PROGRAMACIÓN
HITO GRUPAL

Jose Antonio Ures
Allam E. Miranda Carrasco
Fernando Trujillo

Investigación

1. **Fuentes de datos**

   Data Lake
   Es un tipo de almacenamiento o repositorio de un gran conjunto de datos en bruto que todavía no tienen una finalidad definida.

   Su nombre hace referencia a la flexibilidad, acceso compartido y manejo en tiempo real de un gran volumen de datos entre diferentes usuarios que se benefician o trabajan con dichos datos. Evidentemente, debido a la gran cantidad de información que manipula, trabaja con herramientas del Big Data.
   Su principal objetivo radica en el almacenamiento esquematizado de los datos para poder procesar según los intereses de estudio.
   Uno de sus factores más importantes es que los datos almacenados se actualizan y transforman constantemente, se mantiene vivo gracias a su gestión permanente de la información.
   Es un tipo de repositorio muy utilizado en el ámbito empresarial debido a sus ventajas como estrategia empresarial al reducir costes, aumentar la asertividad en la toma de decisiones y permitir que los usuarios de la empresa unifiquen su conocimiento sobre la información obtenida.
   Sus principales características:
   - Ayuda al manejo del Big Data
   - La posibilidad de gestionar y transformar los datos mientras se encuentran almacenados.
   - La cercanía e interacción que permite con los usuarios.
   - Cuenta con muchas herramientas y productos para cumplir con su objetivo de almacenamiento eficaz.
   - Los metadatos se gestionan de manera automatizada ahorrando tiempo y trabajo a la hora de implementar este repositorio.
   - Modificar la privacidad de los datos y establecer quién tiene acceso, quién solo puede verlos, quién puede modificar o no la información original, etc.

2. **Hadoop y Spark**

   Hadoop
   Es una estructura de software de código abierto para almacenar datos y ejecutar aplicaciones en clústeres de hardware comercial. Proporciona almacenamiento masivo para cualquier tipo de datos, enorme poder de procesamiento y la capacidad de procesar tareas o trabajos concurrentes virtualmente ilimitados.

   Características:
   - Procesamiento distribuido.

- Eficiente.
- Económico.
- Fácilmente escalable.
- Tolerante a fallos.
- Open source.

Spark

Apache Spark es un motor de estadísticas unificado con el que se procesan datos a gran escala con módulos integrados para SQL, transmisión, aprendizaje automático y procesamiento de grafos. Spark puede ejecutarse en Apache Hadoop, Apache Mesos, Kubernetes, por sí solo, en la nube y en varias fuentes de datos.

Características:

- Procesamiento en memoria: Apache Spark es 100 veces más rápido en memoria y 10 veces más rápido en disco que Hadoop MapReduce, para ello necesita más recursos.
- Soporta múltiples lenguajes: Spark tiene APIs disponibles en los lenguajes Java, Scala, Python y R.
- Analítica avanzada: Para ello, soporta consultas SQL y su uso para Machine Learning con librerías de data science como MLlib y GraphX.
- Abstracción RDD (Resilient Distributed Dataset): consiste en una colección inmutable de elementos en memoria distribuída.
- Evaluación perezosa: Las transformaciones sobre los datos solo se resuelven al ejecutar una acción sobre ellos.

3. Python

Python y el Big Data es una de las combinaciones más valiosas a día de hoy. los datos y la información sirven para ser más eficientes, tomar buenas decisiones y conocer a tus clientes.
De hecho, el lenguaje de programación Python y el *big data* están muy relacionados. No por nada se lo considera el mejor lenguaje de programación para el análisis de datos.

Lo cierto es que escoger un lenguaje de programación para el *big data* depende del proyecto que tengas entre manos. No obstante, sea cual sea tu objetivo, Python siempre será una opción de lo más adecuada. Esto se debe a que, además de ser un lenguaje en desarrollo constante, su código simple y sus inmensas bibliotecas (SciPy, Pandas, Numpy o Scikit-Learn...) hacen que sea la opción preferida para la mayoría de los programadores. Algo lógico, ya que comparado con otros lenguajes, Python tiene un código y una sintaxis simple que hace que sea muy fácil de aprender. Con unas pocas líneas de código, puedes ejecutar programas sin más complicaciones.

Además, y esto es una gran ventaja, es de código abierto. Por ello, cualquier persona tiene acceso a sus recursos de forma gratuita. Y, gracias a la gran comunidad de usuarios de Python, podrás encontrar las respuestas a tus dudas fácilmente. Así mismo, otra gran ventaja del lenguaje es su gran velocidad de procesamiento

La forma más común de usar Python para el análisis de datos es para crear y gestionar rápidamente varias estructuras de datos. Por ejemplo, la biblioteca Panda ofrece una gran cantidad de herramientas para analizar, manipular e incluso representar las estructuras de datos y conjuntos de datos complejos.

Scala:

Scala es uno de los lenguajes de programación más usados para el manejo y desarrollo del Big Data. Esto es gracias a lo intuitivo, conciso y preciso que es como lenguaje de programación.

Powebi:

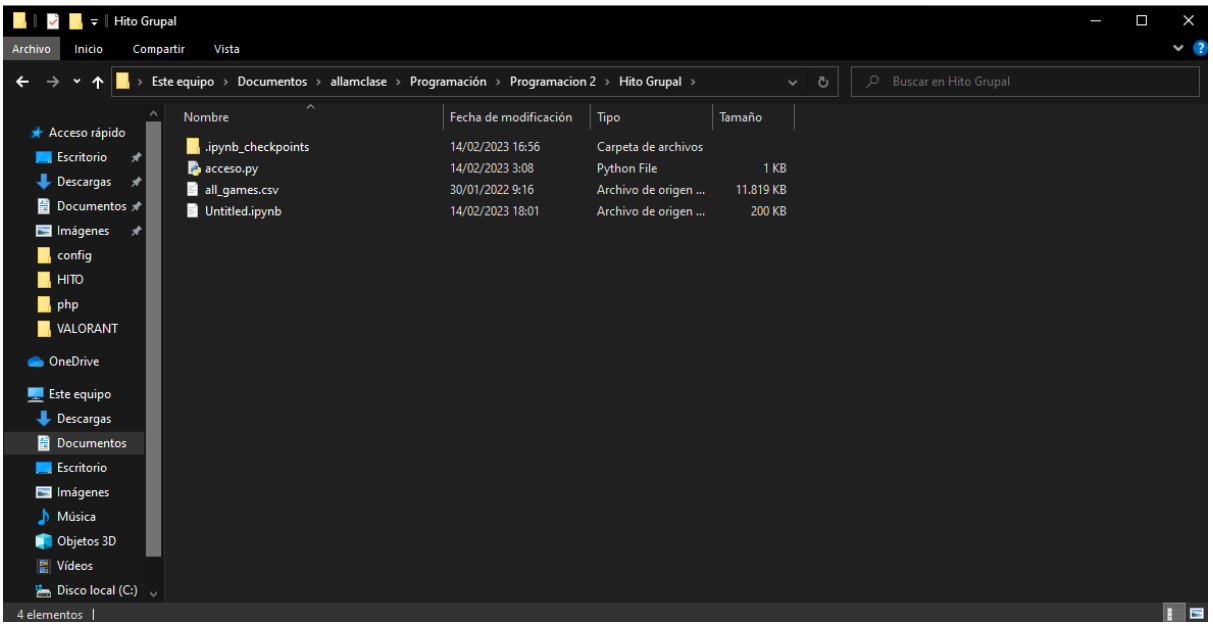
Power BI proporciona servicios de BI basados en la nube, conocidos como "Power BI Services", junto con una interfaz basada en escritorio, denominada "Power BI Desktop". Ofrece capacidades de almacenamiento de datos, incluyendo preparación de datos, descubrimiento de datos y paneles interactivos. En marzo de 2016, Microsoft lanzó un servicio adicional llamado "Power BI Embedded" en Azure, su plataforma en la nube. Uno de los principales diferenciadores del producto es la capacidad de cargar visualizaciones personalizadas.

Tableau:

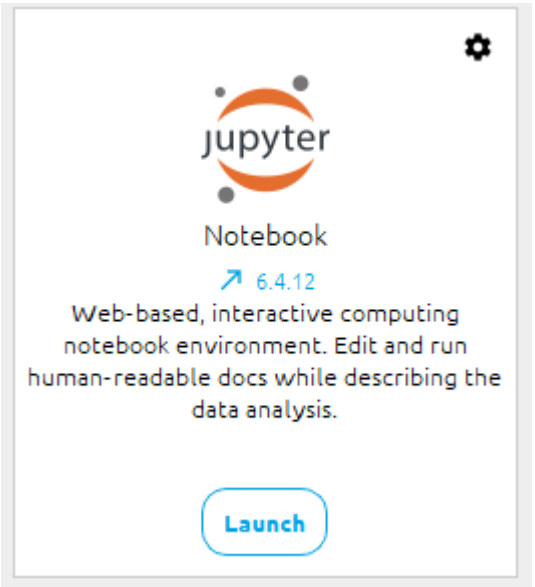
Tableau es una plataforma de análisis visual que transforma la manera en que usamos los datos para resolver problemas. Además, permite a las personas y las organizaciones sacar el máximo partido de los datos. Hace que sea más fácil explorar y administrar los datos. Asimismo, permite descubrir y compartir información más rápidamente a fin de generar grandes cambios en los negocios y en el mundo.

Implementación de la investigación

Para la fase 2 del proyecto usaremos distintas formas de ver los datos de un archivo csv pesado, en este caso vamos a usar una base de datos conseguida en **Kaggle** de todos los juegos del mundo, este archivo pesa unos 11 MB en un csv, en Big Data la mayor parte del trabajo se realiza en Jupyter NoteBook esto se debe a que la implementación de distintos softwares como puede ser Spark son más fáciles de implementar y da un rendimiento muy alto.



Tenemos una base de datos estructurada ya que no existe ningún dato no estructurado ya sea imágenes o audios, abrimos Jupyter con Conda ya que nos ofrece las librerías necesarias para el control de la información, en este caso Pandas y haremos uso de PyPlot.



Abrimos Jupyter desde Conda y entramos a nuestra carpeta del proyecto y vamos a crear un nuevo archivo de Python para importar las distintas librerías.

```
+ Code  + Markdown

import pandas as pd
import plotly.graph_objs as go
from plotly.offline import iplot

df = pd.read_csv("all_games.csv")
df

[12]
```

Importamos Pandas como pd y también PlotLy vamos a leer nuestro primer csv en este caso el de 'all_games.csv' que nos da la información de todas los juegos del mundo y todos los datos de los mismos.

Para esto vamos a crear un DataFrame, un DataFrame de Pandas es una estructura de datos en forma de tabla que permite almacenar y manipular datos de manera eficiente y sencilla en Python.

Sabiendo esto creamos nuestro dataframe de este csv.

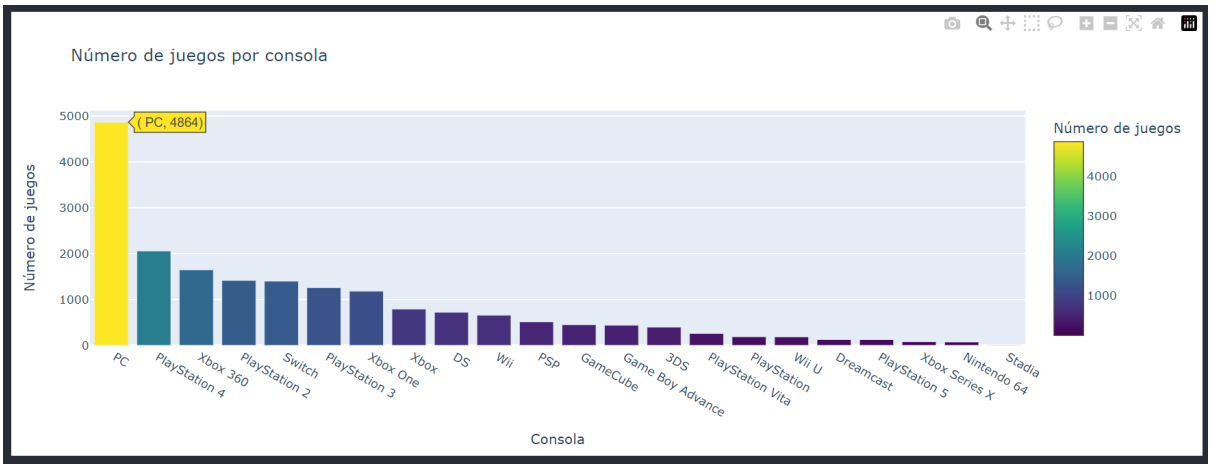
	name	platform	release_date	summary	meta_score	user_review
0	The Legend of Zelda: Ocarina of Time	Nintendo 64	November 23, 1998	As a young boy, Link is tricked by Ganondorf, ...	99	9.1
1	Tony Hawk's Pro Skater 2	PlayStation	September 20, 2000	As most major publishers' development efforts ...	98	7.4
2	Grand Theft Auto IV	PlayStation 3	April 29, 2008	[Metacritic's 2008 PS3 Game of the Year; Also ...	98	7.7
3	SoulCalibur	Dreamcast	September 8, 1999	This is a tale of souls and swords, transcendi...	98	8.4
4	Grand Theft Auto IV	Xbox 360	April 29, 2008	[Metacritic's 2008 Xbox 360 Game of the Year; ...	98	7.9
...	...	...	...	...	...	...
18795	Fast & Furious: Showdown	Xbox 360	May 21, 2013	Fast & Furious: Showdown takes some of the fra...	22	1.3
18796	Drake of the 99 Dragons	Xbox	November 3, 2003	Drake is out for revenge in a supernatural Hon...	22	1.7
18797	Afro Samurai 2: Revenge of Kuma Volume One	PlayStation 4	September 22, 2015	Head out on a journey of redemption, driven by...	21	2.9
18798	Infestation: Survivor Stories (The War Z)	PC	October 15, 2012	(Formerly known as "The War Z") It has been 5 ...	20	1.7
18799	Leisure Suit Larry: Box Office Bust	PC	March 31, 2009	The Leisure Suit Larry: Box Office Bust video ...	20	2.0

Al hacer print de esto nos devuelve una tabla de más de 18000 filas con todos nuestros datos y columnas de el csv ahora vamos a realizar una serie de consultas.

Para esto Plotly nos va a venir perfecto, Plotly es una biblioteca de visualización de datos interactiva de código abierto que permite crear gráficos y visualizaciones en una variedad de lenguajes de programación, incluyendo Python. Plotly es conocido por sus gráficos interactivos y dinámicos que se pueden ajustar y personalizar en tiempo real, lo que los hace útiles para explorar y presentar datos complejos.

```python
# Se crea una Serie con el recuento de la columna "platform" del DataFrame "df"
comp_counts = df["platform"].value_counts()

# Se crea una lista con un objeto "Bar" de Plotly, que contendrá los datos y las opciones de visualización
data = [go.Bar(
    x=comp_counts.index,
    y=comp_counts.values,
    marker=dict(
        color=comp_counts.values,
        colorscale='Viridis',
        colorbar=dict(title='Número de juegos')
    )
)]

# Se crea un objeto "Layout" de Plotly, que contiene las opciones de diseño y de ejes del gráfico
layout = go.Layout(title="Número de juegos por consola", xaxis=dict(title="Consola"), yaxis=dict(title="Número de juegos"))

fig = go.Figure(data=data, layout=layout)

# Se muestra el gráfico utilizando la función "iplot" de Plotly
iplot(fig)
```

Con PlotLy podemos crear consultas como esta donde hemos creado un histograma con el número de juegos por cada consola. Nos muestra una visualización gráfica como esta:



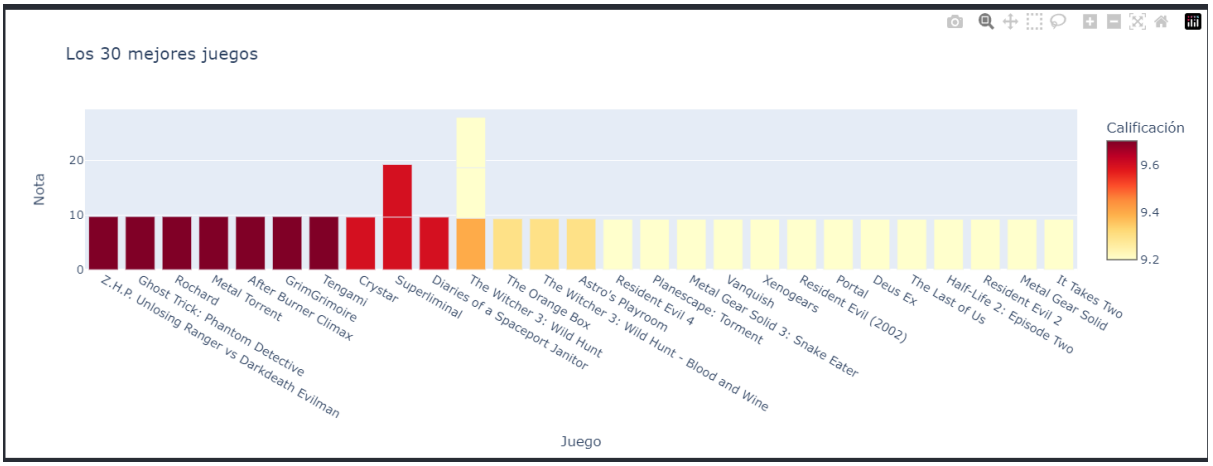
PlotLy es mucho más visual eficiente y escalable que MatPlotLib posteriormente veremos un ejemplo mucho más avanzado.

Con MatPlotLib podemos transformar estas consultas en gráficos más visuales por ejemplo creamos un DataFrame de 20 líneas que muestre los datos de antes pero en una barra horizontal.

```python
# Se convierte la columna "user_review" del DataFrame "df" a valores numéricos. Los valores no numéricos se convierten en NaN.
df["user_review"] = pd.to_numeric(df["user_review"], errors='coerce')

# Se ordena el DataFrame "df" por la columna "user_review" en orden descendente (de mayor a menor) y se seleccionan los primeros 30 juegos.
df_top30 = df.sort_values("user_review", ascending=False).head(30)

# Se crea una lista con un objeto "Bar" de Plotly, que contendrá los datos y las opciones de visualización
data = [go.Bar(
    # El eje x muestra los nombres de los juegos del DataFrame "df_top30"
    x=df_top30["name"],
    # El eje y muestra las calificaciones de los juegos del DataFrame "df_top30"
    y=df_top30["user_review"],
    # Se configura la barra para que su color corresponda a la calificación de los juegos, utilizando la escala de color "YlOrRd"
    marker=dict(
        color=df_top30["user_review"],
        colorscale='YlOrRd',  # Escala de colores
        # Se agrega un colorbar para que la escala de colores esté etiquetada
        colorbar=dict(title='Calificación')
    )
)]

# Se crea un objeto "Layout" de Plotly, que contiene las opciones de diseño y de ejes del gráfico
layout = go.Layout(
    title="Los 30 mejores juegos",
    xaxis=dict(title="Juego"),
    yaxis=dict(title="Nota")
)

# Se crea un objeto "Figure" de Plotly, que contiene tanto los datos como las opciones de diseño
fig = go.Figure(data=data, layout=layout)

# Se muestra el gráfico utilizando la función "show" del objeto "Figure"
fig.show()
```

0.0s

El siguiente código lo usaremos para ordenar los juegos por nota y con un color de degradado distinto vamos a ver el visual.

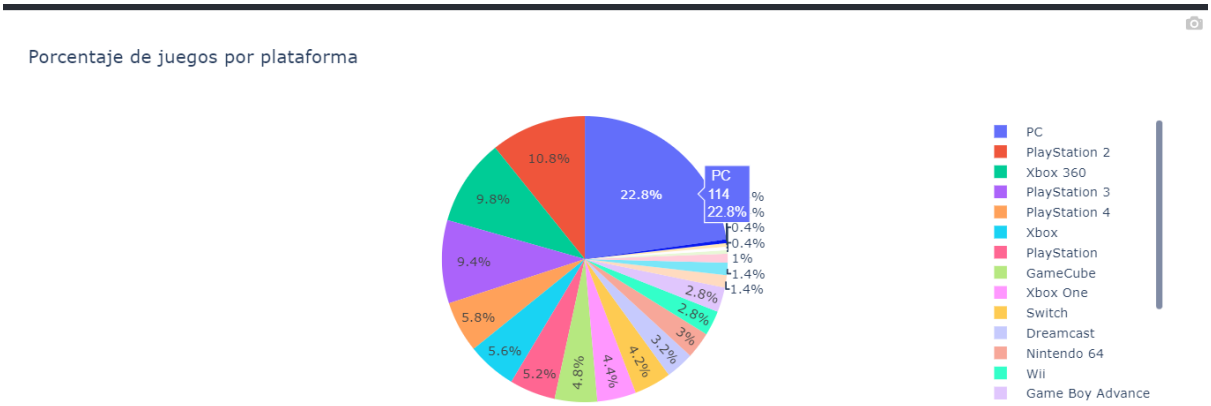


Nos muestra nuestro histograma con las notas

```python
df_top500 = df.head(500)

# Contar el número de juegos por plataforma
platform_counts = df_top500["platform"].value_counts()

# Crear el gráfico de tarta
data = [go.Pie(
    labels=platform_counts.index,
    values=platform_counts.values,
    textinfo='percent',
)]

layout = go.Layout(title="Porcentaje de juegos por plataforma")

fig = go.Figure(data=data, layout=layout)
iplot(fig)
```

0.0s

Ahora mostramos un diagrama de tarta para que nos diga nuestra consulta anterior de las consolas pero con porcentajes

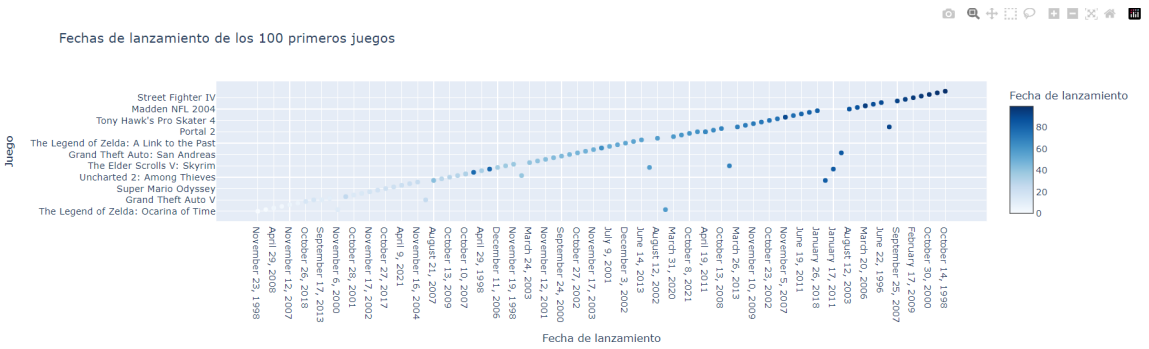


En este caso de los primeros 500 juegos.

```python
# Seleccionar los primeros 100 juegos
df_top100 = df.head(100)

# Crear el gráfico de dispersión con colores graduados
data = [go.Scatter(
    x=df_top100["release_date"],
    y=df_top100["name"],
    mode='markers',
    marker=dict(
        color=df_top100.index,
        colorscale='Blues',  # Escala de colores
        colorbar=dict(title='Fecha de lanzamiento')  # Título de la barra de colores
    ),
    text=df_top100["name"] + '<br>' + df_top100["release_date"]  # Texto con el nombre y la fecha de lanzamiento
)]

layout = go.Layout(
    title="Fechas de lanzamiento de los 100 primeros juegos",
    xaxis=dict(title="Fecha de lanzamiento"),
    yaxis=dict(title="Juego")
)

fig = go.Figure(data=data, layout=layout)
iplot(fig)
```

✓ 0.0s

Ahora mediante un gráfico de dispersión mostraremos los juegos y su fecha de salida.



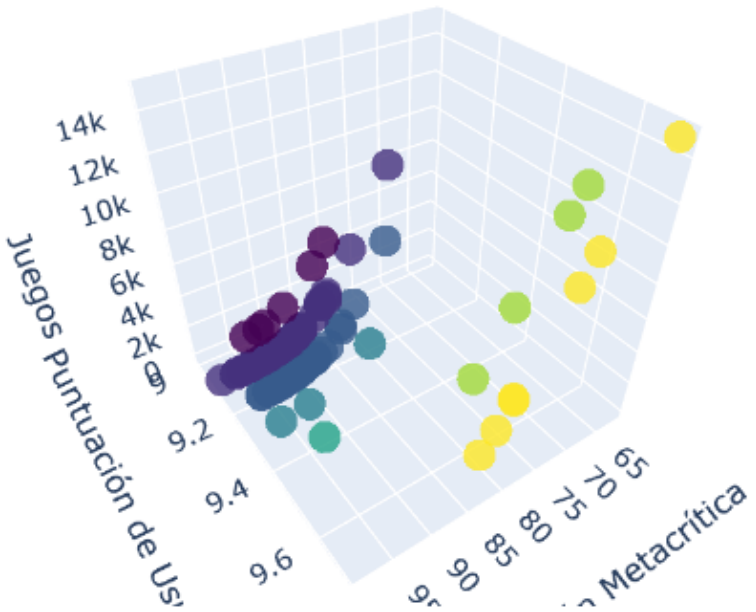
Algo a apreciar es que PlotLy trabaja más rápido con los csv que matplotlib.

Por último vamos a ver una consulta curiosa con un gráfico 3D.

```
# Convertir los valores "tbd" en NaN
df["user_review"] = pd.to_numeric(df["user_review"], errors='coerce')

# Ordenar el DataFrame por "user_review" de mayor a menor y seleccionar los primeros 100 juegos
df_top100 = df.sort_values("user_review", ascending=False).head(100)

# Crear el gráfico 3D
trace = go.Scatter3d(
    x=df_top100["meta_score"],
    y=df_top100["user_review"],
    z=df_top100.index,
    mode="markers",
    marker=dict(
        size=8,
        color=df_top100["user_review"],
        colorscale="Viridis",
        opacity=0.8
    ),
    text=df_top100["name"],
    hoverinfo="text"
)

data = [trace]

layout = go.Layout(
    title="Los 100 mejores juegos",
    scene=dict(
        xaxis=dict(title="Puntuación Metacrítica"),
        yaxis=dict(title="Puntuación de Usuarios"),
        zaxis=dict(title="Juegos")
    )
)

fig = go.Figure(data=data, layout=layout)
iplot(fig)
```

✓ 0.0s

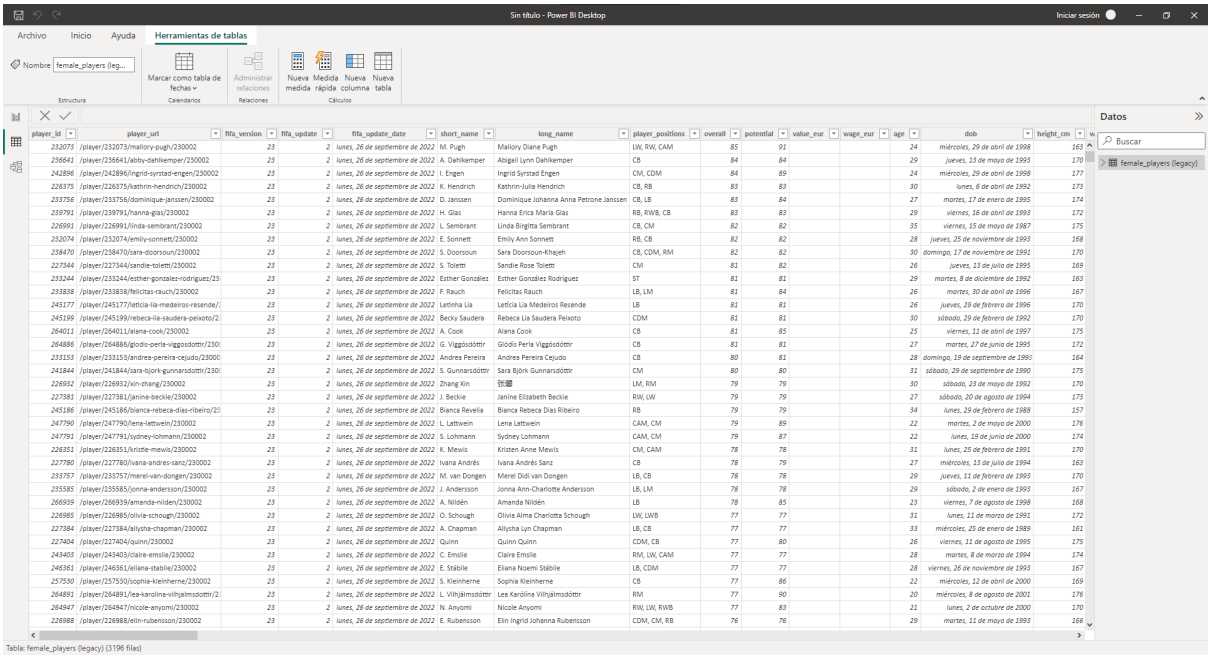
PlotLy nos da la capacidad de ver nuestra información en un espacio interactivo 3D gracias al Scatter 3D.



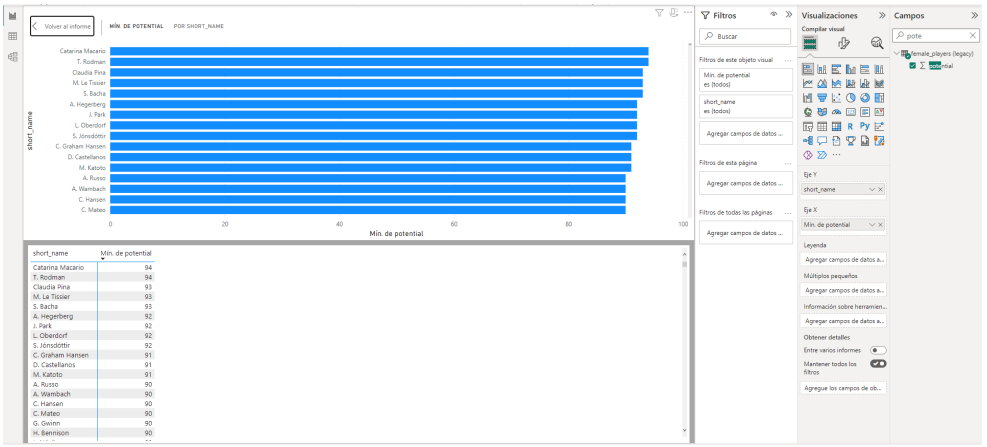
En resumen, Python y Pandas son una combinación de lenguaje de programación y biblioteca de análisis de datos que se enfocan en la simplicidad y la eficiencia en la manipulación de datos, mientras que Scala y Spark son una combinación de lenguaje de programación y marco de trabajo de análisis de datos que se enfocan en la escalabilidad y el procesamiento en paralelo de grandes cantidades de datos.

Tras analizar el manejo de datos con Python vamos a ver cómo se manejan los datos con PowerBi una herramienta mucho más visual y sin necesidad de código para ver gráficos de nuestros csv o base de datos.

Vamos a hacer las consultas con una base de datos de jugadoras femeninas en PowerBI, primero visualizamos el csv que en PowerBi es muy simple.

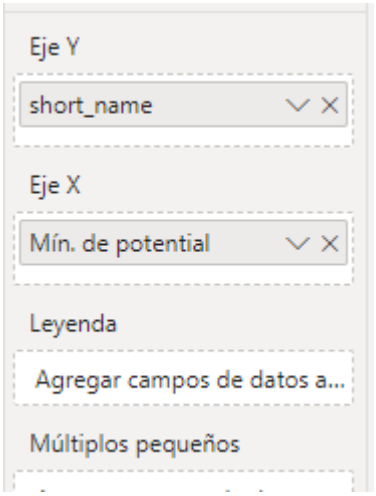


Nos da la información del csv completo. Hacemos la consulta para ver el potencial de las jugadoras ordenado de mayor a menor.



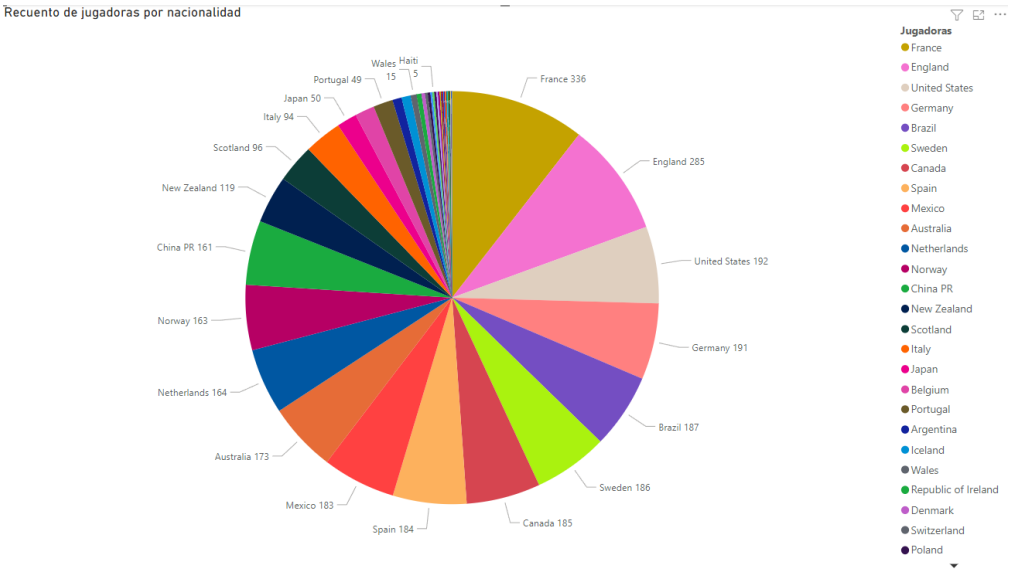
A la derecha tenemos las variables que vamos a usar en el eje x y eje y.

En el panel visual tenemos el gráfico de barras con los nombres de las jugadoras.



Y por último en la parte inferior la tabla con solo los valores elegidos.

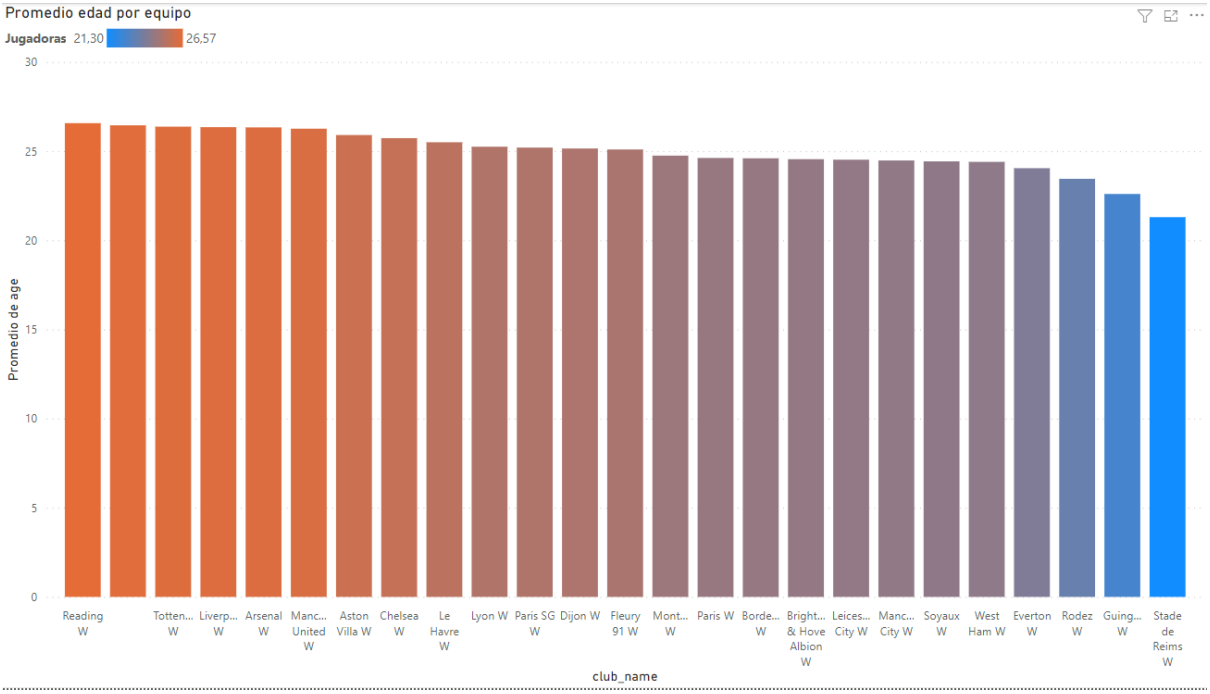
Hacemos la misma consulta de las jugadoras por país solo que en PowerBi nos da mucha más información de manera más visual. Vemos los países, el número de jugadoras y el listado entero.



Tenemos también la tabla de los países y número de Jugadoras.

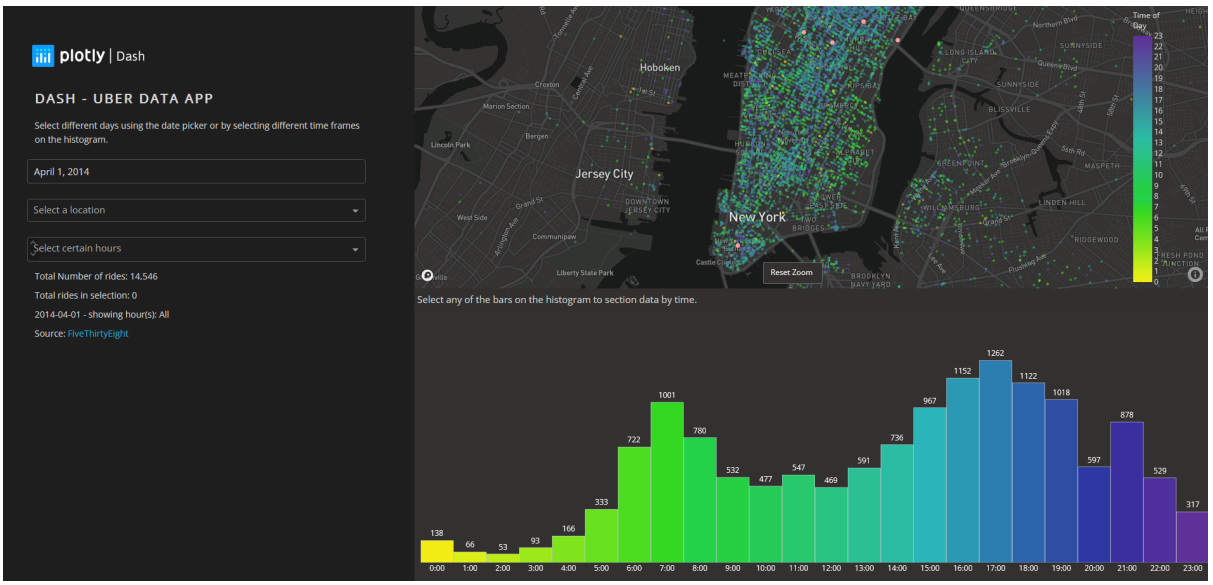
nationality_name	Recuento de short_name
France	336
England	285
United States	192
Germany	191
Brazil	187
Sweden	186
Canada	185
Spain	184
Mexico	183
Australia	173
Netherlands	164
Norway	163
China PR	161
New Zealand	119
Scotland	96
Italy	94
Japan	50
Belgium	49

Ahora haremos la consulta del promedio de edad por equipo.



Nos muestra un gráfico de barras con todos los equipos mucho más visual que en pandas y es más eficiente y rápido, PowerBi aparte de las propias herramientas que tiene, contiene un apartado para añadir código de Python.

En este panel podemos utilizar Pandas y cualquier librería de Python relacionada con análisis de datos.



Proyecto Uber Explicado en Ivoox.

Análisis de la Fase 2

Pandas y Power BI son dos herramientas muy útiles para el acceso y análisis de datos.

Pandas es una biblioteca de Python que proporciona estructuras de datos y funciones para manipular y analizar grandes conjuntos de datos. Es una herramienta muy popular y ampliamente utilizada en la comunidad de ciencia de datos debido a su facilidad de uso y su capacidad para realizar tareas de manipulación de datos de manera rápida y eficiente. Además, es posible integrar Pandas con otros módulos y bibliotecas de Python para realizar tareas avanzadas de análisis de datos.

Power BI es una plataforma de análisis de datos de Microsoft que permite visualizar y compartir información a partir de diferentes fuentes de datos. Es una herramienta intuitiva y fácil de usar que permite crear visuales de manera rápida y eficiente. Además, es posible integrar Power BI con otras aplicaciones de Microsoft, como Excel, para una experiencia de análisis de datos más completa.

Los archivos CSV son una buena forma de acceso a datos porque son fáciles de crear y leer, son compatibles con la mayoría de las aplicaciones de software, y son eficientes en términos de almacenamiento. Además, los archivos CSV permiten el intercambio de datos entre diferentes sistemas y plataformas.

Otras formas de acceso a datos incluyen bases de datos relacionales y no relacionales, archivos JSON, XML, hojas de cálculo, servicios web y API (Application Programming Interface). Cada forma de acceso a datos tiene sus propias ventajas y desventajas, y la elección de la mejor opción dependerá de los requisitos específicos de la aplicación y del tipo de datos que se están accediendo.

Además de Pandas y Power BI, hay muchas otras herramientas y técnicas de análisis de datos que podrían ser mejores en función de las necesidades específicas de su proyecto. Algunas recomendaciones incluyen:

- Utilizar herramientas de análisis de datos en la nube, como Google BigQuery o Amazon Redshift, que permiten el procesamiento de grandes conjuntos de datos de manera escalable y eficiente en términos de costos.

- Utilizar herramientas de aprendizaje automático, como TensorFlow o PyTorch, para realizar tareas avanzadas de análisis de datos, como la clasificación de datos.

- Utilizar herramientas de visualización de datos avanzadas, como Tableau.

También cabe mencionar Scala que hubiese sido más escalable que python y eficaz.

Acceso a Datos mediante ficheros

En el pasado, el acceso a los datos desde los archivos dependía del sistema operativo y del lenguaje de programación utilizado. En general, se debía abrir el archivo, leer su contenido y luego cerrarlo explícitamente.

En el caso de los sistemas operativos de tipo UNIX, se podía utilizar el comando "cat" para visualizar el contenido de un archivo en la línea de comandos. Para acceder a los datos desde un programa, se podía utilizar la función "fopen" de C para abrir el archivo y luego leer su contenido mediante la función "fread".

En el caso de los sistemas operativos de tipo DOS y Windows, se podía utilizar el comando "type" para visualizar el contenido de un archivo en la línea de comandos. Para acceder a los datos desde un programa, se podía utilizar la función "Open" de Visual Basic para abrir el archivo y luego leer su contenido mediante la función "Input".

En resumen, el acceso a los datos desde los archivos era un proceso relativamente manual que requería un conocimiento detallado de los sistemas operativos y los lenguajes de programación utilizados. Sin embargo, con el tiempo, la mayoría de los lenguajes de programación han proporcionado librerías y herramientas más avanzadas para simplificar este proceso y hacerlo más accesible para los desarrolladores.

Acceso a Datos mediante bases de datos

La definición de bases de datos es «aquel conjunto de datos almacenados y estructurados según sus características o tipología para ser utilizados o consultados posteriormente«.

Hasta hace relativamente pocos años, las bases de datos eran analógicas, es decir, contenían información en papel o textos impresos. Sin embargo, con la llegada de la era digital y el Big Data se ha hecho imprescindible el uso de bases de datos informatizadas.

Su origen se remonta a 1884 con Herman Hollerith, que desarrolló el tabulador electromagnético de tarjetas perforadas con el fin de ayudar en el resumen de información y posteriormente a la contabilidad.

El acceso mismo a la base de datos no la realiza el usuario en sí, sino que es el servicio de información el que efectúa las transacciones comunicándose con el DBMS, y luego entrega al usuario los resultados en forma ordenada.

Acceso a Datos mediante api

API significa "interfaz de programación de aplicaciones", en pocas palabras la API es la encargada de la comunicación de datos entre aplicaciones ya sea desde una aplicación web a móvil o viceversa.

Las APIs Rest son las más utilizadas ya que estas definen un conjunto de métodos como GET, POST, PUT Y DELETE que se refiere a la lectura, actualización y eliminación de los datos de nuestro cliente al servidor.

Lo primero a realizar será descargar la librería http por ejemplo de Flutter que nos ayudará a consumir los datos de la API. Para hacer uso de esta API tendrás que registrarte y luego suscribirte a esa API:

Una vez suscrito te brindarán un Token con el que se podrá consumir los datos libremente desde la aplicación, 'X-RapidAPIK-Key' este parámetro contendrá nuestro token.

Para realizar la consulta a los endpoint hacemos uso de Postman ya que nos ayuda mucho en la visualización de los datos mostrados por la API antes de realizar cambios en nuestra aplicación necesitamos habilitar los permisos de conexión a internet para android, para esto nos iremos al manifest.

Lo siguiente será construir la clase que consumirá los endpoint, para esta clase crearemos variables que contendrán la URL, el host y el token, esto es una buena práctica, ya que si nuestra URL base o token se llegaran a cambiar no será necesario modificar cada porción de código en la parte en que se agregó.

Para mostrar la información del provider necesitamos utilizar Provider.of<>(), este es utilizado para obtener los datos de nuestra dependencia, en nuestro ejemplo sería el método getCarsData.

Y finalmente utilizaremos el widget de Consumer para reconstruir el widget List Cars cada vez que se refleje un cambio en el provider CarProvider. En el Consumer debemos definir el Provider que queremos escuchar.

PodCast

Link Ibox:

Webgrafía

Data lake

- https://keepcoding.io/blog/data-lakes/

Diferencia entre datos estructurados y no estructurados

- https://ayudaleyprotecciondatos.es/bases-de-datos/diferencias-entre-datos-estructurados-y-no-estructurados/#:~:text=Los%20datos%20estructurados%20est%C3%A1n%20altamente,de%20recopilar%2C%20procesar%20y%20analizar.

Hadoop

- https://openwebinars.net/blog/que-es-hadoop/
- https://www.sas.com/es_es/insights/big-data/hadoop.html#:~:text=Hadoop%20es%20una%20estructura%20de,o%20trabajos%20concurrentes%20virtualmente%20ilimitados

Apache Spark

- https://aprenderbigdata.com/introduccion-apache-spark/

Acceso a datos mediante base de datos

- http://www.parada.cl/memoria/doc_3_4.html#:~:text=El%20acceso%20mismo%20a%20la,los%20resultados%20en%20forma%20ordenada.
- https://ayudaleyprotecciondatos.es/bases-de-datos/
- https://click-it.es/breve-historia-del-nacimiento-de-las-bases-de-datos/#:~:text=Pero%20si%20hablamos%20de%20bases,y%20posteriormente%20a%20la%20contabilidad.