# Componente Apache Spark.

# En esta práctica se trabajará con la herramienta Apache Spark.

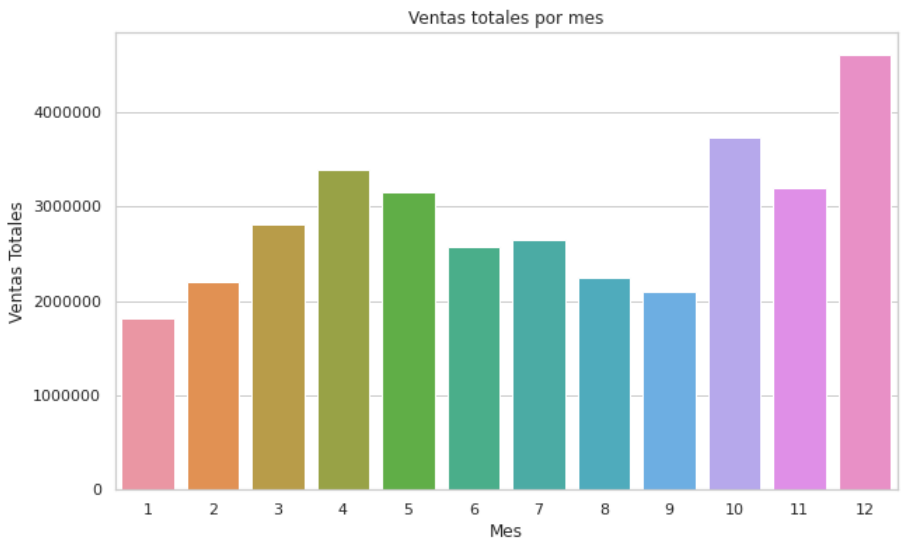
**Se pide:**

- Crear un data frame con todas las tablas y sus datos de la práctica anterior (**table_data_frame**).
- Traduce las consultas SQL que hiciste en la anterior práctica al lenguaje nativo que usa Apache Spark para realizar las mismas operaciones (**native_queries**).
- Realiza las mismas consultas utilizando el lenguage SQL dentro de Apache Spark (**sql_queries**).
- A partir del fichero nombres.json, crea un DataFrame y realiza las siguientes operaciones (**names**):
  - Crea una nueva columna (columna FaltanJubilacion) que calcule cuantos años le faltan para jubilarse (supongamos que se jubila a los 67 años)
  - Crea una nueva columna (columna Apellidos) que contenga XYZ (utilizar la función lit)
  - Elimina las columna Mayor30 y Apellidos.
  - Crea una nueva columna (columna AnyoNac) con el año de nacimiento de cada persona.
  - Añade un id incremental para cada fila (campo Id) y haz que al hacer un show se vea en primer lugar (puedes utilizar la función monotonically_increasing_id) seguidos del Nombre, Edad, AnyoNac, FaltaJubilacion y Ciudad
- A partir del fichero VentasNulos.csv (**sales**):
  - Elimina las filas que tengan al menos 4 nulos.
  - Sustituir los nombres nulos por Empleado
  - Sustituir las ventas nulas por la media de las ventas de los compañeros (redondeado a entero).
  - Los euros nulos por el valor del compañero que menos € ha ganado.
  - La ciudad nula por C.V. y el identificador nulo por XYZ.
- Utilizar UDF para lo que desees (aplica este concepto dentro de Apache Spark como tal y dentro de Spark SQL) (**user_defined_function**)
- Utilizar el dataframe de pandas para visualizar datos estadísticos (los que tu consideres) (**pandas**)
- Utilizando los ficheros movies.tsv y movie-ratings.tsv (**movies**):
  - Crea un DataFrame que contenga los datos de ambos datasets (usando algún tipo de join).
  - Persiste el DataFrame y comprueba que aparece en el Spark UI. ¿Cuánto ocupa?
  - Muestra para cada año, la película con mayor puntuación (año, título de la película, puntuación)
  - Sobre los datos anteriores, obtén también una lista con los nombres de los intérpretes.
  - Averigua las tres parejas de intérpretes han trabajado juntos en más ocasiones. La salida debe tener tres columnas: interprete1, interprete2 y cantidad.

- Tenemos un dataset (en un conjunto de ficheros CSV comprimidos en salesdata.zip) con las ventas de 2019 de una tienda de tecnología (**tech_sales**).
  - Una vez descomprimidos los datos, crea un DataFrame con todos los datos, infiriendo el esquema.
  - Vuelve a realizar la lectura de los datos pero con el siguiente esquema:

```
from pyspark.sql.types import StructType, StructField, StringType, IntegerType, DoubleType
esquema = StructType([
    StructField("Order ID", IntegerType(), False),
    StructField("Product", StringType(), False),
    StructField("Quantity Ordered", IntegerType(), True),
    StructField("Price Each", DoubleType(), False),
    StructField("Order Date", StringType(), False),
    StructField("Purchase Address", StringType(), False)
])
```

  - Tras la lectura, vamos a realizar la limpieza de datos. El primer paso será renombrar la columnas para eliminar los espacios en blanco.
  - Elimina las filas que contengan algún campo nulo.
  - Comprueba si las cabeceras de los archivos aparecen como datos del dataset (por ejemplo, un producto cuyo nombre sea Product). Si fuera el caso, elimina dichas filas.
  - A partir del campo dirección, crea dos nuevas columnas para almacenar la ciudad (City) y el estado (State). Por ejemplo, para la dirección 136 Church St, New York City, NY 10001, la ciudad es New York City y el estado es NY.
  - Modifica el campo con la fecha del pedido para que su formato sea timestamp.
  - Sobre el campo anterior, crea dos nuevas columnas, con el mes (Month) y el año (Year) del pedido.
- Una vez realizada la transformación de los datos anterior, vamos a realizar su carga y extraer información, utilizando Spark SQL siempre que sea posible (**load_and_extraction_sql**):
  - Almacena los datos en formato Parquet en la carpeta salesoutput particionando los datos por año y mes. Tras ejecutar esta operación, comprueba en disco la estructura de archivos creada.
  - Sobre los datos almacenados, realiza una nueva lectura pero solo leyendo los datos de 2019 los cuales deberían estar almacenados en ./salesdataoutput/Year=2019.
  - Averigua cual ha sido el mes que ha recaudado más. Para ello, deberás multiplicar el precio por la cantidad de unidades, y posteriormente, realizar alguna agregación. Sobre el resultado, crea un gráfico similar al siguiente:



- Obtén un gráfico con las 10 ciudades que más unidades han vendido.
- Cantidad de pedidos por Horas en las que se ha realizado un pedido que contenía al menos dos productos:
- Listado con los productos del estado de NY que se han comprado a la vez

**Material de apoyo:**

- https://aitor-medrano.github.io/iabd/spark/spark.html

**A entregar:**

- Los scripts correspondientes de Apache Spark con el nombre que está entre parentesis en negrita en cada apartado.
- Un fichero de TEXTO PLANO con los comandos a ejecutar los scripts de Spark.
- Todo lo anterior se debe de comprimir en un fichero (.zip) llamado IAB_BIU_Tema_03_Practica_05_02_Nombre_Apellidos (sin tildes, ni ñ ni espacios). Si el grupo lo forma más de una persona poner:
    - IAB_BIU_Tema_03_Practica_05_02_Nombre1_Apellidos1_Nombre2_Apellidos2