ANÁLISIS DE RESEÑAS DE VIDEOJUEGOS EN AMAZON CON MAPREDUCE EN HADOOP

[Máster Universitario en Computación en la Nube y de Altas Prestaciones]

[Aguirre Ramírez Diego Alejando] [Liu Cai]

Contenidos

1. Introducción.

Este trabajo consiste en implementar un código de análisis de datos utilizando el modelo de programación MapReduce sobre Apache Hadoop, para procesar y analizar un conjunto de datos de reseñas de videojuegos vendidos en Amazon. Se utilizará el framework de Hadoop Streaming y el lenguaje de programación Python para la implementación, y se realizará un análisis detallado de los datos utilizando las herramientas disponibles en el modelo MapReduce. El objetivo es extraer información relevante de las reseñas de los usuarios y determinar patrones y tendencias en las opiniones de los clientes sobre los videojuegos vendidos en Amazon, siendo los datos extraídos de: http://snap.stanford.edu/data/web-Amazon-links.html.

2. Entorno Hadoop con Python.

Hadoop es un framework de software de código abierto utilizado para el almacenamiento y procesamiento distribuido de grandes conjuntos de datos en clústeres de servidores. El entorno Hadoop consta de varios componentes clave, como el sistema de archivos distribuido Hadoop (HDFS), el motor de procesamiento de datos MapReduce, el gestor de recursos YARN y otros servicios de apoyo.

En cuanto a la elección del lenguaje de programación, Python es una opción popular para el procesamiento de datos en Hadoop, debido a su facilidad de uso, legibilidad y flexibilidad. Además, Python es conocido por su amplia variedad de librerías de ciencia de datos y aprendizaje automático, lo que lo convierte en una herramienta valiosa para el análisis de datos. Java es el lenguaje de programación predeterminado para Hadoop y ofrece un rendimiento superior para algunas tareas, pero su curva de aprendizaje es más empinada que la de Python y el código Java puede ser más verboso y difícil de leer. En resumen, la elección de Python o Java dependerá de las necesidades y preferencias específicas del proyecto. Siendo este caso Python por su facilidad de lectura y depuración, además de contar con múltiples librerías para análisis de datos que podrían ser interesantes para el desarrollo de este trabajo en caso de ser necesitadas.

3. Implementación de las funciones mapper y reducer.

La función map toma como entrada un conjunto de datos y los transforma en una serie de pares clave-valor, por ejemplo, en este caso podrían llegar a usarse:

- product/title
- product/price
- review/userId
- review/profileName
- review/helpfulness
- review/score
- review/time
- review/summary
- review/text

Cada par clave-valor representa un fragmento de los datos de entrada y es procesado por una tarea Map en el clúster. El objetivo de la función map es procesar y filtrar los datos de entrada para prepararlos para la fase de reduce.

La función reduce toma como entrada los pares clave-valor producidos por la fase de map y los combina en un conjunto menor de pares clave-valor, que representan los resultados finales del procesamiento de datos. El objetivo de la función reduce es realizar un resumen o una agregación de los datos de entrada. La función reduce también puede ser escrita por el programador y personalizada para satisfacer las necesidades específicas del proyecto.

4. Datos a analizar.

Para el análisis de datos en este proyecto, se utilizará un conjunto de datos de reseñas de videojuegos vendidos en Amazon. Este conjunto de datos contiene información sobre los videojuegos, como su título, precio y número de ventas, así como información sobre los usuarios que han dejado reseñas, como su identificación de usuario y perfil. Además, el conjunto de datos contiene detalles sobre las reseñas en sí, como la puntuación asignada por el usuario, la fecha en que se publicó la reseña y el texto de la reseña.

El objetivo del análisis de datos es extraer información relevante sobre los videojuegos y los usuarios, como las tendencias en las opiniones de los clientes, los patrones de compra (que tipos de juegos se compran más) y los aspectos más destacados de los videojuegos que los usuarios aprecian (tipos de juego, música, gráficos, etc). Se utilizará el modelo de programación MapReduce en Apache Hadoop y el lenguaje de programación Python para procesar y analizar el conjunto de datos de reseñas y extraer información útil y valiosa sobre los videojuegos y los usuarios.

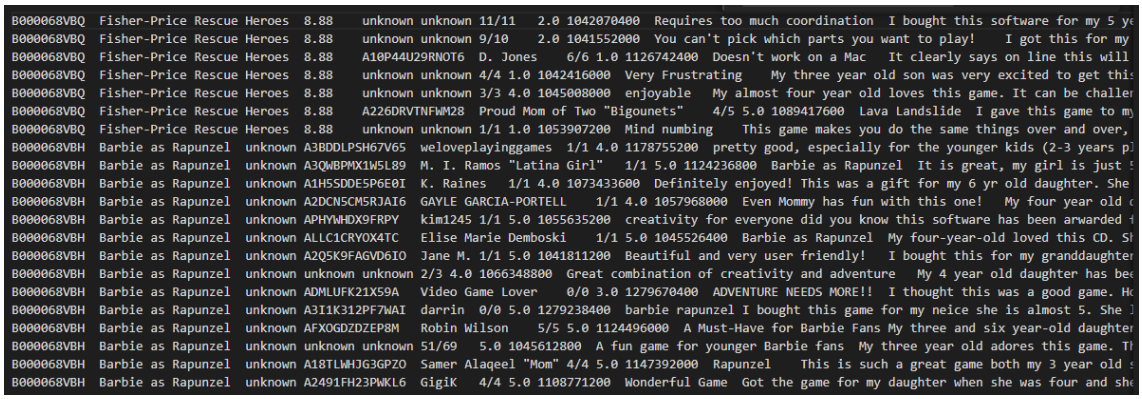
4.1 Preparación de los datos.

Antes de que los datos puedan ser analizados, es importante realizar una preparación previa, que incluye la limpieza de los datos y la eliminación de datos duplicados.

En primer lugar, se debe realizar una exploración inicial de los datos para detectar posibles problemas, como valores faltantes o incorrectos, campos duplicados o errores de formato. A continuación, se deben tomar medidas para corregir estos problemas, por ejemplo, completando los valores faltantes o eliminando los registros con errores.

Una vez que los datos estén limpios, el siguiente paso es eliminar los datos duplicados. Los registros duplicados pueden ser causados por errores en la entrada de datos o por la inclusión de múltiples registros idénticos en el conjunto de datos. La eliminación de los registros duplicados ayuda a garantizar que el análisis se base en datos precisos y confiables.

Así pues, hemos implementado un programa de Python convertir_formato.py para convertir los datos en el siguiente formato:

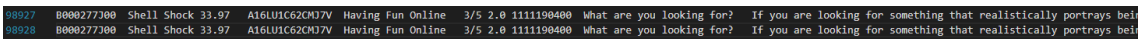


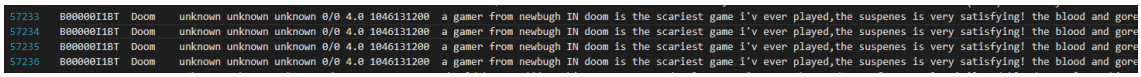
```
B000068VBQ  Fisher-Price Rescue Heroes  8.88    unknown unknown 11/11   2.0 1042070400  Requires too much coordination  I bought this software for my 5 y
B000068VBQ  Fisher-Price Rescue Heroes  8.88    unknown unknown 9/10    2.0 1041552000  You can't pick which parts you want to play!    I got this for my
B000068VBQ  Fisher-Price Rescue Heroes  8.88    A10P44U29RNOT6  D. Jones    6/6 1.0 1126742400  Doesn't work on a Mac   It clearly says on line this will
B000068VBQ  Fisher-Price Rescue Heroes  8.88    unknown unknown 4/4 1.0 1042416000  Very Frustrating    My three year old son was very excited to get thi
B000068VBQ  Fisher-Price Rescue Heroes  8.88    unknown unknown 3/3 4.0 1045008000  enjoyable   My almost four year old loves this game. It can be challe
B000068VBQ  Fisher-Price Rescue Heroes  8.88    A226DRVTNFWM28  Proud Mom of Two "Bigounets"    4/5 5.0 1089417600  Lava Landslide  I gave this game to m
B000068VBQ  Fisher-Price Rescue Heroes  8.88    unknown unknown 1/1 1.0 1053907200  Mind numbing    This game makes you do the same things over and over,
B000068VBH  Barbie as Rapunzel  unknown A3BDDLPSH67V65  weloveplayinggames  1/1 4.0 1178755200  pretty good, especially for the younger kids (2-3 years p
B000068VBH  Barbie as Rapunzel  unknown A3QWBPMX1W5L89  M. I. Ramos "Latina Girl"   1/1 5.0 1124236800  Barbie as Rapunzel  It is great, my girl is just
B000068VBH  Barbie as Rapunzel  unknown A1H5SDDE5P6E0I  K. Raines   1/1 4.0 1073433600  Definitely enjoyed! This was a gift for my 6 yr old daughter. She
B000068VBH  Barbie as Rapunzel  unknown A2DCN5CM5RJAI6  GAYLE GARCIA-PORTELL    1/1 4.0 1057968000  Even Mommy has fun with this one!   My four year old
B000068VBH  Barbie as Rapunzel  unknown APHYWHDX9FRPY   kim1245 1/1 5.0 1055635200  creativity for everyone did you know this software has been arwarded
B000068VBH  Barbie as Rapunzel  unknown ALLC1CRYOX4TC   Elise Marie Demboski    1/1 5.0 1045526400  Barbie as Rapunzel  My four-year-old loved this CD. S
B000068VBH  Barbie as Rapunzel  unknown A2Q5K9FAGVD6IO  Jane M. 1/1 5.0 1041811200  Beautiful and very user friendly!   I bought this for my granddaughte
B000068VBH  Barbie as Rapunzel  unknown unknown unknown 2/3 4.0 1066348800  Great combination of creativity and adventure   My 4 year old daughter has be
B000068VBH  Barbie as Rapunzel  unknown ADMLUFK21X59A   Video Game Lover    0/0 3.0 1279670400  ADVENTURE NEEDS MORE!!  I thought this was a good game. H
B000068VBH  Barbie as Rapunzel  unknown A3I1K312PF7WAI  darrin  0/0 5.0 1279238400  barbie rapunzel I bought this game for my neice she is almost 5. She
B000068VBH  Barbie as Rapunzel  unknown AFXOGDZDZEP8M   Robin Wilson    5/5 5.0 1124496000  A Must-Have for Barbie Fans My three and six year-old daughte
B000068VBH  Barbie as Rapunzel  unknown unknown unknown 51/69   5.0 1045612800  A fun game for younger Barbie fans  My three year old adores this game. T
B000068VBH  Barbie as Rapunzel  unknown A18TLWHJG3GPZO  Samer Alaqeel "Mom" 4/4 5.0 1147392000  Rapunzel    This is such a great game both my 3 year old
B000068VBH  Barbie as Rapunzel  unknown A2491FH23PWKL6  GigiK   4/4 5.0 1108771200  Wonderful Game  Got the game for my daughter when she was four and sh
```

```python
input_file = "Video_Games.txt"
output_file = "Video_Games_formatted.txt"

with open(input_file, "r") as input, open(output_file, "w") as output:
    for line in input:
        line = line.strip()
        if line.startswith("product/") or line.startswith("review/"):
            fields = line.split(": ")
            key = fields[0]
            if len(fields) > 1:
                value = fields[1]
            else:
                value = "Empty"  # Si el valor está vacío
            output.write(value + "\t")  # Tabulador
        else:
            output.write("\n")  # Agrega una nueva línea después de cada reseña

print("Conversión de formato completada. Resultado guardado en", output_file)
```

Cada línea representa una reseña, cada columna está separada por un tabulador y representa por este orden: productid, title, Price, userId, profileName, helpfulness, score, time, summary, text. Además, hemos sustituido los campos vacíos por ¨Empty¨. Para evitar que falten columnas.

Por otro lado, para eliminar la duplicación de datos como:





Hemos implementado un programa eliminar_lineas_dup.py para eliminar las reseñas duplicadas.

```python
input_file = "Video_Games_formatted.txt"
output_file = "Video_Games_reviews.txt"

lineas_vistas = set()
lineas_duplicadas = []

i = 0

with open(input_file, "r") as input, open(output_file, "w") as output:
    for line in input:
        i = i + 1
        if line not in lineas_vistas:
            lineas_vistas.add(line)
            output.write(line)
        else:
            lineas_duplicadas.append(i)

print("Líneas duplicadas eliminadas. Resultado guardado en", output_file)
print("Líneas duplicadas:", lineas_duplicadas)
```

```
PS E:\UPV\CBD\trabajo\programa> e:; cd 'e:\UPV\CBD\trabajo\programa'; & 'C:\Users\Viooola\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\Viooola
\.vscode\extensions\ms-python.python-2023.8.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '54650' '--' 'E:\UPV\CBD\trabajo\programa\convertir
_formato.py'
Conversión de formato completada. Resultado guardado en Video_Games_formatted.txt
PS E:\UPV\CBD\trabajo\programa> e:; cd 'e:\UPV\CBD\trabajo\programa'; & 'C:\Users\Viooola\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\Viooola
\.vscode\extensions\ms-python.python-2023.8.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '54659' '--' 'E:\UPV\CBD\trabajo\programa\eliminar_
lineas_dup.py'
Líneas duplicadas eliminadas. Resultado guardado en Video_Games_reviews.txt
Líneas duplicadas: [109, 736, 2261, 2627, 3076, 3622, 3805, 3878, 4574, 4681, 5047, 5356, 5869, 6056, 6365, 6366, 6792, 7391, 7692, 9191, 9423, 9497, 10509, 125
00, 13238, 13674, 13700, 14091, 14734, 15106, 15534, 15597, 15598, 16901, 18733, 20674, 20931, 21662, 23453, 24048, 24085, 24502, 24830, 25583, 25827, 25985, 26
465, 26628, 26705, 27949, 28908, 31289, 32651, 32796, 33291, 33409, 33501, 35164, 36472, 36920, 38253, 38383, 41510, 42617, 42825, 43056, 43188, 45116, 45855, 4
8111, 49073, 49233, 49304, 50649, 54260, 54692, 54693, 54811, 54850, 55535, 55537, 55542, 55583, 55737, 56236, 56501, 57234, 57235, 57236, 57352, 57930, 58014,
58149, 58961, 58964, 59089, 59212, 59306, 59731, 59733, 59734, 59737, 59970, 59971, 59972, 60351, 60362, 61811, 61878, 62840, 62972, 63158, 63189, 64255, 64256,
 64322, 64533, 64620, 65107, 66935, 67783, 71805, 71806, 71833, 71950, 71964, 73274, 73856, 74152, 74263, 74264, 74303, 77910, 78123, 78259, 78515, 78527, 79588
```

Finalmente, tras ejecutar ambos programas, obtendremos los datos limpios en Video_Games_reviews.txt. Copiamos este fichero desde el sistema de archivos local a HDFS con el comando:

hadoop fs -put /trabajo/dataset/Video_Games_reviews.txt trabajo/input

y para mostrar las primeras 5 líneas:

hadoop fs -text trabajo/input/Video_Games_reviews.txt | head -n 5

4.2.1 Datos sobre los videojuegos de las reseñas.

Para el análisis de los videojuegos, se buscará (de ser posible) extraer los siguientes datos:

- Número de reseñas: Este dato se utilizará para determinar la popularidad de cada videojuego en el conjunto de datos de reseñas.
- Importe total: Este dato se calculará multiplicando el precio del videojuego por el número de reseñas que tiene. Se utilizará para analizar el valor económico de cada videojuego y determinar cuáles son los más rentables.
- Valoración de los videojuegos: Este dato se extraerá de la columna "review/score" y se analizará para determinar cuáles son los videojuegos con las calificaciones más altas y bajas.
- Características: Se extraerá información de las columnas "review/summary" y "review/text" para analizar las características que los usuarios mencionan con mayor frecuencia en las reseñas de los videojuegos. Se analizarán las siguientes características: gráficos, historia, duración, música, personalización, configuración, contenido descargable (DLC), estilo de dibujos animados, y errores técnicos (errores, bugs, glitches).
- Género de los videojuegos: Se utilizarán palabras clave como "multiplayer", "co-op", "survival", "online", "RPG", "rogue-like", "sport", "shooter", "simulation", "strategy", "adventure", "action", "role-playing", "MMORPGs", "FPS", "PvP" y "PvE" para identificar el género de cada videojuego.
- Precio: Se utilizará la columna "product/price" para analizar la relación entre el precio de los videojuegos y su popularidad o calificación.

Ahora, pasaremos a explicar como analizaremos los datos de los videojuegos, que funciones mapper y reducer escribimos y una explicación breve de cómo funcionan.

4.2.2 Número de reseñas.

El primer apartado que se explicará cómo funcionan las funciones mapper y reducer para el "Número de reseñas".

MAPPER:

En el mapper, cada línea de entrada representa un registro de reseña de un videojuego. El código se encarga de dividir la línea en sus campos individuales, utilizando el separador de tabulación ("\t") para separar los valores. A continuación, se verifica que la línea contenga los 10 campos esperados.

Si los campos son válidos, se extraen el "productid" y el "title" del videojuego. Estos campos se utilizan como clave para el mapeo, mientras que el valor se establece en 1, ya que se quiere contar cada reseña individualmente.

Finalmente, se imprime la salida del mapper en el formato "clave \t valor" utilizando el método print().

```python
#!/usr/bin/python

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        (
            productid,
            title,
            Price,
            userId,
            profileName,
            helpfulness,
            score,
            time,
            summary,
            text,
        ) = data

        key = [productid, title]
        print("{0}\t{1}".format(key, 1))
```

REDUCER:

En el reducer, los pares clave-valor de salida del mapper son recibidos en orden. El código verifica si la clave actual es diferente a la clave anterior (thisKey). Si se detecta un cambio de clave, significa que se ha pasado a un nuevo producto y se imprime la clave anterior junto con el recuento acumulado (count).

Después de imprimir la salida, se restablece el contador a cero y se actualiza la clave anterior con la clave actual.

```python
#!/usr/bin/python

import sys

oldKey = None
count = 0

for line in sys.stdin:
    thisKey = line.strip().split("\t")[0]

    if oldKey and oldKey != thisKey:  # nuevo producto
        print("{0}\t{1}".format(oldKey, count))
        count = 0

    oldKey = thisKey
    count = count + 1

if oldKey:
    print("{0}\t{1}".format(oldKey, count))
```

El resultado final será una lista de productos junto con el número total de reseñas que han recibido.

```
alucloud71@hadoopmaster:~/trabajo$ hadoop fs -text trabajo/output/num_review/part-0000
0 | head -n 20
['043940133X', 'Star Wars Math']        7
['0439715571', 'Mortal Kombat 4']       2
['0761547096', 'Star Wars Battlefront'] 7
['1549669109', 'Super Mario RPG']       1
['1591500613', 'Streetfinder (Rand McNally Streetfinder)']      1
['1886846774', 'Think Like a King Chess Workout Family Package']        5
['1886846820', 'The Many Faces of Go']  3
['9752300480', 'Dance Dance Revolution DDR TV Pad (No Console)']        73
['9753086024', '2 x DDR Multi-Platform Super Sensors Energy Super Deluxe Dance Pad (PS
, PS2, XBox, PC, Mac) with DDR Game Ultramix 2 (XBOX)'] 2
['9753425201', 'Sacred Underworld Expansion Pack']      2
['9754585288', 'Dance Dance Revolution Multi-Platform Super Sensors Energy Deluxe Danc
e Pad (PS/PS2/Xbox/PC/Mac) with Konami DDR PC Game']    7
['9755334602', 'Dance Dance Revolution Mario Mix Original Nintendo Dance Pad (Without
Game)'] 6
['975539463X', 'GRAND THEFT AUTO SAN ANDREAS']  11
['9756083077', 'Dance Dance Revolution Pad (PC) with bundle Konami Dance Dance Revolut
ion CD [Win]']  5
['9756663855', '2 Players Dance Dance Revolution Blue Twin DDR TV Pad'] 23
['9758496786', 'DDR Multi-Platform Super Sensors Energy Super Deluxe Dance Pad (PS, PS
2, XBox, PC, Mac) with DDR Game Ultramix 2 (XBOX)']     1
['975912307X', 'Dance Dance Revolution DDR Super Deluxe Pad Version 2.0 with DDR Extre
me 2']  8
['9759960087', 'Zenith LCD Display bundle with BMX XXX Video Game (GameCube)']  2
['B000003SQQ', 'Clay Fighter 63 1/3']   17
['B000006IX3', 'The Tick']      3
```

4.2.3 Importe total.

El siguiente apartado es cómo funcionan las funciones mapper y reducer para el "Importe total".

MAPPER:

En el mapper, cada línea de entrada representa un registro de reseña de un videojuego. Al igual que en el caso anterior, el código divide la línea en sus campos individuales utilizando el separador de tabulación ("\t"). Se verifica que la línea contenga los 10 campos esperados.

Si los campos son válidos, se extraen el "productid", el "title" y el "Price" del videojuego. Estos campos se utilizan como clave para el mapeo, y se verifica que el precio no sea "unknown" o "empty" para asegurarse de que sea un valor válido.

Luego, se imprime la salida del mapper en el formato "clave \t valor" donde la clave es la combinación de "productid" y "title" y el valor es el precio del videojuego.

```python
#!/usr/bin/python

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        (
            productid,
            title,
            Price,
            userId,
            profileName,
            helpfulness,
            score,
            time,
            summary,
            text,
        ) = data

        key = [productid, title]

        if Price != "unknown" and Price != "empty":
            print("{0}\t{1}".format(key, Price))
```

REDUCER:

En el reducer, los pares clave-valor de salida del mapper son recibidos en orden. El código se encarga de acumular el importe total para cada clave de videojuego.

Se inicializa una variable "salesTotal" en 0 y una variable "oldKey" para realizar un seguimiento de la clave anterior. Luego, se recorre cada línea de entrada. Si la longitud de la línea dividida no es igual a 2, se omite la línea.

Se compara la clave actual (thisKey) con la clave anterior (oldKey). Si hay un cambio de clave, se imprime la clave anterior junto con el importe total acumulado (salesTotal), redondeado a 2 decimales.

Luego, se actualiza la clave anterior y se agrega el valor actual (thisSale) al importe total (salesTotal).

Finalmente, se imprime la última clave y el importe total acumulado si existen registros pendientes.

```python
#!/usr/bin/python

import sys

salesTotal = 0
oldKey = None

for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisSale = data_mapped

    if oldKey and oldKey != thisKey:
        print("%s\t%s" % (oldKey, round(salesTotal, 2)))
        salesTotal = 0

    oldKey = thisKey
    salesTotal += float(thisSale)

if oldKey != None:
    print("%s\t%s" % (oldKey, round(salesTotal, 2)))
```

El resultado final será una lista de productos junto con su importe total calculado.

```
alucloud71@hadoopmaster:~$ hadoop fs -text trabajo/output/id_price/part-0000
0 | head -n 20
['1886846774', 'Think Like a King Chess Workout Family Package']        309.
75
['B000006OTB', 'Rascal']        169.32
['B000006OTC', 'Newman Haas Racing']    95.96
['B000006OVD', 'Buster Bros. Collection']       84.68
['B000006OWS', 'Riven'] 1079.85
['B000006OWT', 'Forsaken']      479.88
['B000006P0K', 'Tekken 2']      2149.57
['B000006RGO', 'Deathtrap Dungeon']     329.89
['B000006RGP', 'Ninja'] 399.36
['B000006RGQ', 'Fighting Force']        492.1
['B000006RGS', 'Tomb Raider']   3293.4
['B000007NJA', 'Forsaken ']     221.94
['B000007NJB', 'Jeremy Mcgrath Supercross 98']  59.67
['B000007NJC', 'WWF Warzone']   392.4
['B000007NJD', 'Bust A Move 2'] 608.8
['B000007NJE', 'All-Star Baseball 99']  549.89
['B000009QCY', 'Trap Gunner']   249.8
['B00000DMA8', 'Command & Conquer']     637.07
['B00000DMAB', 'Microsoft Flight Simulator 98/World of Flight 98']      169.
52
['B00000DMAC', 'Grand Theft Auto']      169.83
```

4.2.4 Valoración de los videojuegos

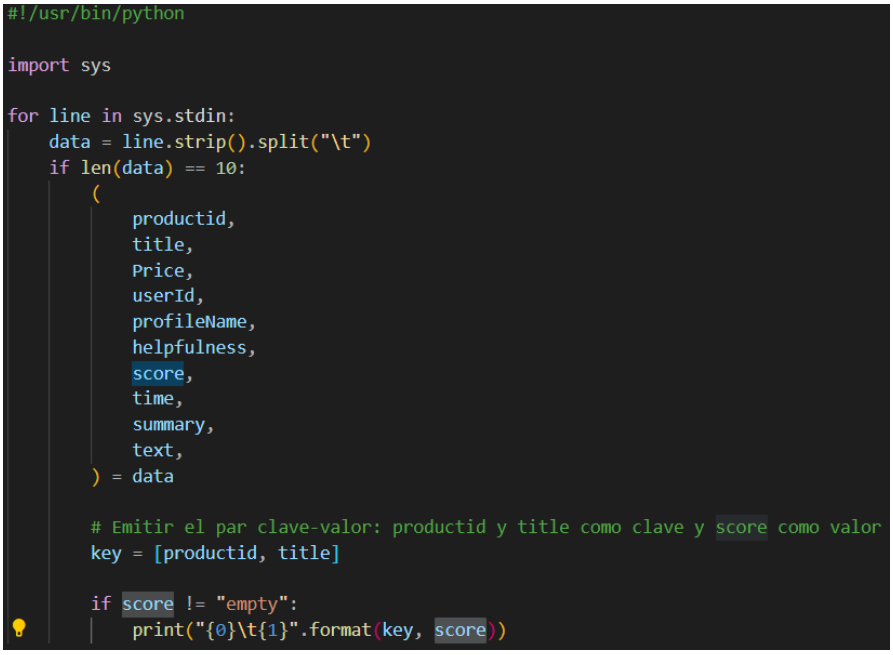
El siguiente apartado es cómo funcionan las funciones mapper y reducer para el "Valoración de videojuegos".

MAPPER:

En el mapper, se procesa cada línea de entrada que representa un registro de reseña de un videojuego. Se divide la línea en sus campos individuales utilizando el separador de tabulación ("\t"). Se verifica que la línea contenga los 10 campos esperados.

Si los campos son válidos, se extraen el "productid", el "title" y el "score" del videojuego. Estos campos se utilizan como clave para el mapeo. Además, se verifica que el campo "score" no sea "empty".

Luego, se imprime la salida del mapper en el formato "clave \t valor" donde la clave es la combinación de "productid" y "title" del videojuego, y el valor es la puntuación ("score").



```python
#!/usr/bin/python

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        (
            productid,
            title,
            Price,
            userId,
            profileName,
            helpfulness,
            score,
            time,
            summary,
            text,
        ) = data

        # Emitir el par clave-valor: productid y title como clave y score como valor
        key = [productid, title]

        if score != "empty":
            print("{0}\t{1}".format(key, score))
```

REDUCER:

En el reducer, se reciben los pares clave-valor de salida del mapper en orden. El código acumula la suma de las puntuaciones y cuenta el número de puntuaciones para cada videojuego.

Se inicializan variables, como "current_productid" para realizar un seguimiento del producto actual, "score_sum" para acumular la suma de las puntuaciones y "count" para contar el número de puntuaciones.

Se recorren las líneas de entrada y se divide el producto y la puntuación. Si hay un cambio en el producto actual, se calcula el promedio de puntuación dividiendo la suma de las puntuaciones por el recuento de puntuaciones y se emite el resultado para el producto anterior.

Luego, se actualizan las variables con el nuevo producto y se acumula la puntuación y se incrementa el contador.

Finalmente, se calcula el promedio de puntuación para el último producto y se emite el resultado, teniéndose solo en cuenta los datos que tengan 10 reseñas o más

```python
#!/usr/bin/python

import sys

current_productid = None
score_sum = 0
count = 0

for line in sys.stdin:
    productid, score = line.strip().split("\t")

    if current_productid and current_productid != productid:
        # Calcular el score promedio y emitir el resultado
        if count >= 10:
            average_score = score_sum / count
            print("{0}\t{1}".format(current_productid, round(average_score, 2)))

        score_sum = 0
        count = 0

    # Actualizar el producto actual y acumular el score y cuenta el numero
    current_productid = productid
    score_sum += float(score)
    count = count + 1

# Calcular el score promedio para el último producto
if current_productid:
    if count >= 10:
        average_score = score_sum / count
        print("{0}\t{1}".format(current_productid, round(average_score, 2)))
```

El resultado final será una lista de productos junto con su puntuación promedio redondeada a 2 decimales.

```
alucloud71@hadoopmaster:~/trabajo$ hadoop fs -text trabajo/output/score/part-00000 | h
ead -n 20
['9752300480', 'Dance Dance Revolution DDR TV Pad (No Console)']        2.23
['975539463X', 'GRAND THEFT AUTO SAN ANDREAS']  3.73
['9756663855', '2 Players Dance Dance Revolution Blue Twin DDR TV Pad'] 2.91
['B000003SQQ', 'Clay Fighter 63 1/3']   3.88
['B000006OTB', 'Rascal']        2.0
['B000006OVE', 'Breath of Fire III']    4.38
['B000006OVF', 'Marvel Super Heroes']   4.5
['B000006OVI', 'Street Fighter Collection']     3.6
['B000006OVJ', 'Mega Man Legends']      4.52
['B000006OVK', 'X-Men vs. Street Fighter']      3.8
['B000006OVL', 'Street Fighter EX Plus Alpha']  4.6
['B000006OWS', 'Riven'] 4.33
['B000006OWT', 'Forsaken']      3.0
['B000006P0J', 'Tekken']        4.11
['B000006P0K', 'Tekken 2']      4.72
['B000006P0M', 'Ace Combat 2']  4.53
['B000006P0N', 'Time Crisis plus Guncon']       4.89
['B000006P0P', 'Klonoa']        4.95
['B000006RGO', 'Deathtrap Dungeon']     2.91
['B000006RGQ', 'Fighting Force']        3.74
```

4.2.5 Características

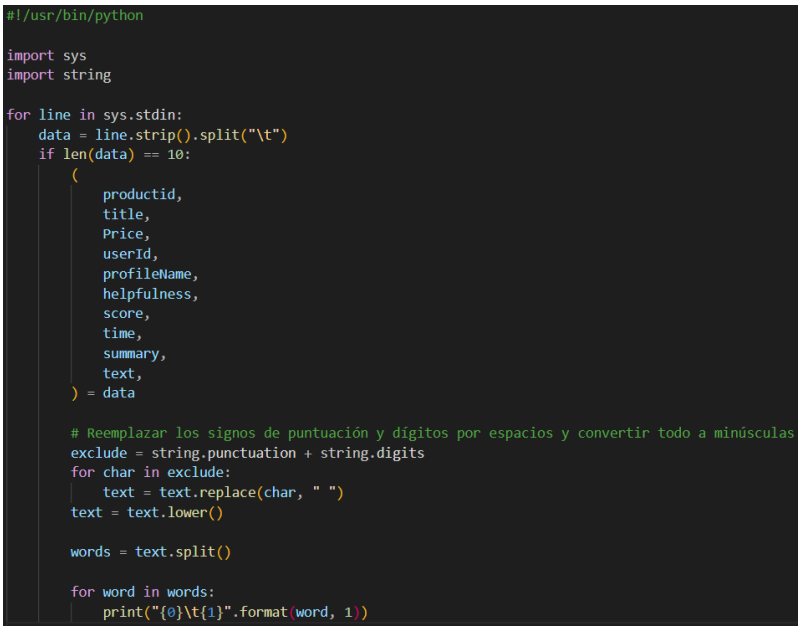
MAPPER:

En el mapper, se procesa cada línea de entrada que representa un registro de reseña de un videojuego. Se divide la línea en sus campos individuales utilizando el separador de tabulación ("\t"). Se verifica que la línea contenga los 10 campos esperados.

Si los campos son válidos, se extraen el "productid", el "title" y el "text" del videojuego. A continuación, se realiza un procesamiento del texto para extraer las características mencionadas en el resumen y el texto de la reseña.

Se eliminan los signos de puntuación y se convierte todo a minúsculas para normalizar el texto. Luego, se dividen las palabras y se emite cada palabra con un valor de 1 para contar su aparición.

Finalmente, se imprime la salida del mapper en el formato "palabra \t valor" donde la palabra es una característica mencionada en el resumen o texto de la reseña y el valor es siempre 1.



```python
#!/usr/bin/python

import sys
import string

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        (
            productid,
            title,
            Price,
            userId,
            profileName,
            helpfulness,
            score,
            time,
            summary,
            text,
        ) = data

        # Reemplazar los signos de puntuación y dígitos por espacios y convertir todo a minúsculas
        exclude = string.punctuation + string.digits
        for char in exclude:
            text = text.replace(char, " ")
        text = text.lower()

        words = text.split()

        for word in words:
            print("{0}\t{1}".format(word, 1))
```

REDUCER:

En el reducer, se reciben los pares clave-valor de salida del mapper en orden. El código acumula el recuento de apariciones de cada palabra característica mencionada en las reseñas de los videojuegos.

Se utiliza un diccionario, "wordCounts", para almacenar el recuento de apariciones de cada palabra.Se recorren las líneas de entrada y se divide la palabra y el recuento. Si la palabra ya está en el diccionario, se incrementa su recuento. Si no está, se agrega al diccionario con un recuento inicial de 1.

```python
#!/usr/bin/python

import sys

wordCounts = {}

for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisWord, count = data_mapped

    if thisWord in wordCounts:
        wordCounts[thisWord] += int(count)
    else:
        wordCounts[thisWord] = int(count)

# Output the word counts for words with more than 4 occurrences
for word, count in wordCounts.items():
    if count > 4:
        print("{0}\t{1}".format(word, count))
```

Finalmente, se imprime la salida del reducir solo para las palabras que tienen más de 4 apariciones en las reseñas. Esto se hace para filtrar palabras poco frecuentes o ruido. (Cabe añadir que este valor podría cambiar, pero, se decidió que 4 es un buen número para acotar los datos).

El resultado final será una lista de palabras características mencionadas en las reseñas de videojuegos, junto con su recuento total. Además, estos resultados se pueden usar para sacar otros datos como por ejemplo palabras clave.

```
alucloud71@hadoopmaster:~/trabajo$ hadoop fs -text trabajo/output/wordcount/part-00000
 | head -n 20
a       1334125
aa      1111
aaa     332
aaaa    20
aaaaa   6
aaaaaa  9
aaaah   8
aaagh   5
aaah    10
aaas    9
aabout  7
aac     7
aachen  6
aacute  261
aah     18
aan     5
aand    30
aang    8
aaragorn        6
aare    7
```

4.3.1 Datos sobre los usuarios de las reseñas.

Para el análisis de los usuarios, se buscará extraer los siguientes datos:

- Comportamiento de los usuarios: Se analizará la columna "review/userId" para determinar cuáles son los usuarios más activos en términos de dejar reseñas. Este dato será útil para identificar a los usuarios influyentes y comprender mejor sus preferencias de videojuegos.

4.3.2 Usuarios más activos.

MAPPER:

En el mapper, se procesa cada línea de entrada que representa un registro de reseña de un videojuego. Se divide la línea en sus campos individuales utilizando el separador de tabulación ("\t"). Se verifica que la línea contenga los 10 campos esperados.

Si los campos son válidos, se extraen el "userId" y el "profileName" del usuario que ha dejado la reseña. Estos campos se utilizan como clave para el mapeo, y el valor se establece en 1 para indicar la ocurrencia de un usuario activo.

Luego, se imprime la salida del mapper en el formato "clave \t valor" donde la clave es la combinación de "userId" y "profileName" del usuario, y el valor es siempre 1.

El resultado final será una lista de usuarios junto con el recuento de sus reseñas, lo que permite identificar a los usuarios más activos en el conjunto de datos.

```python
#!/usr/bin/python

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        (
            productid,
            title,
            Price,
            userId,
            profileName,
            helpfulness,
            score,
            time,
            summary,
            text,
        ) = data

        key = [userId, profileName]
        print("{0}\t{1}".format(key, 1))
```

Para el reducer podemos utilizar la misma función de **reducer_num_id.py**.

El resultado final será una lista de los usuarios más activos con la cantidad de veces que han realizado una reseña.

```
alucloud71@hadoopmaster:~/trabajo$ hadoop fs -text trabajo/output/num_review_user/part
-00000 | head -n 20
['A00089042GMZ1I1K4M9KZ', 'Isayah C']   1
['A002557013OCU5T8231WO', 'Liam']       1
['A006458827ALF2J23JJTO', 'Matthew Fudge']      3
['A0065547FPGGG31RKMMP', 'Siegmund Early Learnining Center']    1
['A007657513WTSD9WVVH6X', 'vincent petersen']   1
['A0086945VK8HVOJ6RRB6', 'Ryan Latimer']        2
['A01050672L68UYWSNJKPO', 'jesse']      1
['A0138820SJHLLHWDUR6I', 'Ryan Wingo']  1
['A01689603CJXMISQVROYV', 'Ruben Vazquez']      1
['A01803881H1UK1GI0QQ98', 'Connor Lark']        1
['A01882661RPBZMY5R180K', 'Juan Alfredo']       1
['A01891683KZVKDIZT6BS6', 'Sunniva Johnsen']    1
['A0198067UBJPDLCUWNFG', 'abraham zuniga']      1
['A023023334QO4HEJJIFCP', 'sherry']     1
['A0251100106LM0BAJ9E0V', 'Sonnelion']  1
['A026093527869JOYGRDX3', 'Sam']        1
['A02643303F51BUOR3J400', 'squirrel sniper']    1
['A0266076X6KPZ6CCHGVS', 'mslyndean']   1
['A02726203GDVLE8TEEOH1', 'Justin Reed']        1
['A0326148384CSDVOOAIE2', 'Ann Collins']        1
```

5.1 Análisis de los resultados obtenidos a partir de los datos: Sorteo de datos y palabras clave.

Antes de comenzar el análisis hemos implementado la función sort_result.py La función toma un archivo de entrada que contiene pares clave-valor, la función clasifica los pares clave-valor según los valores numéricos y permite la opción de clasificar en orden ascendente o descendente, dependiendo de la presencia del argumento opcional. Si se proporciona el argumento opcional "-r" o "--reverse", la lista se ordena en sentido inverso. Finalmente, los pares clave-valor ordenados se escriben en un archivo de salida.

```python
import argparse

# arguments
parser = argparse.ArgumentParser(description="Sort key-value pairs based on values.")
parser.add_argument("input_file", help="Input file")
parser.add_argument("output_file", help="Output file")
parser.add_argument(
    "-r", "--reverse", action="store_true", help="Reverse the sort order"
)
args = parser.parse_args()

with open(args.input_file, "r") as file:
    lines = file.readlines()
    data = [line.strip().split("\t") for line in lines]

sorted_data = sorted(data, key=lambda x: float(x[1]), reverse=args.reverse)

# sorted data
with open(args.output_file, "w") as file:
    for pair in sorted_data:
        file.write(f"{pair[0]}\t{pair[1]}\n")
```

Por ejemplo, para utilizar para ordenar el número de reseñas de orden mayor a menor:

python src/sort_result.py output/num_review/part-00000 output/num_review/part-00000_max –r

```
alucloud71@hadoopmaster:~/trabajo$ cat output/num_review/part-00000_max | head -n 30
['B000FKBCX4', 'Spore'] 3292
['B000B9RI0K', 'Xbox LIVE 3 Month Gold Membership']     3091
['B000N5Z2L4', 'Xbox LIVE 12 Month Gold Membership Card']       3082
['B0009VXBAQ', 'Wii']   2341
['B00005NZ1G', 'Halo']  1781
['B000NDRT62', 'Xbox 360 LIVE 4000 Points']     1623
['B00005TNI6', 'Final Fantasy X']       1603
['B000FO4KO8', 'Nintendo DS Lite Polar White']  1570
['B00078ZGTA', 'Nintendo DS Titanium']  1483
['B000066TS5', 'Kingdom Hearts']        1394
['B0000696CZ', 'Grand Theft Auto Vice City']    1374
['B00000JRSB', 'Final Fantasy VII']     1348
['B00005O0I2', 'GREATEST HITS:Grand Theft Auto III']    1239
['B0000296O5', 'Final Fantasy VIII']    1182
['B00005Q8M0', 'Super Smash Bros Melee']        1182
['B00006IJJI', 'Gamecube Console Platinum']     1040
['B0007TFLLC', 'PlayStation Portable (PSP) Value Pack'] 1027
['B00000K2R4', 'Sega Dreamcast Console']        1003
['B00009WNZA', 'The Sims 2']    938
['B00000DMB3', 'The Legend of Zelda']   936
['B0001VGFK2', 'Grand Theft Auto']      925
['B00005ML10', 'Metal Gear Solid 2']    905
['B00008J7NZ', 'Halo 2']        895
['B000084318', 'The Legend of Zelda']   893
['B000067FDW', 'World of Warcraft']     872
['B000KRXAGE', 'Wii Play with Wii Remote']      856
['B000AQ690Y', '28 Days Later [UMD for PSP] (2003)']    854
['B00004Y57G', 'Final Fantasy IX']      802
['B000067FDY', 'Star Wars Galaxies']    782
['B000B43OY4', 'Xbox 360 Pro Console 20GB [Old Version]']       781
```

Y de orden menor a mayor:

python src/sort_result.py output/part-00000 output/part-00000_min

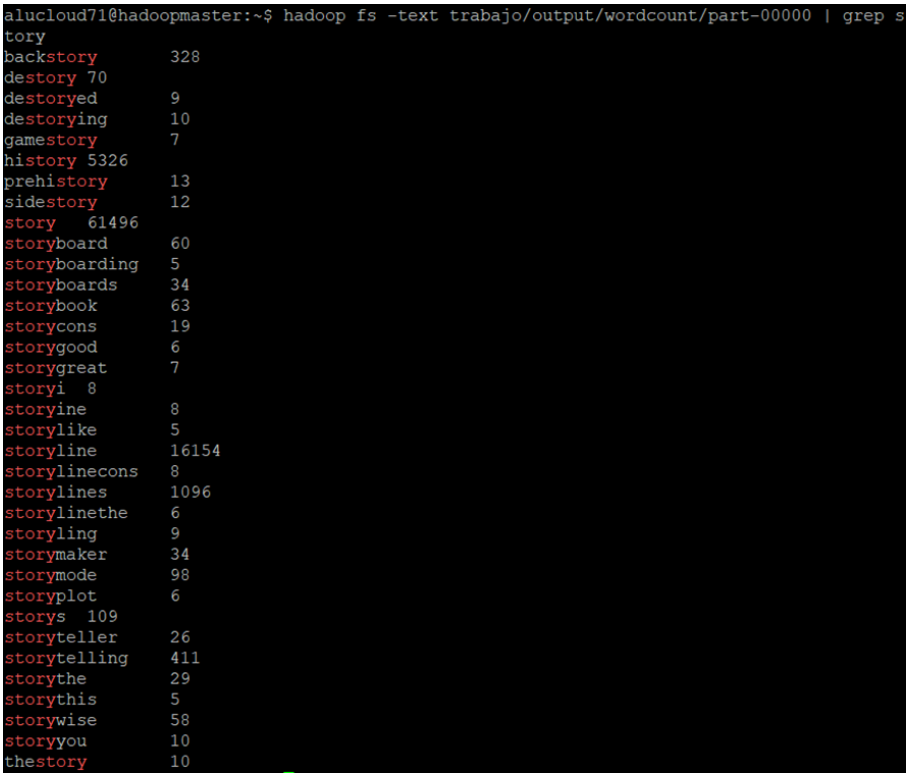
```
alucloud71@hadoopmaster:~/trabajo$ cat output/part-00000_min | head -n 30
aaagh   5
aan     5
aayla   5
abandonned      5
abbility        5
abbot   5
aberration      5
abetter 5
abhorrently     5
abolish 5
aboot   5
abounded        5
abra    5
abraham 5
abruptness      5
abscent 5
absently        5
absol   5
absolut 5
absolutaly      5
absolutelly     5
absolutlely     5
absoutly        5
absulutly       5
absurdities     5
accelaration    5
accelator       5
accellerate     5
accels  5
accentuates     5
```

Por otro lado, para realizar el análisis de valoración de los videojuegos hemos utilizado un comando para introducir las palabras clave y guardarlas en un fichero que pasaremos finalmente a Excel para realizar las gráficas:

El comando es:

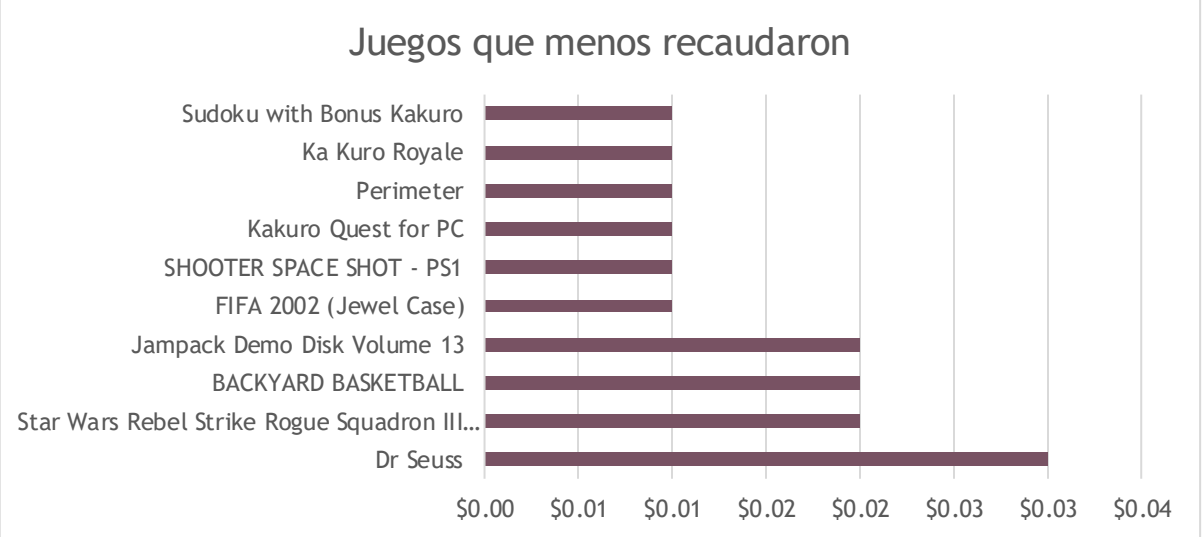
hadoop fs -text trabajo/output/wordcount/part-00000 | grep <key-word> > <key-word>.txt

Y un ejemplo de uso es el siguiente:



5.2 Análisis de los resultados obtenidos a partir de los datos: importe total

Gracias a ordenar los datos podemos observar de los datos, entre otras cosas, que el videojuego que más dinero recaudó fue Final Fantasy VII y uno de los que menos el FIFA 2002 (Jewel Case).



5.3 Análisis de los resultados obtenidos a partir de los datos: Valoración de los videojuegos.

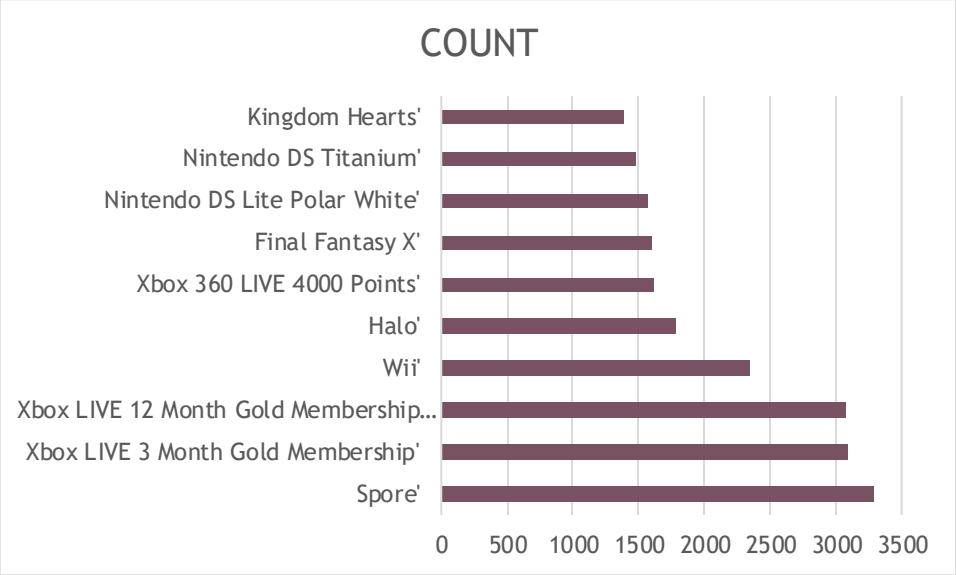
Aquí tenemos listados los juegos con mayor valoración y los que menos tienen.

```
alucloud71@hadoopmaster:~/trabajo$ cat output/part-00000_max | head -n 30
['B00000IWYS', 'Tetris Dx']     5.0
['B00002STRJ', 'International Superstar Soccer Pro 98-PLAYSTION']       5.0
['B00002STXR', 'Maui Mallard in Cold Shadow']   5.0
['B00002SVBA', 'X-COM'] 5.0
['B000035XJA', 'General Chaos'] 5.0
['B000035XLF', 'Zombies Ate My Neighbors']      5.0
['B000035XNC', 'Herzog Zwei']   5.0
['B000035Y34', 'Soul Blazer']   5.0
['B00005NOFK', 'The Weakest Link']      5.0
['B00005OUKK', 'Command & Conquer']     5.0
['B00005TQ1P', 'R.L. Stine Goosebumps'] 5.0
['B000065SQD', 'Operation Flashpoint']  5.0
['B000066BZ1', "Spyro Collectors' Edition"]     5.0
['B0006I5FKQ', 'NFL Street 2']  5.0
['B000ANYFTY', "Marc Ecko's Getting Up"]        5.0
['B000MZ8QWM', 'Nintendo Ds Gameboy Advance Sp Replacement Ac Adapter'] 5.0
['B00023JJUW', 'Game Boy Advance SP Classic NES']       4.97
['B000AQD6H8', 'Zuma (Jewel Case)']     4.96
['B000HFGKN4', 'Super Castlevania IV']  4.96
['B000006P0P', 'Klonoa']        4.95
['B000089SB6', 'Medieval']      4.95
['B000021Y5Q', 'NFL Blitz']     4.94
['B000035Y6N', 'Chrono Trigger']        4.94
['B000035Y76', 'Lufia II']      4.94
['B000035Y78', 'Kirby Super Star']      4.94
['B00004SVNS', 'Bubble Bobble'] 4.94
['B00004TTTV', 'NFL Blitz 2001']        4.94
['B00004XOWL', "Baldur's Gate II"]      4.94
['B00007JQTB', "Neverwinter Nights Collector's Edition"]        4.94
['B00001IVR8', 'StarCraft Expansion Pack']      4.93
```

```
alucloud71@hadoopmaster:~/trabajo$ python src/sort_result.py output/part-00000 output/
part-00000_min
alucloud71@hadoopmaster:~/trabajo$ cat output/part-00000_min | head -n 30
['B000FBF676', 'Kitty Luv']     1.26
['B0002NS7UU', 'Mall of America Tycoon']        1.27
['B000E8Z7Y2', 'NRA Gun Club']  1.29
['B000J14PHY', 'Super Aneurysm/Pop Pop Pop (Jewel Case)']       1.3
['B0002IBEQO', 'World Poker Championship']      1.31
['B0002NS7VO', 'Cold Case Files']       1.33
['B000EI3U0A', 'Curious George']        1.33
['B00007E1KC', "Cinderella's Castle Designer"]  1.36
['B00007E7AC', 'Big Biz Tycoon']        1.36
['B000AAQZPE', 'Namco Museum 50th Anniversary'] 1.45
['B0000CGB2N', 'Drake'] 1.5
['B000I8O8GQ', 'Major League Baseball 2K7']     1.5
['B000HG78XE', 'World Series of Poker'] 1.53
['B000212VGS', 'Screen Test DVD Game']  1.57
['B000FKBCX4', 'Spore'] 1.57
['B000MIO1O6', 'Sudokuro']      1.57
['B000189K34', 'Fight Club']    1.58
['B00004XS3N', 'Pinball Madness 3']     1.6
['B00004YKI6', 'Dark Angel']    1.62
['B000056HGD', 'Top Gear Dare Devil']   1.62
['B00005V131', 'Sniper']        1.62
['B000FJ16OU', 'Total 3D Home & Landscape Design Suite Version 9 [Old Version]']     1
.62
['B000HRZIL2', 'Riddle of the Sphinx']  1.62
['B00000JHPT', 'Superman']      1.65
['B000BKF2I4', 'Star Wars Galaxies']    1.66
['B000ENWU0Q', 'Crazy Burger (Jewel Case)']     1.69
['B000EO0WPU', 'Mahjongg Tiles Of Time']        1.71
['B000056QGR', 'NCAA Final Four']       1.72
['B00005K1Z4', "Frank Herbert's Dune"]  1.73
['B000F3JDPA', 'PSP iFM FM Tuner']      1.73
```

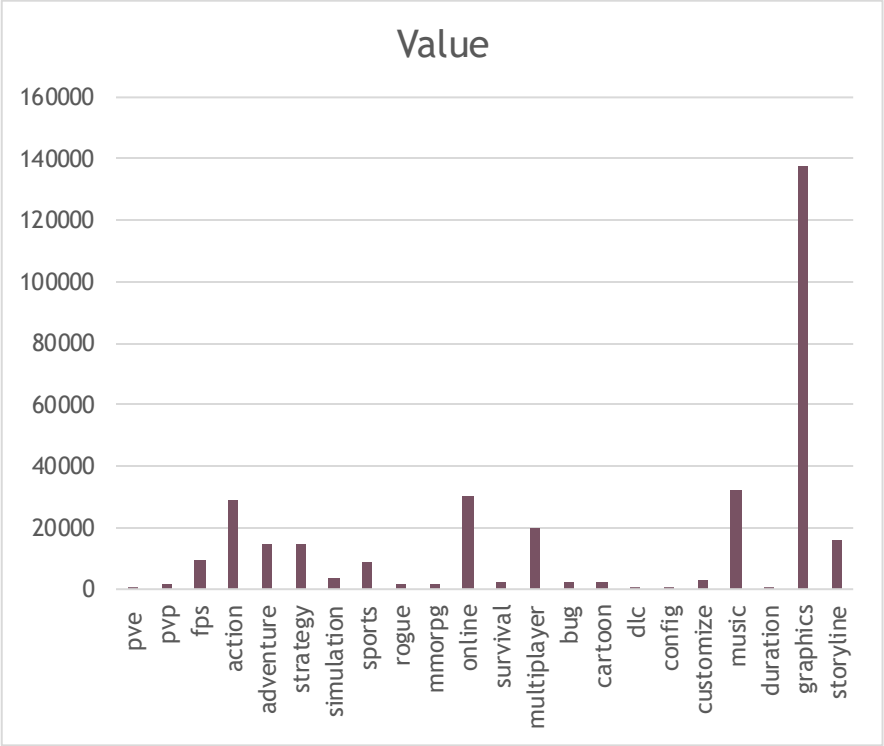
5.4 Análisis de los resultados obtenidos a partir de los datos: Popularidad de los videojuegos.

En este apartado mostráramos una gráfica donde se muestran los videojuegos y consolas más populares, donde la popularidad se valora en la cantidad de reseñas que tuvo el videojuego o la consola.



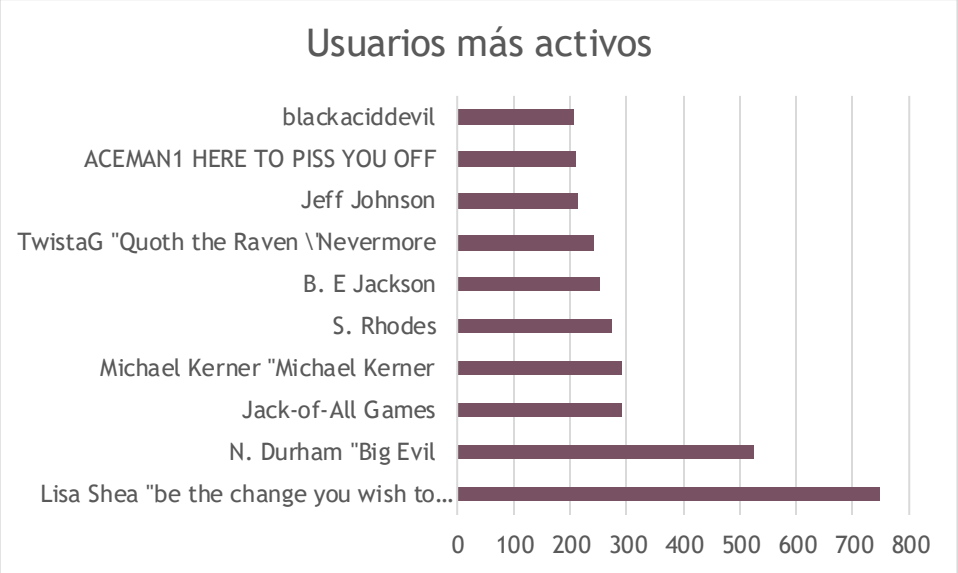
5.5 Análisis de los resultados obtenidos a partir de los datos: Palabras clave.

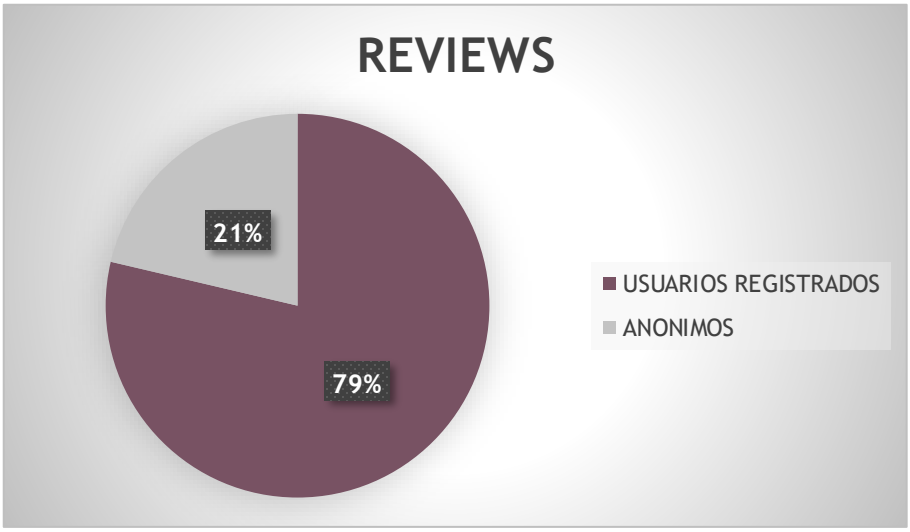
Hemos sacado también una gráfica con las palabras clave para poder averiguar que tipo de juego y características de este valora más un jugador, en este caso a los jugadores les gustan los juegos online y que tengan buenos gráficos.



5.6 Análisis de los resultados obtenidos a partir de los datos: Actividad de los usuarios.

Respecto a la actividad de los usuarios hemos sacado dos gráficas, la primera nos indica que usuarios son los más activos a la hora de publicar reseñas y la segunda nos muestra el porcentaje de usuarios anónimos que comentan contra los que están registrados.





6. Conclusión.

En conclusión, el uso de la herramienta de Hadoop y el modelo de programación MapReduce es de gran importancia en este trabajo académico de análisis de datos de reseñas de videojuegos. Hadoop proporciona un entorno escalable y distribuido que permite procesar grandes volúmenes de datos de manera eficiente y efectiva. A través de la implementación de funciones mapper y reducer, se ha logrado extraer información valiosa y relevante de los datos de reseñas.

La utilización de Hadoop y MapReduce ha permitido realizar análisis exhaustivos sobre los videojuegos y los usuarios en base a las características, puntuaciones, número de reseñas y comportamiento de los usuarios. Estos datos analizados proporcionan una visión detallada de los videojuegos más populares, las preferencias de los usuarios y las características más valoradas.

Estos datos pueden ser utilizados para tomar decisiones estratégicas en la industria de los videojuegos, como la identificación de patrones de éxito en los videojuegos más populares, la mejora de las características valoradas por los usuarios y la personalización de la oferta de videojuegos. Además, la información sobre los usuarios más activos puede ser utilizada para la segmentación de mercado y la creación de estrategias de marketing dirigidas.

7. Trabajo futuro y mejoras.

Una mejora interesante para este trabajo académico sería cruzar los datos de reseñas de videojuegos con información adicional, como el historial de compras de los usuarios, preferencias de género, interacciones en redes sociales u otros datos demográficos relevantes. Esto permitiría construir modelos predictivos para determinar qué juegos podrían interesarle a un usuario en particular. Estos modelos se podrían utilizar para ofrecer anuncios personalizados, recomendaciones de videojuegos y mejorar la experiencia del usuario en plataformas de venta de videojuegos.

Además del análisis de las características de los videojuegos, sería valioso estudiar los patrones de comportamiento de los usuarios en relación con las reseñas y sus interacciones con los videojuegos. Esto podría involucrar la identificación de patrones de juego, tiempos de sesión, preferencias de juego en línea, preferencias de plataforma, entre otros. El análisis de estos patrones ayudaría a comprender mejor a los usuarios y a adaptar la oferta de videojuegos y servicios en consecuencia.

Otra línea de trabajo interesante sería utilizar técnicas de análisis predictivo para identificar los videojuegos que tienen un mayor potencial de ventas futuras en función del título. Se podrían aplicar técnicas de procesamiento del lenguaje natural y aprendizaje automático para analizar características como el título, la sinopsis o los géneros de los videojuegos y predecir su éxito en términos de ventas o popularidad.

Por último, una mejora importante sería evaluar cuál es el impacto de las reseñas de videojuegos en la toma de decisiones de compra de los usuarios. Esto podría involucrar la construcción de modelos que tengan en cuenta la puntuación, el contenido y la relevancia de las reseñas para predecir la probabilidad de compra de un juego en función de las opiniones de otros usuarios. Esta información podría ser valiosa para los desarrolladores de videojuegos y los minoristas para comprender cómo las reseñas influyen en las decisiones de los consumidores y adaptar sus estrategias de marketing en consecuencia.

8. Contribución de los participantes.

En este apartado vamos a indicar la contribución de los miembros en el trabajo.

1. Introducción. Cai

2. Entorno Hadoop con Python. Cai

3. Implementación de las funciones mapper y reducer. Diego

4. Datos a analizar. Diego

4.1 Preparación de los datos. Cai

4.2.1 Datos sobre los videojuegos de las reseñas. Diego

4.2.2 Número de reseñas. Diego

4.2.3 Importe total. Cai

4.2.4 Valoración de los videojuegosCai

4.2.5 Características Cai

4.3.1 Datos sobre los usuarios de las reseñas. Diego

4.3.2 Usuarios más activos. Diego

5.1 Análisis de los resultados obtenidos a partir de los datos: Sorteo de datos y palabras clave. Cai y Diego

5.2 Análisis de los resultados obtenidos a partir de los datos: importe total Cai y Diego

5.3 Análisis de los resultados obtenidos a partir de los datos: Valoración de los videojuegos. Cai y Diego

5.4 Análisis de los resultados obtenidos a partir de los datos: Popularidad de los videojuegos. Cai y Diego

5.5 Análisis de los resultados obtenidos a partir de los datos: Palabras clave. Cai y Diego

5.6 Análisis de los resultados obtenidos a partir de los datos: Actividad de los usuarios. Cai y Diego

6. Conclusión. Cai y Diego

7. Trabajo futuro y mejoras. Cai y Diego