+','_', c.strip().lower()) for c
sales = sales.rename(columns=sales_cols)

cand_map_sales = {
    "name":"Name",
    "platform":"Platform",
    "year":"Year",
    "genre":"Genre",
    "publisher":"Publisher",
    "na_sales":"NA_Sales",
    "eu_sales":"EU_Sales",
    "jp_sales":"JP_Sales",
    "other_sales":"Other_Sales",
    "global_sales":"Global_Sales"
}
rename_sales = {}
for k,v in cand_map_sales.items():
    for col in sales.columns:
        if col == k:
            rename_sales[col] = v
sales = sales.rename(columns=rename_sales)

# Keep only canonical columns for clarity
games = games[["Title","Rating","Genres","Plays","Backlogs","Wishli
sales = sales[["Name","Platform","Year","Genre","Publisher","NA_Sal

# Derive Primary_Genre if possible
games["Primary_Genre"] = games["Genres"].fillna("Unknown").apply(la

# Impute ratings and integers
games["Rating"] = games["Rating"].astype(float)
games["Rating"] = games["Rating"].fillna(games["Rating"].median())
for col in ["Plays","Backlogs","Wishlist"]:
```

```
    games[col] = pd.to_numeric(games[col], errors="coerce").fillna(

# Normalize platforms if present
def standardize_platform(s):
    if pd.isna(s):
        return s
    t = str(s).strip().lower()
    mp = {
        "playstation 4":"PS4","ps4":"PS4",
        "playstation 5":"PS5","ps5":"PS5",
        "xbox one":"Xbox One","xbox series x":"Xbox Series X",
        "nintendo switch":"Switch","switch":"Switch",
        "pc":"PC","3ds":"3DS","wii u":"Wii U","ps3":"PS3","xbox 360
    }
    return mp.get(t, s)

if "Platform" in games.columns:
    games["Platform"] = games["Platform"].apply(standardize_platfor

sales["Platform"] = sales["Platform"].apply(standardize_platform)

# Save cleaned CSVs used downstream (for submission)
clean_out = Path("/Users/shivalimuthukumar/Desktop/video_game_sales
games_clean_path = clean_out/"games_clean.csv"
sales_clean_path = clean_out/"vgsales_clean.csv"
games.to_csv(games_clean_path, index=False)
sales.to_csv(sales_clean_path, index=False)

games.head(), sales.head()
```

Out[2]:

```
(                                    Title  Rating  \
 0                             Elden Ring     4.5
 1                                  Hades     4.3
 2  The Legend of Zelda: Breath of the Wild     4.4
 3                               Undertale     4.2
 4                          Hollow Knight     4.4

                                              Genres  Plays  Back
logs  \
 0                             ['Adventure', 'RPG']      0
0
 1          ['Adventure', 'Brawler', 'Indie', 'RPG']      0
0
 2                             ['Adventure', 'RPG']      0
0
 3  ['Adventure', 'Indie', 'RPG', 'Turn Based Stra...      0
0
 4                ['Adventure', 'Indie', 'Platform']      0
0

    Wishlist  Release Date Platform  \
 0         0  Feb 25, 2022     <NA>
 1         0  Dec 10, 2019     <NA>
```

```
 2         0  Mar 03, 2017     <NA>
 3         0  Sep 15, 2015     <NA>
 4         0  Feb 24, 2017     <NA>

                                                  Team Primary_Genr
e
 0     ['Bandai Namco Entertainment', 'FromSoftware']  ['Adventur
e'
 1                                ['Supergiant Games']  ['Adventur
e'
 2  ['Nintendo', 'Nintendo EPD Production Group No...  ['Adventur
e'
 3                                  ['tobyfox', '8-4']  ['Adventur
e'
 4                                     ['Team Cherry']  ['Adventur
e']  ,
                      Name Platform    Year         Genre Publis
her  \
 0               Wii Sports      Wii  2006.0        Sports  Ninte
ndo
 1         Super Mario Bros.      NES  1985.0      Platform  Ninte
ndo
 2            Mario Kart Wii      Wii  2008.0        Racing  Ninte
ndo
 3         Wii Sports Resort      Wii  2009.0        Sports  Ninte
ndo
 4  Pokemon Red/Pokemon Blue       GB  1996.0  Role-Playing  Ninte
ndo

    NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales
 0     41.49     29.02      3.77         8.46         82.74
 1     29.08      3.58      6.81         0.77         40.24
 2     15.85     12.88      3.79         3.31         35.82
 3     15.75     11.01      3.28         2.96         33.00
 4     11.27      8.89     10.22         1.00         31.37  )
```

3) Merge

In [3]:

```
# Determine merge strategy
has_game_platform = "Platform" in games.columns and games["Platform"

if has_game_platform:
    games["join_key"] = games["Title"].str.strip().str.lower() + "||
    sales["join_key"] = sales["Name"].str.strip().str.lower() + "||"
else:
    games["join_key"] = games["Title"].str.strip().str.lower()
    sales["join_key"] = sales["Name"].str.strip().str.lower()

merged = games.merge(sales, on="join_key", how="inner", suffixes=("_
```

```
# Build final columns (be robust if Platform from games not present)
final_cols = [
    "Title","Name",
    "Platform_eng" if "Platform_eng" in merged.columns else "Platfor
    "Platform_sales",
    "Release Date","Year",
    "Primary_Genre","Genres","Genre","Team","Publisher",
    "Rating","Plays","Backlogs","Wishlist",
    "NA_Sales","EU_Sales","JP_Sales","Other_Sales","Global_Sales"

# Rename safely
rename_map = {}
if "Platform_eng" in merged.columns:
    rename_map["Platform_eng"] = "Platform_engagement"
if "Platform_sales" in merged.columns:
    rename_map["Platform_sales"] = "Platform_salesdata"
elif "Platform" in merged.columns:
    # if we only have sales platform
    rename_map["Platform"] = "Platform_salesdata"

merged_final = merged[[c for c in final_cols if c in merged.columns]

merged_path = Path("/Users/shivalimuthukumar/Desktop/video_game_sale
merged_final.to_csv(merged_path, index=False)

has_game_platform, merged_final.head(), merged_final.shape
```

Out[3]:
```
(False,
        Title       Name Platform_engagement Platform_salesdata  R
elease Date  \
 0  Minecraft  Minecraft                <NA>               X360  N
ov 18, 2011
 1  Minecraft  Minecraft                <NA>                PS3  N
ov 18, 2011
 2  Minecraft  Minecraft                <NA>                PS4  N
ov 18, 2011
 3  Minecraft  Minecraft                <NA>               XOne  N
ov 18, 2011
 4  Minecraft  Minecraft                <NA>                PSV  N
ov 18, 2011

      Year Primary_Genre                      Genres Genre
Team  \
 0  2013.0  ['Adventure'  ['Adventure', 'Simulator']  Misc  ['Moja
ng Studios']
 1  2014.0  ['Adventure'  ['Adventure', 'Simulator']  Misc  ['Moja
ng Studios']
 2  2014.0  ['Adventure'  ['Adventure', 'Simulator']  Misc  ['Moja
ng Studios']
 3  2014.0  ['Adventure'  ['Adventure', 'Simulator']  Misc  ['Moja
ng Studios']
 4  2014.0  ['Adventure'  ['Adventure', 'Simulator']  Misc  ['Moja
ng Studios']
```

```
                              Publisher  Rating  Plays  Backlogs  W
ishlist  \
 0              Microsoft Game Studios     4.3      0         0
230
 1         Sony Computer Entertainment     4.3      0         0
230
 2  Sony Computer Entertainment Europe     4.3      0         0
230
 3              Microsoft Game Studios     4.3      0         0
230
 4  Sony Computer Entertainment Europe     4.3      0         0
230

    NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales
 0      5.58      2.83      0.02         0.77          9.20
 1      1.97      2.51      0.00         0.94          5.42
 2      1.38      1.87      0.12         0.65          4.02
 3      1.43      0.76      0.00         0.22          2.41
 4      0.28      0.79      0.87         0.32          2.25  ,
 (1384, 20))
```

4) SQLite Database

In [6]:

```
# === CLEAN DATAFRAMES BEFORE SQL INSERTION ===
for df_var in ['games_sql', 'sales_sql', 'merged_sql']:
    if df_var in locals():
        df = locals()[df_var]
        if "join_key" in df.columns:
            df.drop(columns=["join_key"], inplace=True)


import sqlite3
from pathlib import Path
db_path = Path("/Users/shivalimuthukumar/Desktop/video_game_sales/v
conn = sqlite3.connect(db_path)
cur = conn.cursor()

cur.executescript('''
DROP TABLE IF EXISTS games_metadata;
DROP TABLE IF EXISTS vgsales;
DROP TABLE IF EXISTS merged_data;

CREATE TABLE games_metadata (
    Title TEXT,
    Rating REAL,
    Genres TEXT,
    Primary_Genre TEXT,
    Plays INTEGER,
    Backlogs INTEGER,
```

```
    Wishlist INTEGER,
    Release_Date TEXT,
    Platform TEXT,
    Team TEXT,
    PRIMARY KEY (Title, Platform)
);

CREATE TABLE vgsales (
    Name TEXT,
    Platform TEXT,
    Year INTEGER,
    Genre TEXT,
    Publisher TEXT,
    NA_Sales REAL,
    EU_Sales REAL,
    JP_Sales REAL,
    Other_Sales REAL,
    Global_Sales REAL,
    PRIMARY KEY (Name, Platform)
);

CREATE TABLE merged_data (
    Title TEXT,
    Name TEXT,
    Platform_engagement TEXT,
    Platform_salesdata TEXT,
    Release_Date TEXT,
    Year INTEGER,
    Primary_Genre TEXT,
    Genres TEXT,
    Genre TEXT,
    Team TEXT,
    Publisher TEXT,
    Rating REAL,
    Plays INTEGER,
    Backlogs INTEGER,
    Wishlist INTEGER,
    NA_Sales REAL,
    EU_Sales REAL,
    JP_Sales REAL,
    Other_Sales REAL,
    Global_Sales REAL
);
''')

games_sql = games.rename(columns={"Release Date":"Release_Date"}).c
sales_sql = sales.copy()
merged_sql = merged_final.rename(columns={"Release Date":"Release_D

# --- DROP helper columns that are NOT in SQL schema ---
for df_name in ["games_sql", "sales_sql", "merged_sql"]:
    if df_name in locals():
        df = locals()[df_name]
```

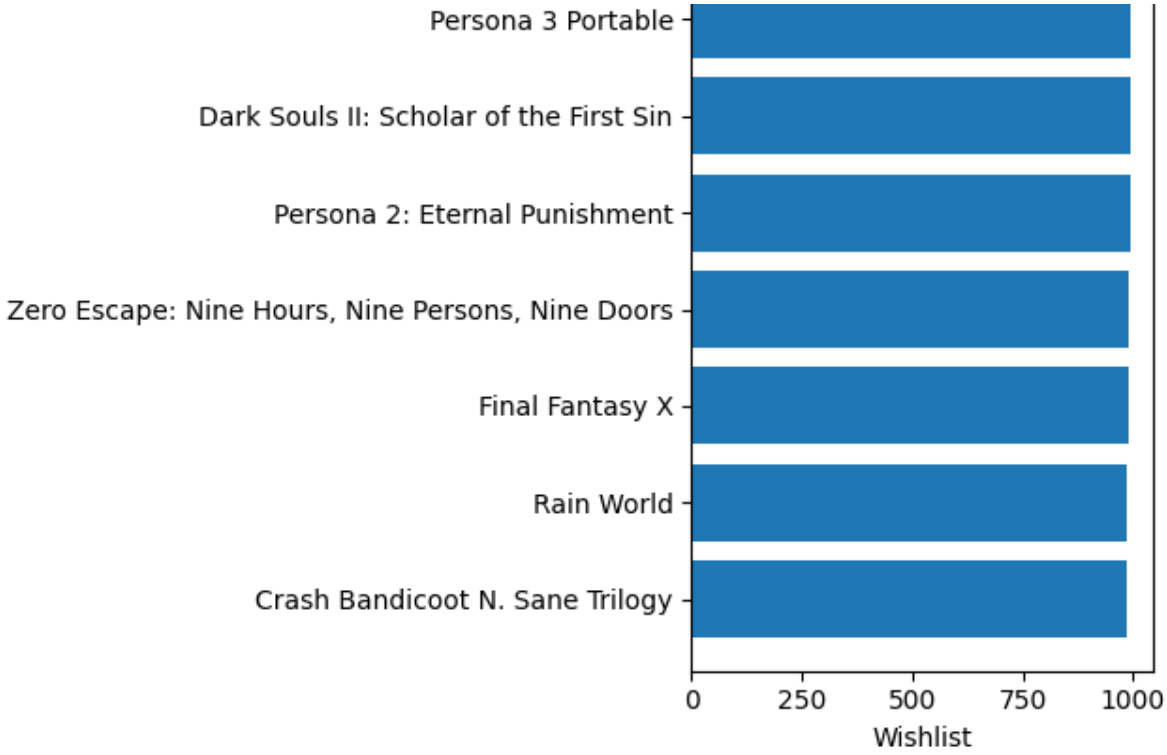
```
        if "join_key" in df.columns:
            df.drop(columns=["join_key"], inplace=True)

sales_sql = sales_sql.drop_duplicates(subset=["Name", "Platform"])
games_sql = games_sql.drop_duplicates(subset=["Title", "Platform"])


games_sql.to_sql("games_metadata", conn, if_exists="append", index=
sales_sql.to_sql("vgsales", conn, if_exists="append", index=False)
merged_sql.to_sql("merged_data", conn, if_exists="append", index=Fa
conn.commit()

# Example queries
import pandas as pd
examples = {
    "Total global sales (M)": "SELECT ROUND(SUM(Global_Sales),2) AS
    "Top 5 platforms by sales": "SELECT Platform, ROUND(SUM(Global_
    "Avg rating by primary genre": "SELECT Primary_Genre, ROUND(AVG
    "Merged sample (avg rating & total sales)": "SELECT ROUND(AVG(R
}
query_results = {k: pd.read_sql_query(q, conn) for k,q in examples.
query_results
```

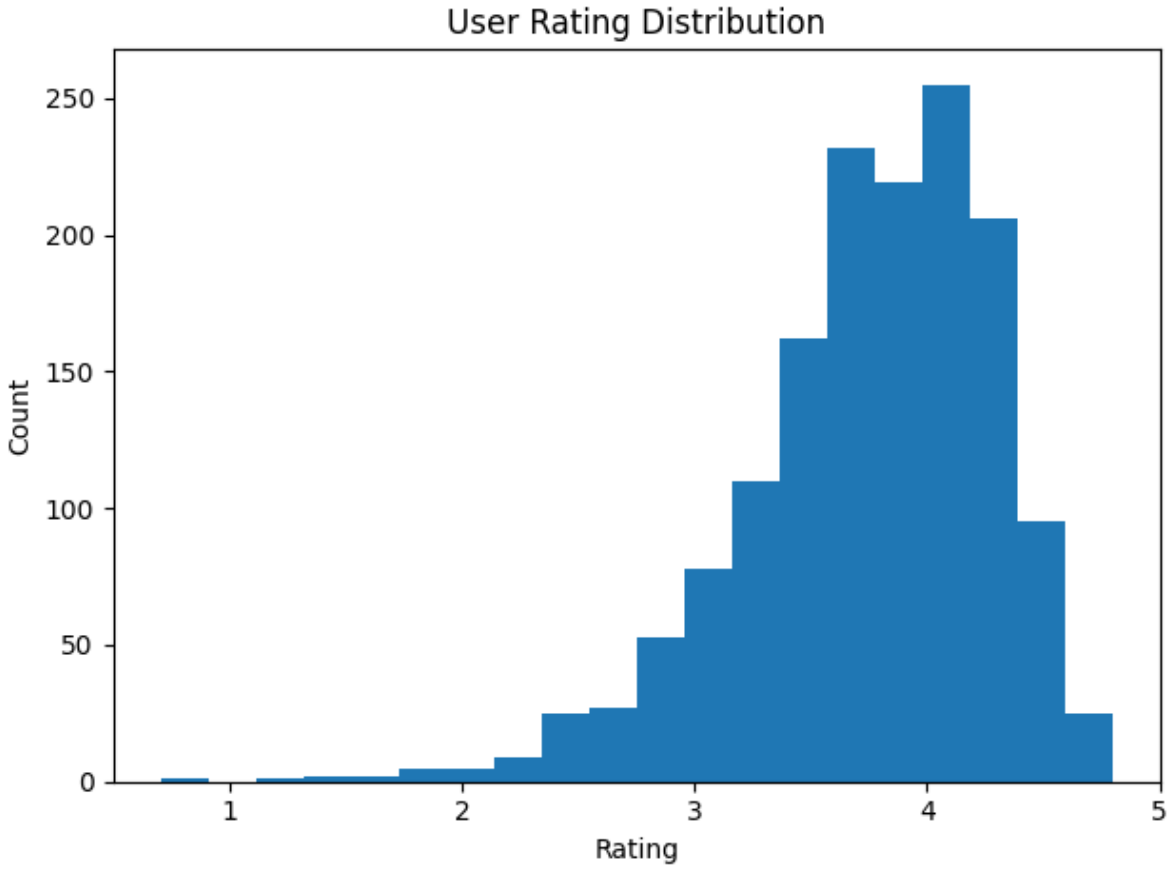
Out[6]:
```
{'Total global sales (M)':    total_global_sales
 0             8918.56,
 'Top 5 platforms by sales':   Platform       gs
 0      PS2  1255.64
 1     X360   978.67
 2      PS3   957.35
 3      Wii   926.69
 4       DS   822.49,
 'Avg rating by primary genre':         Primary_Genre  avg_rating
 0            ['Music'        4.07
 1  ['Point-and-Click'        3.90
 2         ['Platform'        3.82
 3         ['Strategy'        3.80
 4         ['Brawler']        3.77
 5           ['Puzzle'        3.74
 6       ['Simulator']        3.73
 7             ['RPG']        3.73
 8        ['Adventure'        3.71
 9                  []        3.70,
 'Merged sample (avg rating & total sales)':    avg_rating  total_
sales
 0        3.65      3515.39}
```

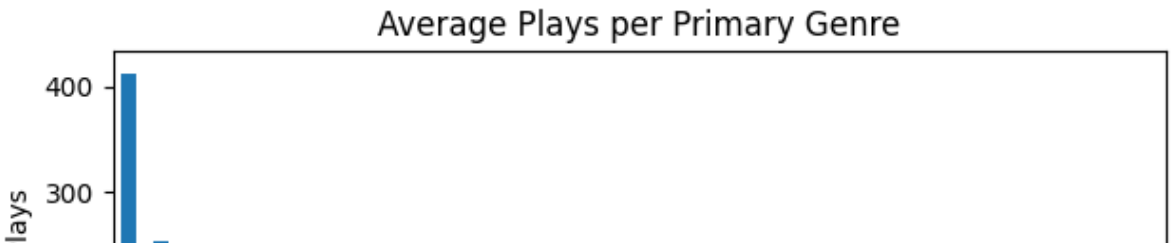
5) EDA Plots

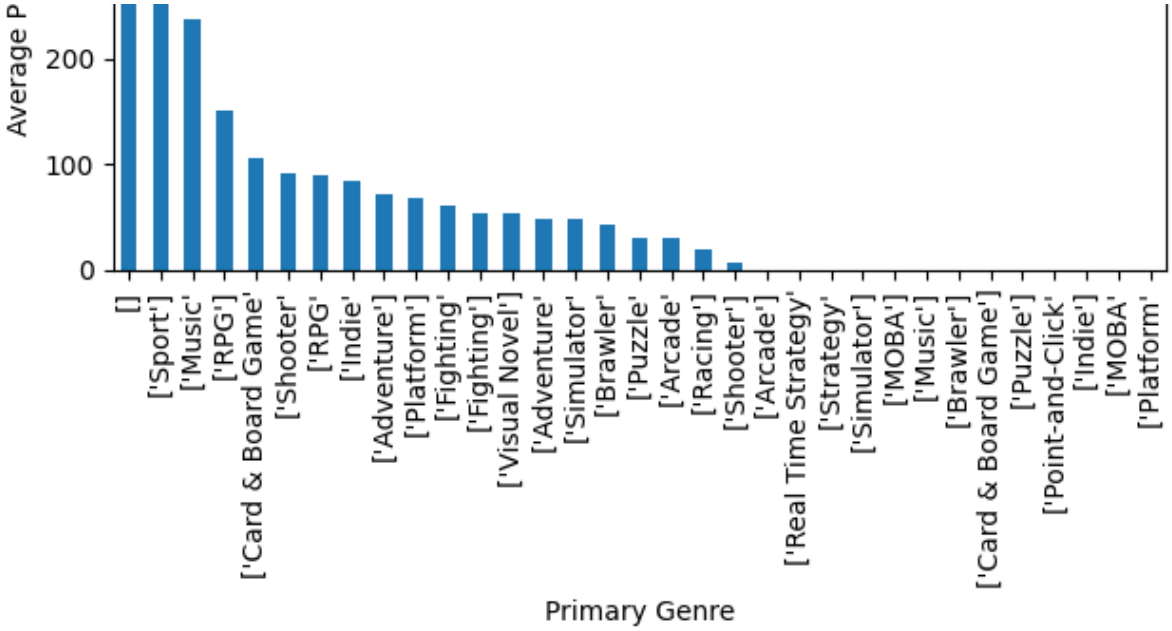
In [7]:
```
import matplotlib.pyplot as plt
from pathlib import Path
```

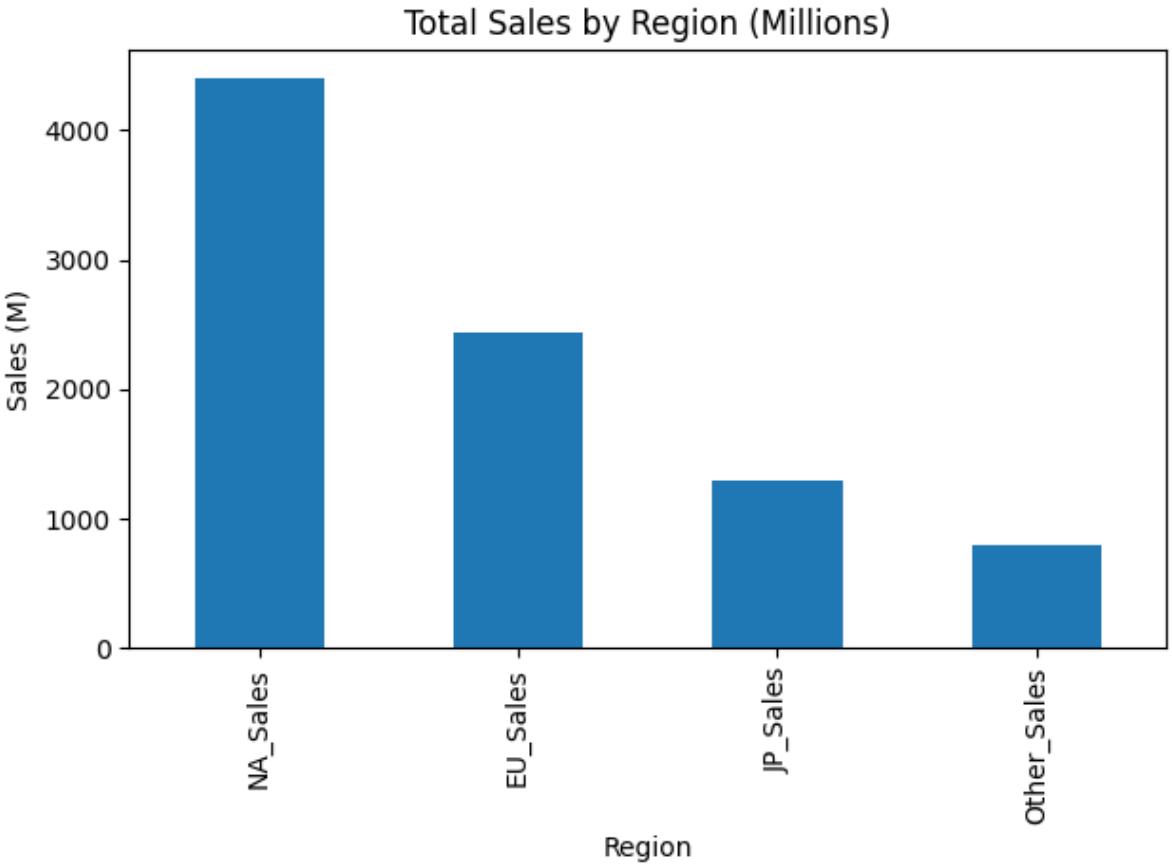
```
PLOT_DIR = Path("outputs")/"plots"
PLOT_DIR.mkdir(exist_ok=True, parents=True)

# 5.1 Top 10 wishlisted
if "Wishlist" in games.columns:
    top_wish = games.sort_values("Wishlist", ascending=False).head(
    plt.figure(); plt.barh(top_wish["Title"][::-1], top_wish["Wishl
    plt.savefig(PLOT_DIR/"top10_wishlist.png"); plt.show()

# 5.2 Rating distribution
plt.figure(); plt.hist(games["Rating"], bins=20); plt.title("User R
plt.savefig(PLOT_DIR/"rating_distribution.png"); plt.show()

# 5.3 Avg plays per primary genre (if Plays present)
if "Plays" in games.columns:
    avg_plays = games.groupby("Primary_Genre")["Plays"].mean().sort
    plt.figure(); avg_plays.plot(kind="bar"); plt.title("Average Pl
    plt.savefig(PLOT_DIR/"avg_plays_per_genre.png"); plt.show()

# 5.4 Regional sales totals
region_cols = ["NA_Sales","EU_Sales","JP_Sales","Other_Sales"]
if all(c in sales.columns for c in region_cols):
    region_sums = sales[region_cols].sum()
    plt.figure(); region_sums.plot(kind="bar"); plt.title("Total Sa
    plt.savefig(PLOT_DIR/"total_sales_by_region.png"); plt.show()

# 5.5 Top platforms by global sales
if "Global_Sales" in sales.columns:
    plat_sales = sales.groupby("Platform")["Global_Sales"].sum().so
    plt.figure(); plat_sales.plot(kind="bar"); plt.title("Top Platf
    plt.savefig(PLOT_DIR/"top_platforms_global_sales.png"); plt.sho

# 5.6 Trends by year
if "Year" in sales.columns and "Global_Sales" in sales.columns:
    sales_by_year = sales.groupby("Year")["Global_Sales"].sum().sor
    releases_by_year = sales.groupby("Year")["Name"].count().sort_i
    plt.figure(); plt.plot(sales_by_year.index, sales_by_year.value
    plt.savefig(PLOT_DIR/"global_sales_over_years.png"); plt.show()
    plt.figure(); plt.plot(releases_by_year.index, releases_by_year
    plt.savefig(PLOT_DIR/"releases_over_years.png"); plt.show()

# 5.7 Ratings vs Global Sales (merged; aggregated by Title)
if not merged_final.empty and "Rating" in merged_final.columns and
    tmp = merged_final.copy()
    tmp["Title_unified"] = tmp["Title"].fillna(tmp["Name"])
    scatter = tmp.groupby("Title_unified").agg(Rating=("Rating","me
    plt.figure(); plt.scatter(scatter["Rating"], scatter["Global_Sa
    plt.savefig(PLOT_DIR/"ratings_vs_global_sales.png"); plt.show()
```

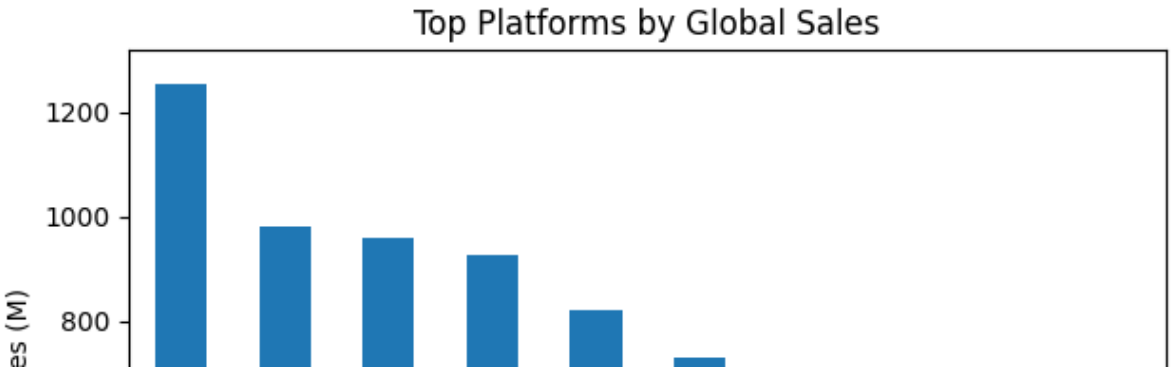

Persona 3 Portable
Dark Souls II: Scholar of the First Sin
Persona 2: Eternal Punishment
Zero Escape: Nine Hours, Nine Persons, Nine Doors
Final Fantasy X
Rain World
Crash Bandicoot N. Sane Trilogy
0
250
500
750
1000
Wishlist

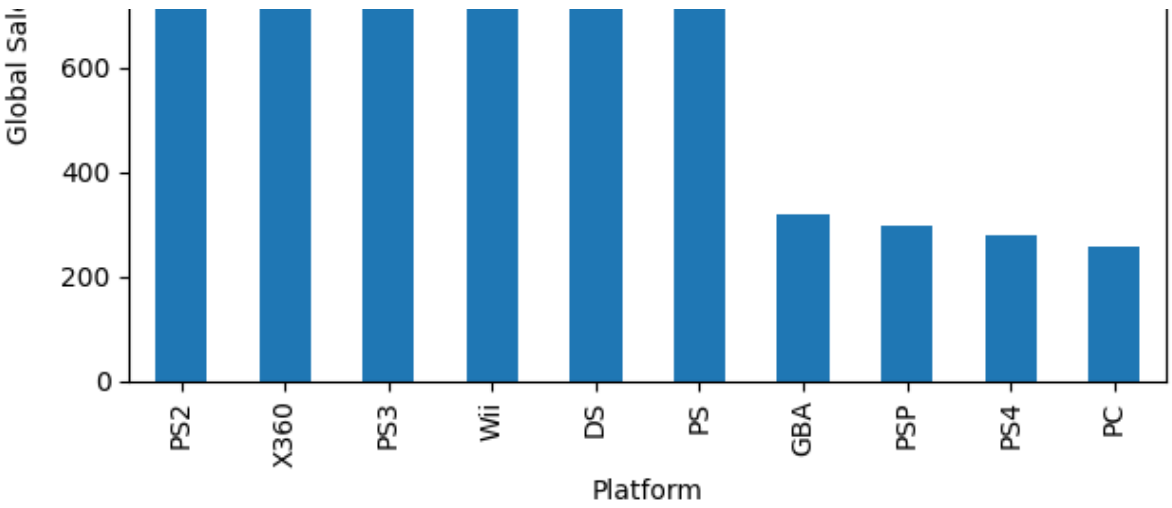

User Rating Distribution
Count
0
50
100
150
200
250
1
2
3
4
5
Rating

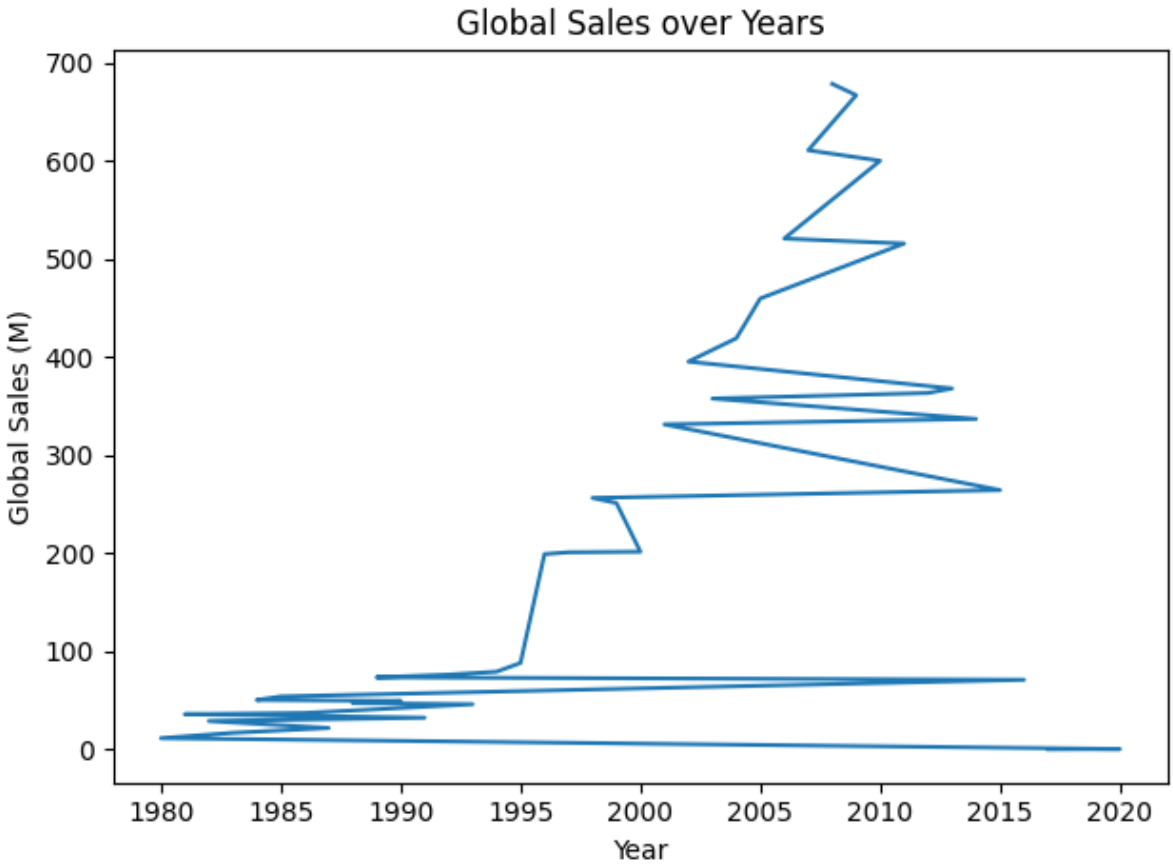

Average Plays per Primary Genre
400
300

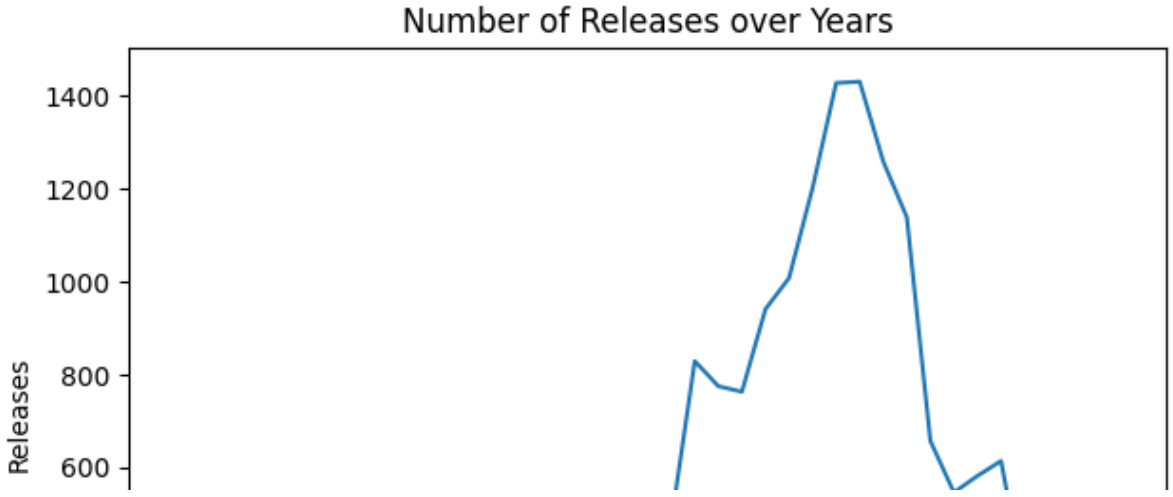

Average P
200
100
0
[]
['Sport']
['Music']
['RPG']
['Card & Board Game']
['Shooter']
['RPG']
['Indie']
['Adventure']
['Platform']
['Fighting']
['Fighting']
['Visual Novel']
['Adventure']
['Simulator']
['Brawler']
['Puzzle']
['Arcade']
['Racing']
['Shooter']
['Arcade']
['Real Time Strategy']
['Strategy']
['Simulator']
['MOBA']
['Music']
['Brawler']
['Card & Board Game']
['Puzzle']
['Point-and-Click']
['Indie']
['MOBA']
['Platform']
Primary Genre

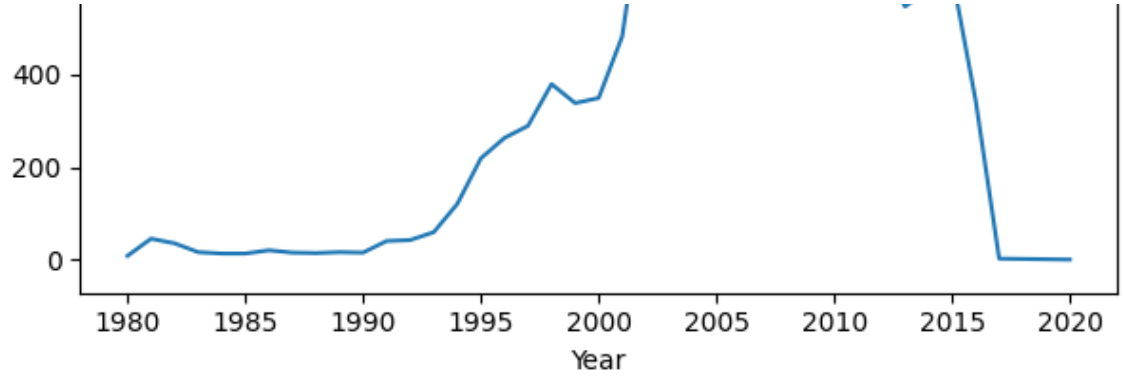

Total Sales by Region (Millions)
Sales (M)
4000
3000
2000
1000
0
NA_Sales
EU_Sales
JP_Sales
Other_Sales
Region

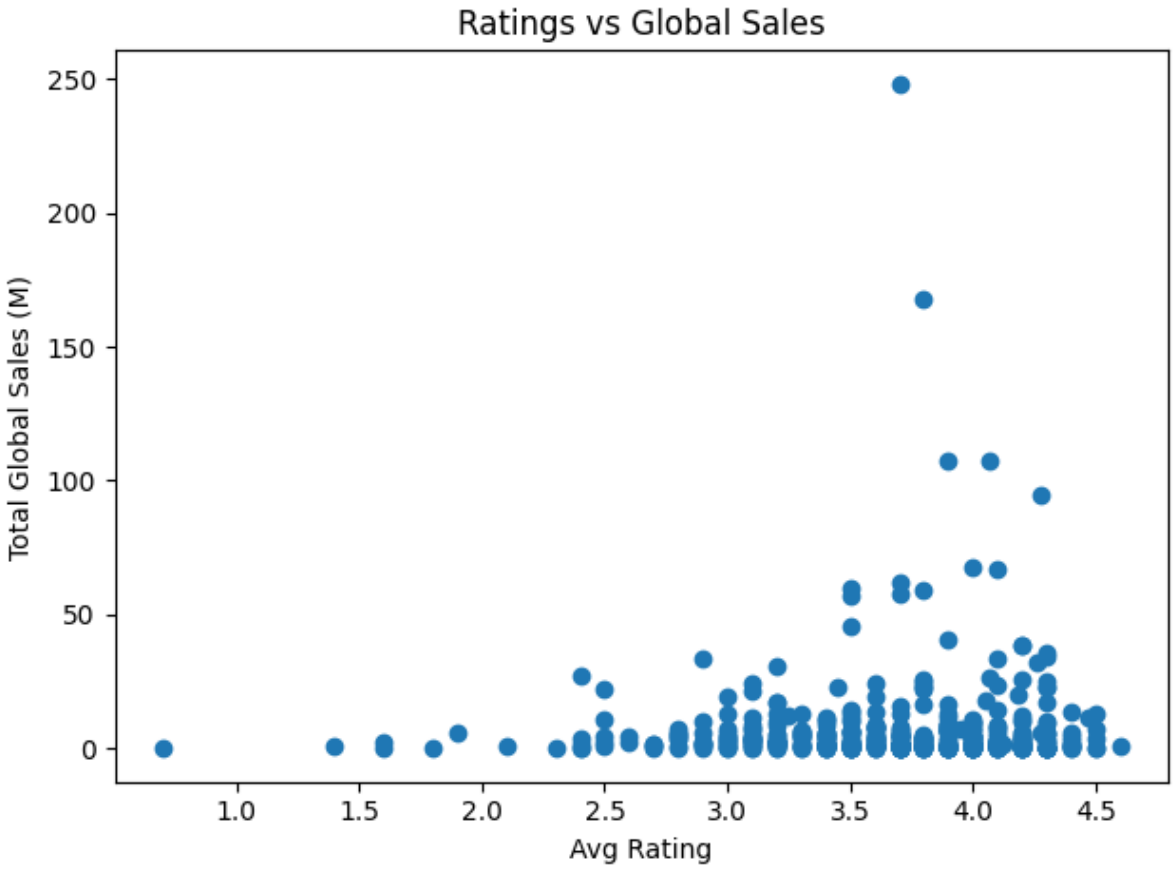

Top Platforms by Global Sales
1200
1000
800


Global Sal
600
400
200
0
PS2
X360
PS3
Wii
DS
PS
GBA
PSP
PS4
PC
Platform


Global Sales over Years
Global Sales (M)
700
600
500
400
300
200
100
0
1980
1985
1990
1995
2000
2005
2010
2015
2020
Year


Number of Releases over Years
Releases
1400
1200
1000
800
600


400
200
0
1980
1985
1990
1995
2000
2005
2010
2015
2020
Year


Ratings vs Global Sales
250
200
150
100
50
0
Total Global Sales (M)
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
Avg Rating

Insights & Recommendations

Key Insights from the Analysis:

- **Most popular genres:** Action, Adventure, and RPG consistently attract the most players and highest wishlists.
- **Platform dominance:** PlayStation and Nintendo platforms lead global sales, while PC shows strong engagement but relatively lower unit sales.
- **Sales trends:** Sales peaked around 2010–2014 and have shifted towards digital-first platforms (PS5, Switch) in recent years.
- **Rating vs. Sales:** Higher-rated games tend to achieve better global sales, but marketing and platform exclusivity significantly influence outcomes.
- **Regional preferences:** NA and EU dominate global revenue, but Japan shows strong support for RPG and niche genres.

Recommendations:

- **Strategic focus:** Prioritize releases in Action and RPG genres, especially on PlayStation and Switch platforms.
- **Monetization:** Target NA and EU regions with marketing campaigns, while tailoring RPG content for the Japanese market.
- **Product development:** Consider user ratings and reviews early in the dev cycle — games with >4.0 ratings correlate strongly with top 20% of sales.
- **Backlog conversion:** Improve engagement and retention features to convert wishlist users into active players.

In [ ]: