This code is a helper one. By the assist of this code I was able to arrange tags of games and add them to 4 readbale columns.

**Cell 1:**

This cell finds and arranges all overview.csv files in the location of all downloaded game data frame files.

import os

import pandas as pd

# Set the main folder path

main\_folder = '../data/raw/game lib'

# Create an empty list to hold each game's overview dataframe

df\_list = []

# Traverse through the folder and load all overview.csv files

for game\_folder in os.listdir(main\_folder):

    folder\_path = os.path.join(main\_folder, game\_folder)

    overview\_file = os.path.join(folder\_path, 'overview.csv')

    # Check if overview.csv exists

    if os.path.exists(overview\_file):

        # Read the CSV file and append to the list

        df\_overview = pd.read\_csv(overview\_file)

        df\_list.append(df\_overview)

# Concatenate all individual dataframes into one large dataframe

df\_mass\_game = pd.concat(df\_list, ignore\_index=True)

# Now you have a massive dataframe (df\_mass\_game) that contains all games' overview data

**Cell 2:**

This cell takes tags column from df\_mass\_game data frame, takes all words by seperating them using comma (,) and adds them to to list named tags\_list. If any word has aduplicate it does not add to list.

# We will create a massive text from tag column.

# it takes every word from every row and put it in a list.

# if it is already in the list, it will not be added again.

# then it will be converted to a string and saved to a text file.

# Create an empty list to hold all tags

tag\_list = []

# Traverse through the dataframe and extract all tags

for index, row in df\_mass\_game.iterrows():

    tags = row['tags'].split(',')

    for tag in tags:

        tag = tag.strip()

        if tag not in tag\_list:

            tag\_list.append(tag)

# Convert the list to a string

tag\_str = ', '.join(tag\_list)

# Save the string to a text file to location as tags.txt

with open('tags.txt', 'w') as f:

    f.write(tag\_str)

# Now you have a text file (tags.txt) that contains all unique tags from the overview data

tag\_list: "Online Co-Op, Survival, Early Access, Open World Survival Craft, Crafting, Base Building, Sci-fi, Co-op Campaign, Exploration, Multiplayer, FPS, Sandbox, First-Person, Simulation, Building, Science, Immersive Sim, Aliens, Adventure, PvE, Investigation, Mystery, 2D, Crime, Pixel Graphics, Story Rich, Visual Novel, Detective, Text-Based, Singleplayer, Choose Your Own Adventure, Great Soundtrack, Funny, Anime, Point & Click, Comedy, LGBTQ+, Strategy, RTS, Mythology, Action RTS, Fantasy, Top-Down, Magic, Level Editor, Moddable, PvP, Resource Management, Co-op, Tutorial, 3D, Psychological Horror, Lovecraftian, Horror, Action, Noir, Cinematic, Dark, Realistic, Third Person, Female Protagonist, Psychological, Supernatural, Survival Horror, Surreal, Thriller, Combat, Narration, Metroidvania, Puzzle, Puzzle Platformer, Platformer, Atmospheric, Relaxing, Cute, Nonlinear, Indie, Controller, Retro, Action Roguelike, Side Scroller, 2D Platformer, Souls-like, Hack and Slash, Roguelite, Action RPG, 2D Fighter, Roguelike, Action-Adventure, Cyberpunk, Conversation, Choices Matter, Linear, Multiple Endings, 3D Platformer, Cartoony, RPG, Dark Fantasy, Dark Humor, Open World, Lore-Rich, Underwater, Beat 'em up, Colony Sim, Character Customization, 2.5D, Medieval, Stylized, Board Game, Turn-Based, Hentai, Sexual Content, Party Game, Card Game, Casual, Family Friendly, Memes, Free to Play, Turn-Based Tactics, JRPG, Collectathon, Party-Based RPG, Colorful, Drama, Emotional, Shooter, Auto Battler, Inventory Management, Asynchronous Multiplayer, Roguelike Deckbuilder, Tactical, Card Battler, Loot, Replay Value, Hand-drawn, Cozy, Deckbuilding, Turn-Based Strategy, Difficult, Arcade, Procedural Generation, Vampire, City Builder, Life Sim, War, Music, Violent, Sports, BMX, Physics, Bikes, Skateboarding, Military, Stealth, VR, Gambling, Experimental, Blood, Dating Sim, Mature, Nudity, NSFW, Racing, Automobile Sim, Driving, Massively Multiplayer, 3D Vision, 6DOF, Management, Grand Strategy, Historical, 4X, Economy, Real-Time with Pause, Post-apocalyptic, CRPG, Dungeon Crawler, Time Management, Modern, Education, Heist, Looter Shooter, America, Team-Based, Gore, Walking Simulator, Hacking, Battle Royale, Extraction Shooter, Class-Based, Swordplay, Zombies, Dwarf, Bullet Hell, Top-Down Shooter, Arena Shooter, Space, Score Attack, Dragons, Artificial Intelligence, Trading, Romance, Football (American), Competitive, Third-Person Shooter, Local Co-Op, Local Multiplayer, Mechs, Turn-Based Combat, Political, Nature, Transportation, Offroad, World War II, eSports, Time Manipulation, Cooking, Job Simulator, Cartoon, Farming Sim, Farming, Agriculture, Mining, 1980s, 1990's, Robots, Automation, Voxel, Ninja, Spectacle fighter, Gun Customization, Isometric, Perma Death, Flight, Villain Protagonist, Capitalism, Real-Time, Psychedelic, Outbreak Sim, Minimalist, Immersive, 3D Fighter, Fighting, Character Action Game, Classic, Tower Defense, FMV, Interactive Fiction, Martial Arts, Narrative, Tactical RPG, Cats, Wholesome, Social Deduction, Real Time Tactics, Comic Book, Superhero, Wargame, Tanks, Alternate History, Diplomacy, Hex Grid, Logic, Old School, Basketball, 4 Player Local, Gothic, Dystopian, Transhumanism, Philosophical, MMORPG, Strategy RPG, Baseball, Creature Collector, Hero Shooter, Sniper, Idler, Vehicular Combat, Combat Racing, Vikings, Futuristic, Trains, Demons, Destruction, MOBA, God Game, Hunting, Dark Comedy, Parody, Satire, Space Sim, Split Screen, Clicker, Quick-Time Events, Conspiracy, Beautiful, Short, Word Game, Remake, Tennis, Soundtrack, Trading Card Game, Warhammer 40K, RPGMaker, Wrestling"

**Mid process:**

Before I start to code Cell 3, I take all the words collected from tag\_list and arrange them under 4 topics.

game type, style, mechanics and perspective

With help of an AI I gather them under related tag to be ready for 4 new columns.

**Cell 3:**

In this cell, I create 4 tag column and prepare them as python dictionary to add them overview datframe.

Each column has its matches.

Before arranging and adding volues to columns, first I check for all matches and if any column would look empty, the code will add dash ‘-’, so column would not has a Null value.

After this correction another level of code starts to fit suitable tag to its patent column to overview data frame.

After testing here I added this whole code to 03\_processing\_data.ipynb file. You can see its brief explanation inside that code ReadMe\_03 file.

# Define the tag categories with the new format

tag\_categories = {

    'tag game type': [

        'Action Roguelike', '2D Platformer', 'Hack and Slash', 'Roguelite', 'Action RPG', 'FPS', 'Multiplayer',

        'Open World', 'Survival', 'Crafting', 'Co-op Campaign', 'Base Building', 'Simulation', 'Exploration',

        'Visual Novel', 'Shooter', 'RPG', 'Strategy', 'Sandbox', 'Action-Adventure', 'Sports', 'Puzzle', 'Fighting',

        'Roguelike', 'Adventure', 'Card Game', 'RTS'

    ],

    'tag style': [

        'Souls-like', 'Cyberpunk', 'Story Rich', 'Multiple Endings', 'Anime', 'Pixel Graphics', 'Retro', 'Cartoony',

        'Fantasy', 'Dark Fantasy', 'Comedy', 'LGBTQ+', 'Realistic', 'Gothic', 'Mythology', 'Noir', 'Dystopian',

        'Sci-fi', 'Lovecraftian', 'Atmospheric', 'Cute'

    ],

    'tag mechanics': [

        'Side Scroller', 'Controller', 'Choices Matter', 'Crafting', 'Base Building', 'Co-op Campaign',

        'Turn-Based', 'Real-Time', 'Procedural Generation', 'Deckbuilding', 'Inventory Management',

        'Physics', 'Roguelike Deckbuilder', 'Tactical', 'PvP', 'PvE', 'Resource Management', 'Exploration', 'Combat',

        'Building', 'Moddable', 'Level Editor', 'Narration'

    ],

    'tag perspective': [

        '2D', '3D', 'Isometric', 'First-Person', 'Third Person', 'Top-Down', 'Side-Scrolling', 'Third-Person Shooter',

        'VR', 'Over-the-Shoulder', '3D Vision', '2.5D', 'First-Person Shooter'

    ]

}

# Function to assign tags to categories with a single '-' if no match found

def categorize\_tags(tag\_string, categories):

    tags = tag\_string.split(',')  # Split tags by commas

    categorized = {key: [] for key in categories}  # Prepare empty lists for each category

    for tag in tags:

        tag = tag.strip()  # Clean the tag

        found = False

        for category, keywords in categories.items():

            if tag in keywords:

                categorized[category].append(tag)

                found = True

        if not found:

            continue

    # Fill empty categories with a single '-'

    for category in categorized:

        if not categorized[category]:

            categorized[category] = ['-']  # Use single '-' if no tag matches

    return categorized

# Apply the function to df\_overview and create the new columns

for category in tag\_categories.keys():

    df\_overview[category] = df\_overview['tags'].apply(lambda x: ', '.join(categorize\_tags(x, tag\_categories)[category]) if pd.notnull(x) else '-')

# Drop the original 'tags' column

df\_overview = df\_overview.drop(columns=['tags'])