**Proyecto de evaluación Actividad 2**

Logotipo

Descripción generada automáticamente

**Estudiante: Juan Carlos González Ibarra**

**Correo:** [**jcarlos0284@gmail.com**](mailto:jcarlos0284@gmail.com)

**Fecha: 17 de mayo, 2024**

**Contenido**

1. Tema
   1. Planteamiento del problema
   2. Objetivos
2. Metodología de Desarrollo
3. Extracción de Datos

3.1 Fuentes de extracción de datos

3.2 Que es la extracción de datos

3.3 Extracción de datos con Python

1. Data Clean

4.1 Descripción de los datos

4.2 Carga de Datos

4.3 Preprocesar los datos

4.3.1 Fusionar data frames

4.3.2 Eliminar variables irrelevantes

4.4 Definición de la(s) Variable(s) Objetivo(s).

4.5 Almacenamiento de data frame en base de datos relacional

**4. Data Clean**

**4.1 Descripción de los datos**

Los data sets están en formato CSV son:

* consolas.csv
* consolas2.csv
* juegos\_mas\_vendidos.csv
* vgsales.csv
* video\_games.csv

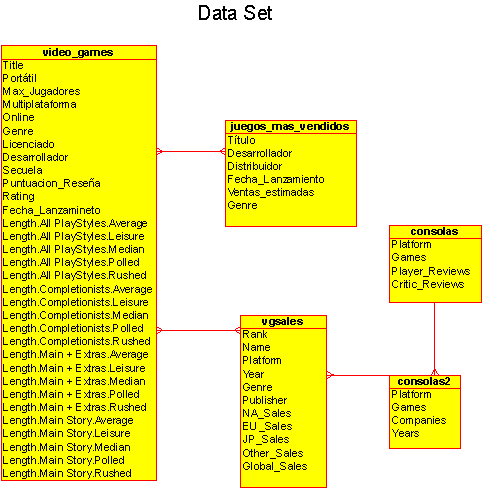
La información que contiene los archivos CSV es la siguiente:

**consolas.csv & consolas2.csv** : Contiene información de las plataformas de videojuego, juegos desarrollados, y críticas.

* Atributos:
  + **Platform**: Este atributo es de tipo string y describe el nombre de la consola de juegos para la cual se desarrollaron los juegos.
  + Games\_x: Este atributo es de tipo entero y representa la cantidad de juegos que se han desarrollado para la consola específica representada en la columna "Platform".
  + **Player Reviews**: Este atributo es de tipo entero y muestra la cantidad de comentarios o críticas realizadas por los jugadores acerca de la consola.
  + **Critic Reviews:** Este atributo es de tipo entero y muestra la cantidad de críticas realizadas por críticos profesionales acerca de la consola.
  + **Companies:** Este atributo también es una combinación de cadena de texto e integer y muestra la cantidad de compañías de desarrollo de videojuegos que han desarrollado títulos para la consola.
  + **Years:** Este atributo es una cadena de texto con datos enteros que indica el rango de años durante los cuales la consola ha estado en el mercado. Por ejemplo, un valor de "2001 - 2020" indicaría que la consola se lanzó en 2001 y se sigue utilizando hasta 2020.

**juegos\_mas\_vendidos.csv, vgsales.csv y video\_games.csv**: Contiene información de los juegos más vendidos por plataforma, genero, etc., y estadísticas del promedio de tiempo que pasa el jugador jugando.

* Atributos:
  + **'Titulo'**: tipo string El nombre del videojuego.
  + **'Compañia de desarrollo'**: tipo string La empresa o estudio que desarrolló el videojuego.
  + **'Fecha de Lanzamiento'**: tipo int La fecha en que el videojuego fue lanzado oficialmente al mercado.
  + '**Genero**': tipo string El tipo o categoría de juego al que pertenece el videojuego (ej. acción, aventura, estrategia, etc.).
  + '**Plataforma**': tipo string El sistema o dispositivo en el que el juego fue lanzado y puede ser jugado (ej. PlayStation, Xbox, PC, etc.).
  + **'Multiplataforma'**: tipo string Un indicador de si el videojuego está disponible en más de una plataforma.
  + **'Portatil'**: tipo bool Un indicador de si el videojuego está disponible en plataformas portátiles (ej. Nintendo Switch, PSP, etc.).
  + **'Online'**: tipo bool Un indicador de si el videojuego tiene funcionalidad de juego en línea.
  + **'Max Jugadores'**: tipo bool El número máximo de jugadores que pueden jugar el videojuego simultáneamente.
  + **'Rating'**: tipo int La clasificación de edad o recomendación de edad para el videojuego (ej. E para todos, T para adolescentes, M para maduro, etc.).
  + **'Ranking'**: tipo int El lugar que ocupa el videojuego en una lista ordenada, generalmente en términos de popularidad o ventas.
  + **'Puntuacion** Reseña': tipo float La calificación o puntuación promedio que los críticos de videojuegos han dado al videojuego.
  + **'Secuela'**: tipo bool Un indicador de si el videojuego es una secuela de un videojuego anterior.
  + **'Licenciado'**: tipo bool Un indicador de si el videojuego está basado en una licencia existente (ej. una película, una serie de televisión, un cómic, etc.).
  + **'Ventas** **NA'**: tipo float La cantidad de copias del videojuego que se han vendido en América del Norte.
  + **'Ventas** **EU'**: tipo float La cantidad de copias del videojuego que se han vendido en Europa.
  + **'Ventas** **JP'**: tipo float La cantidad de copias del videojuego que se han vendido en Japón.
  + **'Ventas** **Otros'**: tipo float La cantidad de copias del videojuego que se han vendido en otras regiones del mundo.
  + **'Ventas Globales'**: tipo float La cantidad total de copias del videojuego que se han vendido en todo el mundo.
  + **'Ventas Estimadas'**: tipo float Una estimación de la cantidad total de copias del videojuego que se venderán en el futuro, basada en las tendencias actuales de ventas.
  + **'Length.All PlayStyles.Average'**: tipo float Representa la duración promedio de juego teniendo en cuenta todos los estilos de juego.
  + **'Length.All PlayStyles.Leisure'**: tipo float Muestra la duración de juego para aquellos jugadores que disfrutan el juego de una manera más relajada o casual, abarcando todos los estilos de juego.
  + **'Length.All PlayStyles.Median'**: tipo float Es la mediana de la duración de juego, tomando en cuenta todos los estilos de juego.
  + **'Length.All PlayStyles.Polled'**: tipo float Número de observaciones o muestras utilizadas para calcular la duración promedio de todos los estilos de juego.
  + **'Length.All PlayStyles.Rushed'**: tipo float Refleja la duración de juego para aquellos jugadores que avanzan de manera más apresurada, considerando todos los estilos de juego.
  + **'Length.Completionists.Average'**: tipo float Es la duración promedio de juego para aquellos jugadores completistas, quienes se esfuerzan por alcanzar el 100% del juego.
  + **'Length.Completionists.Leisure'**: tipo float Muestra la duración del juego para los jugadores completistas que juegan de manera más relajada o casual.
  + **'Length.Completionists.Median'**: tipo float Es la mediana de la duración de juego para los jugadores completistas.
  + **'Length.Completionists.Polled'**: tipo float Número de observaciones o muestras utilizadas para calcular la duración promedio de los jugadores completistas.
  + **'Length.Completionists.Rushed'**: tipo float Representa la duración de juego para los jugadores completistas que avanzan a través del juego de una manera más apresurada.
  + **'Length.Main + Extras.Average'**: tipo float Muestra la duración promedio de juego cuando se juega la historia principal además de los contenidos extras.
  + **'Length.Main + Extras.Leisure'**: tipo float Duración del juego para aquellos jugadores que disfrutan la historia principal y los contenidos extras de una manera más relajada o casual.
  + **'Length.Main + Extras.Median':** tipo float Es la mediana de la duración del juego, considerando la historia principal y los contenidos extras.
  + **'Length.Main + Extras.Polled':** tipo float Número de observaciones o muestras utilizadas para calcular la duración promedio del juego principal más los contenidos extras.
  + **'Length.Main + Extras.Rushed'**: tipo float Refleja la duración de juego para aquellos jugadores que avanzan de manera más apresurada a través de la historia principal y los contenidos extras.
  + **'Length.Main Story.Average'**: tipo float Muestra la duración promedio de la historia principal del juego.
  + **'Length.Main Story.Leisure':** tipo float Representa la duración de la historia principal del juego para aquellos jugadores que la disfrutan de una manera más relajada o casual.
  + **'Length.Main Story.Median':** tipo float Es la mediana de la duración de la historia principal del juego.
  + **'Length.Main Story.Polled':** tipo float Número de observaciones o muestras utilizadas para calcular la duración promedio de la historia principal.
  + **'Length.Main Story.Rushed':** tipo float Indica la duración de la historia principal del juego para aquellos jugadores que la avanzan de manera más apresurada.



**4.2 Carga de Datos**

1. Instalación de módulos

# Instalación de módulos

!pip install pandas

!pip install matplotlib

!pip install networkx

!pip install -U scikit-learn

!pip install sqlalchemy

2. Importar módulos

# Importar modulos

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LinearRegression

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix

import seaborn as sns

from sklearn.metrics import accuracy\_score

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import OneHotEncoder

from sklearn.metrics import classification\_report, confusion\_matrix

from sklearn.compose import ColumnTransformer

3. Leer los archivos CSV

# Leer los archivos CSV

df\_juegos\_top = pd.read\_csv('data/juegos\_mas\_vendidos.csv', encoding='utf-8')

df\_consolas = pd.read\_csv('data/consolas.csv', encoding='utf-8')

df\_consolas2 = pd.read\_csv('data/consolas2.csv', encoding='utf-8')

df\_vgsales = pd.read\_csv('data/vgsales.csv', encoding='utf-8')

df\_video\_games = pd.read\_csv('data/video\_games.csv', encoding='utf-8')

**4.3 Preprocesar los datos**

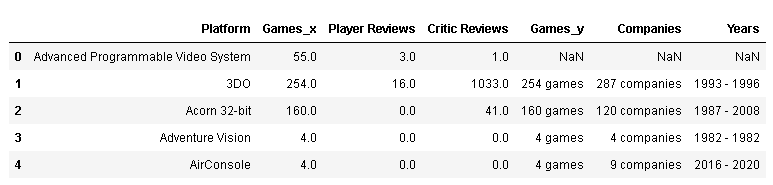
**4.3.1 Fusionar data frames**

Se van a fusionar dos DataFrames (df\_consolas y df\_consolas2), los atributos en común son "Platform" que representa el nombre de la Plataforma en que se ejecuta el Videojuego.

# Fusiona los DataFrames de Consolas

df\_consolas = pd.merge(df\_consolas, df\_consolas2, on='Platform', how='outer')

df\_consolas.head()



Se van a fusionar tres DataFrames (df\_juegos, df\_vgsales, df\_video\_games), los atributos en común son "Título" en df\_juegos, "Name" en df\_vgsales y "Title" en df\_video\_games que representa el nombre del videojuego.

# Primero, cambiamos los nombres de las columnas para que sean iguales en todos los DataFrames

df\_juegos\_top.rename(columns={'Título': 'Title'}, inplace=True)

df\_vgsales.rename(columns={'Name': 'Title'}, inplace=True)

# Se fusionas los DataFrames utilizando la columna 'Title' como atributo en comun

df\_juegos = df\_juegos\_top.merge(df\_vgsales, on='Title', how='outer')  # esto fusiona df\_juegos\_top y df\_vgsales

df\_juegos = df\_juegos.merge(df\_video\_games, on='Title', how='outer')  # esto añade df\_video\_games al DataFrame fusionado

# Muestra el DataFrame resultante

df\_juegos.head()



**4.3.2 Eliminar variables irrelevantes**

El DataFrame df\_consolas, se va eliminar el atributo 'Games\_y' ya que se repite con el atributo 'Games\_x', y se van a renombrar los atributos resultantes:

* 'Platform': 'Plataforma',
* 'Companies': 'Compañia',
* 'Games\_x': 'Juegos\_Desarrollados',
* 'Years': 'Anios',
* 'Player Reviews': 'Resenas\_Jugador',
* 'Critic Reviews': 'Cantidad\_Criticas'

Se van a remplazar todos los valores nulos (NaN) en el DataFrame de acuerdo con el tipo de dato como lo son:

* 'Plataforma' con 'Desconocido'.
* 'Compañia' con 'Desconocido'.
* 'Juegos\_Desarrollados con 0.
* 'Anios' con 0.
* 'Resenas\_Jugador' con 0.
* 'Cantidad\_Criticas' con 0.

# Elimina las columnas originales 'Games\_x' y 'Games\_y'

df\_consolas = df\_consolas.drop(['Games\_y'], axis=1)

# Extrae los números de las cadenas de texto en la columna 'Companies' y convierte estos números a enteros

df\_consolas['Companies'] = df\_consolas['Companies'].str.extract('(\d+)').astype(float)

# Reemplaza todos los valores nulos (NaN) en el DataFrame con 0

df\_consolas = df\_consolas.fillna(0)

df\_consolas = df\_consolas.rename(columns={

    'Platform': 'Plataforma',

    'Companies': 'Compañia',

    'Games\_x': 'Juegos\_Desarrollados',

    'Years': 'Anios',

    'Player Reviews': 'Resenas\_Jugador',

    'Critic Reviews': 'Cantidad\_Criticas'

})

# Reacomodar los atributos según el orden deseado, puedes utilizar el método reindex()

df\_consolas = df\_consolas.reindex(columns=['Plataforma', 'Compañia', 'Juegos\_Desarrollados', 'Anios', 'Resenas\_Jugador', 'Cantidad\_Criticas'])

df\_consolas['Plataforma'].fillna('Sin información', inplace=True)

df\_consolas['Compañia'].fillna('Sin información', inplace=True)

df\_consolas['Juegos\_Desarrollados'].fillna(0, inplace=True)

df\_consolas['Anios'].fillna(0, inplace=True)

df\_consolas['Resenas\_Jugador'].fillna(0, inplace=True)

df\_consolas['Cantidad\_Criticas'].fillna(0, inplace=True)

df\_consolas.head()



Del DataFrame **df\_juegos** se realiza un procesado más amplio de los datos:

* Se va a reemplazar los valores nulos en la columna 'Genre\_x' con los valores correspondientes de la columna 'Genre\_y'.
* Se va a eliminar la columna 'Genre\_y' del DataFrame, esto se hace porque ya se llenó los valores nulos en 'Genre\_x' con los valores correspondientes.
* Se van a eliminar las columnas: 'Fecha\_Lanzamiento', 'Unnamed: 11', 'Fecha\_Lanzamineto', 'Desarrollador\_y', 'Genre', 'Desarrollador\_x' y 'Distribuidor'.
* Se van a renombrar las columnas del DataFrame y se proporciona un diccionario en el que se especifica el nuevo nombre de cada columna.
* Se reorganizan las columnas para una mejor comprensión.
* Se define una lista llamada 'columnas' que contiene los nombres de las columnas que se van a modificar, y se va iterar sobre cada columna en la lista para reemplazar los valores nulos que tengan.
* Para los atributos de tipo string se va a remplazar los espacios vacios con por 'Desconocido'.
* Se define una lista llamada 'valores\_incorrectos' ('2007', '2009', '2010', '2011', '2012', '2013', '2014', '2015'), que contiene los valores que se van a reemplazar en el atributo Genero por 'Desconocido'.
* Para los atributos de tipo int o float se va a remplazar los espacios vacios con 0.
* Para los atributos de tipo bool se van a remplazar los espacios vacios con 'False'.

# los datos en Genre\_x y reemplazar los valores NaN con los valores correspondientes de Genre\_y

df\_juegos['Genre\_x'] = df\_juegos['Genre\_x'].fillna(df\_juegos['Genre\_y'])

df\_juegos = df\_juegos.drop(columns='Genre\_y')

df\_juegos.head()

Tabla

Descripción generada automáticamente

Se van a eliminar los atributos que no se utilizan en el análisis

# Se van a eliminar los atributos que no se utilizan en el analisis

df\_juegos = df\_juegos.drop(['Fecha\_Lanzamiento', 'Unnamed: 11', 'Fecha\_Lanzamineto', 'Desarrollador\_y','Genre', 'Desarrollador\_x', 'Distribuidor' ], axis=1)

Se renombran los atributos resultantes

# Se renombran los atributos resultantes

df\_juegos = df\_juegos.rename(columns={

    'Title': 'Titulo',

    'Genre\_x': 'Genero',

    'Rank': 'Ranking',

    'Platform': 'Plataforma',

    'Year': 'Fecha\_Lanzamiento',

    'Publisher': 'Compañia\_Desarrollo',

    'NA\_Sales': 'Ventas\_NA',

    'EU\_Sales': 'Ventas\_EU',

    'JP\_Sales': 'Ventas\_JP',

    'Other\_Sales': 'Ventas\_Otros',

    'Global\_Sales': 'Ventas\_Globales'

})

Reacomodo de atributos para una mejor comprensión

# Reacomodo de atributos para una mejor comprension

df\_juegos = df\_juegos[[

    'Titulo',

    'Compañia\_Desarrollo',

    'Fecha\_Lanzamiento',

    'Genero',

    'Plataforma',

    'Multiplataforma',

    'Portátil',

    'Online',

    'Max\_Jugadores',

    'Rating',

    'Ranking',

    'Puntuacion\_Reseña',

    'Secuela',

    'Licenciado',

    'Ventas\_NA',

    'Ventas\_EU',

    'Ventas\_JP',

    'Ventas\_Otros',

    'Ventas\_Globales',

    'Ventas\_estimadas',

    'Length.All PlayStyles.Average',

    'Length.All PlayStyles.Leisure',

    'Length.All PlayStyles.Median',

    'Length.All PlayStyles.Polled',

    'Length.All PlayStyles.Rushed',

    'Length.Completionists.Average',

    'Length.Completionists.Leisure',

    'Length.Completionists.Median',

    'Length.Completionists.Polled',

    'Length.Completionists.Rushed',

    'Length.Main + Extras.Average',

    'Length.Main + Extras.Leisure',

    'Length.Main + Extras.Median',

    'Length.Main + Extras.Polled',

    'Length.Main + Extras.Rushed',

    'Length.Main Story.Average',

    'Length.Main Story.Leisure',

    'Length.Main Story.Median',

    'Length.Main Story.Polled',

    'Length.Main Story.Rushed'

]]

Se crea el atributo columnas con los atributos de tipo string y se van a remplazar los espacios vacios con 'Desconocido'

# Se crea el atributo columnas con los atributos de tipo string

# Se van a remplazar los espacios vacios con 'Desconocido'

columnas = ['Titulo','Compañia\_Desarrollo','Genero','Plataforma','Rating']

for columna in columnas:

    df\_juegos[columna] = df\_juegos[columna].fillna('Desconocido')

    df\_juegos[columna] = df\_juegos[columna].astype(str)

La columna 'Titulo' de df\_juegos sólo contendrá el texto antes de los dos puntos para cada juego.

# este código, la columna 'Titulo' de df\_juegos sólo contendrá el texto antes de los dos puntos para cada juego.

df\_juegos['Titulo'] = df\_juegos['Titulo'].apply(lambda x: x.split(':')[0])

Reemplazar espacios en blanco con guiones bajos en la columna 'Titulo'

# Reemplazar espacios en blanco con guiones bajos en la columna 'Titulo'

df\_juegos['Titulo'] = df\_juegos['Titulo'].str.replace(' ', '\_')

Definir los valores a reemplazar en una lista por los valores en la columna 'Genero'

# Definir los valores a reemplazar en una lista

valores\_incorrectos = ['2007', '2009', '2010', '2011', '2012', '2013', '2014', '2015']

# Reemplazar los valores en la columna 'Genero'

df\_juegos['Genero'] = df\_juegos['Genero'].replace(valores\_incorrectos, 'Desconocido')

Se crea el atributo columnas con los atributos de tipo int o float y se van a remplazar los espacios vacios con '0'.

# Se crea el atributo columnas con los atributos de tipo int o float

# Se van a remplazar los espacios vacios con '0'

columnas = ['Fecha\_Lanzamiento','Max\_Jugadores','Ranking','Puntuacion\_Reseña','Ventas\_NA','Ventas\_EU',

          'Ventas\_JP','Ventas\_Otros','Ventas\_Globales','Ventas\_estimadas']

for columna in columnas:

    df\_juegos[columna] = df\_juegos[columna].fillna(0)

Se crea el atributo columnas con los atributos de tipo bool y se van a remplazar los espacios vacios con 'False'.

# Se crea el atributo columnas con los atributos de tipo bool

# Se van a remplazar los espacios vacios con 'False'

columnas = ['Multiplataforma', 'Portátil', 'Online', 'Secuela', 'Licenciado']

for columna in columnas:

    df\_juegos[columna] = df\_juegos[columna].fillna(False)

Se crea el atributo columnas con los atributos de tipo float y se van a remplazar los espacios vacios con 'Desconocido'.  
# Se crea el atributo columnas con los atributos de tipo float

# Se van a remplazar los espacios vacios con 'Desconocido'

columnas = ['Length.All PlayStyles.Average','Length.All PlayStyles.Leisure','Length.All PlayStyles.Median',

            'Length.All PlayStyles.Polled','Length.All PlayStyles.Rushed','Length.Completionists.Average',

            'Length.Completionists.Leisure','Length.Completionists.Median','Length.Completionists.Polled',

            'Length.Completionists.Rushed','Length.Main + Extras.Average','Length.Main + Extras.Leisure',

            'Length.Main + Extras.Median','Length.Main + Extras.Polled','Length.Main + Extras.Rushed',

            'Length.Main Story.Average','Length.Main Story.Leisure','Length.Main Story.Median',

            'Length.Main Story.Polled','Length.Main Story.Rushed']

for columna in columnas:

    df\_juegos[columna] = df\_juegos[columna].fillna(0)

Se eliminan las columnas no necesarias

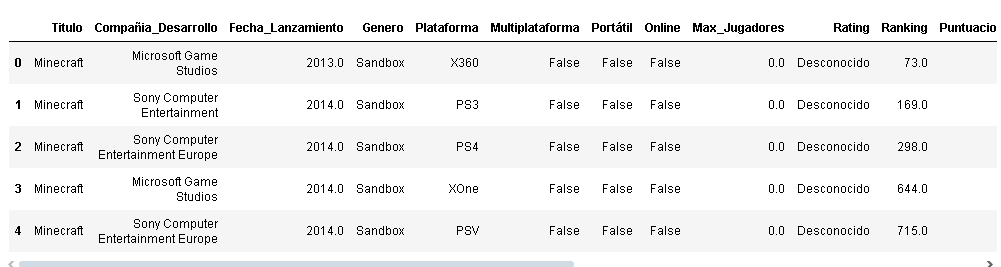
# Lista de columnas a eliminar

columnas\_a\_eliminar = ['Length.All PlayStyles.Average','Length.All PlayStyles.Leisure','Length.All PlayStyles.Median','Length.All PlayStyles.Polled','Length.All PlayStyles.Rushed','Length.Completionists.Average','Length.Completionists.Leisure','Length.Completionists.Median','Length.Completionists.Polled','Length.Completionists.Rushed','Length.Main + Extras.Average','Length.Main + Extras.Leisure','Length.Main + Extras.Median','Length.Main + Extras.Polled','Length.Main + Extras.Rushed','Length.Main Story.Average','Length.Main Story.Leisure','Length.Main Story.Median','Length.Main Story.Polled','Length.Main Story.Rushed']

# Eliminar las columnas del DataFrame

df\_juegos = df\_juegos.drop(columnas\_a\_eliminar, axis=1)

df\_juegos.head()



**4.4 Definición de la(s) Variable(s) Objetivo(s).**

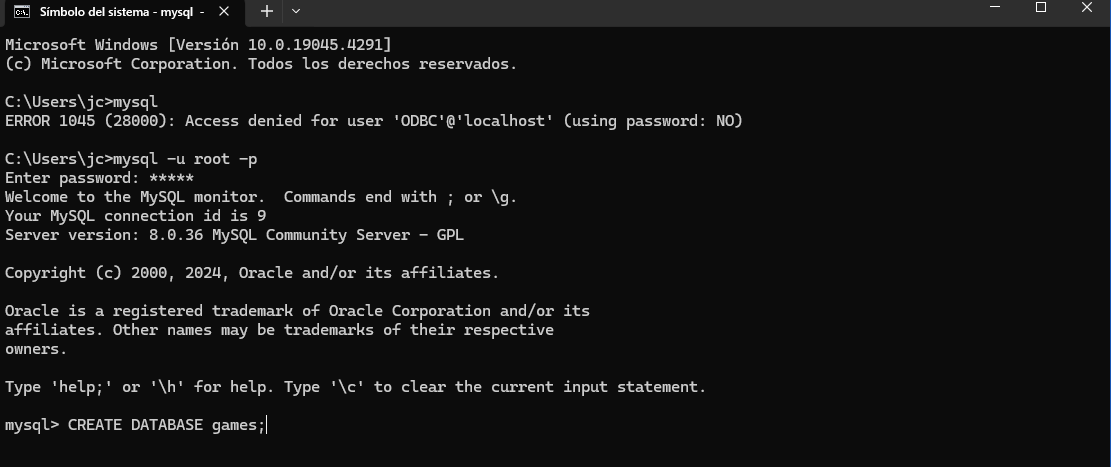
\*\*El objetivo es clasificar el 'Genero de juego' y la 'Plataforma'\*\*, ya sea de forma separada o conjunta, para obtener la información de que tipo de Genero y Plataforma se puede desarrollar el Videojuego; por lo tanto, la correlación, clasificación y predicción del Género y Plataforma de Juego estará dada \*\*por el análisis de Videojuegos Exitosos\*\*.

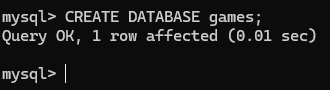
La variable objetivo será determinar qué tipo de género de juego y plataforma serán útiles para la creación de un nuevo videojuego exitoso. En este contexto, el éxito de un Videojuego este dado por la cantidad de Copias Vendidas del Videojuego que está definido en el atributo 'Ventas Globales', por lo que las Variables a utilizar para alcanzar el Objetivo del proyecto pueden incluir:

* **Género del juego**: Esta variable representa la categoría o tipo de juego al que pertenece, como acción, aventura, estrategia, deportes, etc.
* **Plataforma**: Esta variable indica el sistema o dispositivo en el que el juego será lanzado y jugado, como PlayStation, Xbox, PC, Nintendo Switch, etc.
* **Ventas Globales:** Esta variable representa las ventas totales del juego en todo el mundo. Se va a utilizar como una medida de éxito para determinar si un juego es exitoso o no.
* **Otras variables:** También se pueden considerar otras variables relevantes para el análisis, como el rating del juego, las puntuaciones de reseñas, la compañía de desarrollo, la fecha de lanzamiento, etc., dependiendo de la disponibilidad de datos y la relevancia para el objetivo específico.

**4.5 Almacenamiento de data frame en base de datos relacional**

El proceso para el almacenamiento de datos en una base de datos relacional MySQL teniendo un data frame llamado **df\_juegos** es crear una base de datos que defina las tablas, columnas, tipos de datos de las variables que se obtuvieron del proceso de Data Clean.

1. Se genera una base de datos desde la terminal de comandos:



1. Desde jupyter notebook se establece el código para convertir el data frame **df\_juegos** en una tabla en la base de datos Games. Esto se hacer a través de una conexión con el servidor de base de datos y utilizando el módulo sqlalchemy y pymysql.

!pip install pymysql

1. SQLAlchemy es una biblioteca que facilita la interacción con bases de datos relacionales y la función create\_engine se utiliza para crear un objeto de motor SQLAlchemy, que permite comunicarse con la base de datos **Games**.

from sqlalchemy import create\_engine

1. Desde la aplicación de jupyter notebook se establece una conexión a la base de datos MySQL. Se proporciona los datos de acceso (nombre de usuario y contraseña) y la ubicación del servidor MySQL (host y puerto).

# Configurar la conexión a la base de datos MySQL

usuario = 'root'

contraseña = 'mysql'

host = 'localhost'

puerto = '3306'

base\_de\_datos = 'Games'

motor = create\_engine(f'mysql+pymysql://{usuario}:{contraseña}@{host}:{puerto}/{base\_de\_datos}')

1. Se declara una **tabla llamada videojuegos** donde se almacenará las variables de la data frame **df\_juegos**.

# Nombre de la tabla

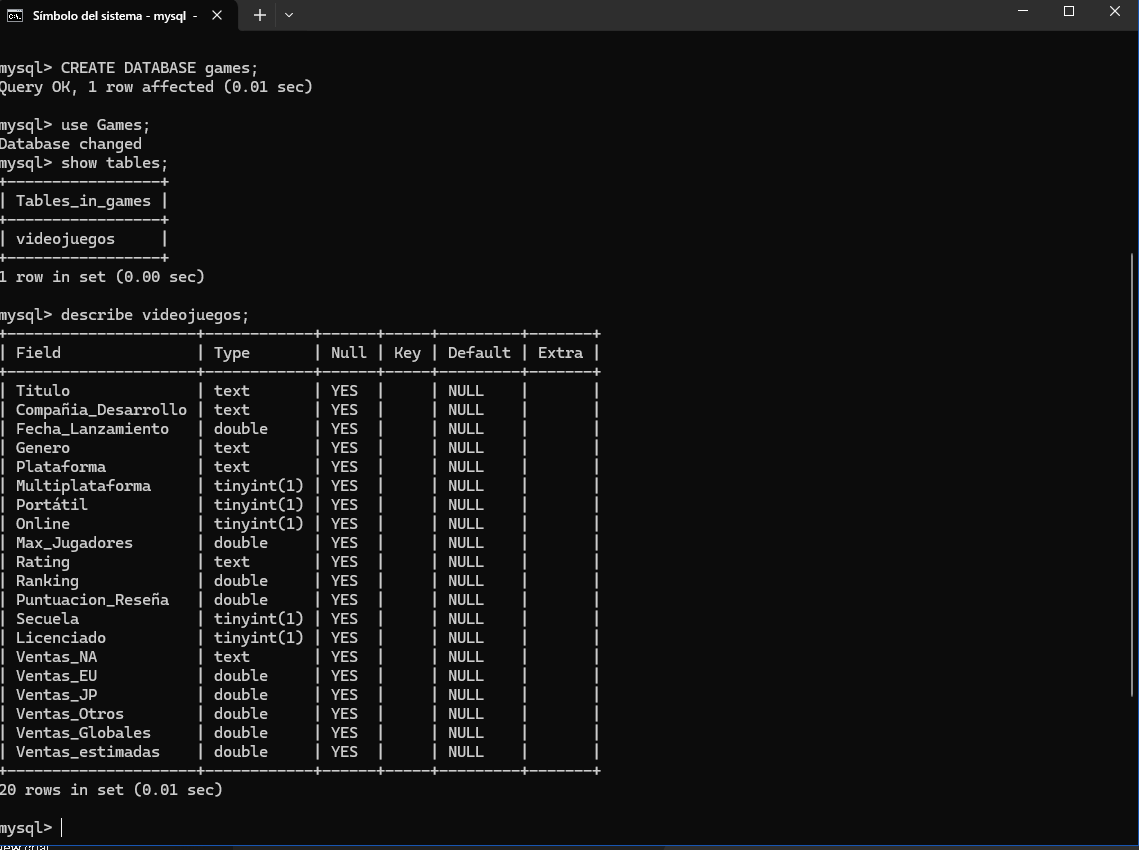
nombre\_de\_tabla = 'videojuegos'  # Cambia el nombre de la tabla según sea necesario

# Crear una tabla en MySQL a partir del DataFrame df\_juegos

df\_juegos.to\_sql(nombre\_de\_tabla, con=motor, index=False, if\_exists='replace')

print(f"DataFrame df\_juegos convertido a tabla '{nombre\_de\_tabla}' en MySQL exitosamente.")





**8. Bibliográfia**

* Documentación oficial de Python: https://docs.python.org/3/
* Documentación oficial de Pandas: https://pandas.pydata.org/docs/
* Documentación oficial de Matplotlib: <https://matplotlib.org/stable/contents.html>
* Documentación oficial de Seaborn: <https://seaborn.pydata.org>
* Documentación oficial de Scikit-learn: https://scikit-learn.org/stable/
* Documentación oficial de Numpy: https://numpy.org/doc/stable/
* Analisis Exploratorio de Datos https://www.aprendemachinelearning.com/analisis-exploratorio-de-datos-pandas-python/
* Conceptos en Python <https://www.geeksforgeeks.org>
* Dudas <https://stackoverflow.com/questions/tagged/pandas+python>
* Analisis de datos <https://ocw.uc3m.es/course/view.php?id=230>
* Diccionarios de datos en data frame <https://github.com/nsheikh23/COVID_StockMarket_Analysis/blob/master/52_Week.ipynb>
* Procesamiento de data frames en pandas https://barcelonageeks.com/eliminar-una-o-varias-columnas-de-pyspark-dataframe/
* Data Clean <https://github.com/mramshaw/Data-Cleaning>
* Ploteo de datos <https://github.com/tomimester/python-histogram/blob/master/plot-histogram-python-pandas.ipynb>
* Data Cleaning in Python: the Ultimate Guide (2020) <https://towardsdatascience.com/data-cleaning-in-python-the-ultimate-guide-2020-c63b88bf0a0d>
* Regresión lineal en Python: un análisis detallado <https://www.cienciadedatos.net/documentos/py10-regresion-lineal-python.html>
* Aplicando Machine Learning para predecir éxitos de videojuegos   
  <https://www.saturdays.ai/projects/Videojuegos.html>
* Machine Learning para la predicción de ventas https://www.aprendemachinelearning.com/regresion-lineal-en-espanol-con-python/
* McKinney, Wes. Python for Data Analysis. O'Reilly Media, Inc, 2017.
* VanderPlas, Jake. Python Data Science Handbook: Essential Tools for Working with Data. O'Reilly Media, Inc, 2016.