# Práctica 4: Programación en Apache Spark

Se propone la realización de 4 scripts. Los scripts deben realizarse usando Notebooks o bien como scripts que se puedan enviar con `spark-submit`.

## Normas:

- Los scripts deben incluir comentarios que expliquen los pasos realizados.
- La salida de los scripts debe seguir el formato indicado en cada uno de los ejercicios (incluyendo el nombre y orden de las columnas).
- Se debe entregar un fichero comprimido con los scripts de la práctica debidamente comentados.

# Ejercicio 1

Extraer información de los ficheros `cite75_99.txt` y `apat63_99.txt`. Crear un script que haga lo siguiente:

a. A partir del fichero `cite75_99.txt` obtener el número de citas que ha recibido cada patente. Debes obtener un DataFrame de la siguiente forma en el fichero `dfCitas.parquet`:

| NPatente | ncitas |
|---|---|
| 4943137 | 1 |
| 2959285 | 1 |
| 3004604 | 1 |
| 5060509 | 1 |
| 5708825 | 1 |
| 4549461 | 1 |
| 4756599 | 1 |

b. A partir del fichero `apat63_99.txt`, crear un DataFrame que contenga el número de patente, el país y el año de concesión (columna `GYEAR`), descartando el resto de campos del fichero. Ese DataFrame debe tener la siguiente forma, y estar en el fichero `dfInfo.parquet`:

| NPatente | País | Año |
|---|---|---|
| 4101646 | JP | 1978 |
| 4186332 | US | 1980 |
| 4920512 | JP | 1990 |
| 3512825 | ET | 1970 |
| 3797992 | US | 1974 |
| 5394547 | US | 1995 |
| 4299230 | JP | 1981 |
| 4348596 | US | 1982 |

## Requisitos

- Ambos DataFrames se debe salvar en formato Parquet con compresión gzip. Comprueba el número de particiones de cada DataFrame y el número de ficheros generados.
- El script debe aceptar argumentos en línea de comandos, es decir, para su ejecución se debe poder indicar la ruta a los ficheros de entrada y el nombre de los directorios de salida. Por ejemplo, para la ejecución en local:

```
spark-submit --master 'local[*]' --num-executors 4 --driver-memory 4g p1.py path_a_cite75_99.txt
path_a_apat63_99.txt dfCitas.parquet dfInfo.parquet
```

## Ejemplo:

```
#! /usr/bin/env python3
from pyspark.sql import SparkSession
import sys
#
# Script para extraer información de los ficheros cite75_99.txt y apat63_99.txt.
# a) A partir del fichero cite75_99.txt obtener el número de citas de cada patente.
#    Debes obtener un DataFrame de la siguiente forma:
#   +--------+------+
```

```
#   |NPatente|ncitas|
#   +--------+------+
#   | 3060453|  3   |
#   | 3390168|  6   |
#   | 3626542| 18   |
#   | 3611507|  5   |
#   | 3000113|  4   |
#
# b) A partir del fichero apat63_99.txt, crear un DataFrame que contenga el número de patente,
# el país y el año, descartando el resto de campos del fichero.
# Ese DataFrame debe tener la siguiente forma:
#
#   +--------+----+----+
#   |NPatente|País|Año |
#   +--------+----+----+
#   | 3070801| BE| 1963|
#   | 3070802| US| 1963|
#   | 3070803| US| 1963|
#   | 3070804| US| 1963|
#   | 3070805| US| 1963|
#
# Ejecutar en local con:
# spark-submit --master 'local[*]' --driver-memory 4g p1.py path_a_cite75_99.txt
path_a_apat63_99.txt dfCitas.parquet dfInfo.parquet
# Ejecución en un cluster YARN:
# spark-submit --master yarn --num-executors 8 --driver-memory 4g p1.py
path_a_cite75_99.txt_en_HDFS path_a_apat63_99.txt_en_HDFS dfCitas.parquet dfInfo.parquet

def main():
    # Comprueba el número de argumentos
    # sys.argv[1] es el primer argumento, sys.argv[2] el segundo, etc.
    if len(sys.argv) != 5:
        print(f"Uso: {sys.argv[0]} cite75_99.txt apat63_99.txt dfCitas.parquet dfInfo.parquet")
        exit(-1)

    spark: SparkSession = SparkSession\
        .builder\
        .appName("Practica 1 de Tomás")\
        .getOrCreate()

    # Cambio la verbosidad para reducir el número de
    # mensajes por pantalla
    spark.sparkContext.setLogLevel("FATAL")
    # Código del programa
    ...

if __name__ == "__main__":
    main()
```

**Nota:** Para hacer pruebas más rápidamente podéis hacer un sampleo de los ficheros grandes y trabajar con una versión más reducida.

## Ayuda en la realización del ejercicio:

- Comprueba que tienes acceso a los ficheros
- Cargamos datos en dataframes

En el shell de Unix (aquí se muestra para `patentes-mini`):

```
# Descarga de los ficheros
wget -qq https://github.com/dsevilla/tcdm-public/raw/refs/heads/24-25/datos/patentes-mini.tar.gz

# Descomprimimos
tar xzf patentes-mini.tar.gz

# Listamos ficheros
ls -lh patentes-mini/cite75_99.txt
head patentes-mini/cite75_99.txt
ls -lh patentes-mini/apat63_99.txt
head patentes-mini/apat63_99.txt
```

En el shell de Unix (aquí se muestra para `patentes`):

```
# Descarga de los ficheros
wget -qq https://github.com/dsevilla/tcdm-public/raw/refs/heads/24-25/datos/patentes.7z

# Descomprimimos
7zr x patentes.7z
```

```
# Listamos ficheros
ls -lh patentes/cite75_99.txt
head patentes/cite75_99.txt
ls -lh patentes/apat63_99.txt
head patentes/apat63_99.txt
```

Para cargar los datos de los ficheros `cite75_99.txt` y `apat63_99.txt` podéis usar el siguiente código:

```
def load_data(spark: SparkSession, path_cite: str, path_apat: str) -> tuple[DataFrame, DataFrame]:
    cites: DataFrame = (spark
        .read
        .option("inferSchema", "true")
        .option("header", "true")
        .csv(path_cite))
    cites.printSchema()
    cites.show()
    print(cites.count())
    apat: DataFrame = (spark
        .read
        .option("inferSchema", "true")
        .option("header", "true")
        .csv(path_apat))
    apat.printSchema()
    apat.show()
    print(apat.count())
    return cites, apat
```

# Ejercicio 2

Script que, a partir de los datos en Parquet de la práctica anterior, obtenga para cada país y para cada año el total de patentes, el total de citas obtenidas por todas las patentes, la media de citas y el máximo número de citas.

- Obtener solo aquellos casos en los que existan valores en ambos ficheros (inner join).
- Cada país tiene que aparecer con su nombre completo, obtenido del fichero `country_codes.txt`, residente en el disco local
- El DataFrame generado debe estar ordenado por el máximo número de citas, país y año.

Ejemplo de salida:

| País | Año | NumPatentes | TotalCitas | MediaCitas | MaxCitas |
|---|---|---|---|---|---|
| Japan | 1975 | 15 | 17 | 1.1333333333333333 | 3 |
| United States | 1982 | 129 | 131 | 1.0155038759689923 | 3 |
| Germany | 1993 | 10 | 11 | 1.1 | 2 |
| Hungary | 1970 | 2 | 3 | 1.5 | 2 |
| Japan | 1983 | 29 | 30 | 1.0344827586206897 | 2 |
| Japan | 1984 | 44 | 45 | 1.0227272727272727 | 2 |
| Japan | 1986 | 51 | 52 | 1.0196078431372548 | 2 |
| Sweden | 1972 | 2 | 3 | 1.5 | 2 |
| United States | 1963 | 58 | 59 | 1.0172413793103448 | 2 |
| United States | 1965 | 79 | 80 | 1.0126582278481013 | 2 |
| United States | 1966 | 94 | 95 | 1.0106382978723405 | 2 |
| United States | 1967 | 122 | 123 | 1.0081967213114753 | 2 |
| United States | 1973 | 142 | 144 | 1.0140845070422535 | 2 |
| United States | 1974 | 168 | 169 | 1.005952380952381 | 2 |

## Requisitos

- El DataFrame obtenido se debe guardar en un único fichero CSV sin comprimir y con cabecera.

**Ayuda en la realización del ejercicio:**

- Lectura de fichero parquet.
- Cargamos el fichero con los códigos del país en un diccionario.
- Guardar un DataFrame en un único fichero CSV sin comprimir y con cabecera.

```
from pyspark import Broadcast
# Leo el fichero de citas
```

```
dfCitas: DataFrame = (spark.read.format("parquet")
               .option("mode", "FAILFAST")
               .load("patentes-mini/dfCitas.parquet"))

# Cargamos el fichero con los códigos del pais
ccDict = dict()
with open("patentes-mini/country_codes.txt") as ccfile:
    for row in ccfile.readlines():
        code, country = row.split('\t')
        ccDict[code] = country.strip()
bcastCCDict: Broadcast[dict[str,str]] = spark.sparkContext.broadcast(ccDict)

# Lo guardamos como un único fichero CSV
(dfCitas.coalesce(1)
       .write.format("csv")
       .mode("overwrite")
       .option("header", True)
       .save("patentes-mini/p2"))
```

## Ejercicio 3

Obtener a partir de los ficheros Parquet creados en el ejercicio 1 un DataFrame que proporcione, para un grupo de países especificado, las patentes ordenadas por número de citas, de mayor a menor, junto con una columna que indique el ango (posición de la patente en esa país/año según las citas obtenidas):

La salida del script debe ser como sigue:

| País | Año | Npatente | Ncitas | Rango |
|---|---|---|---|---|
| ES | 1963 | 3093080 | 20 | 1 |
| ES | 1963 | 3099309 | 19 | 2 |
| ES | 1963 | 3081560 | 9 | 3 |
| ES | 1963 | 3071439 | 9 | 3 |
| ES | 1963 | 3074559 | 6 | 4 |
| ES | 1963 | 3114233 | 5 | 5 |
| ES | 1963 | 3094845 | 4 | 6 |
| ES | 1963 | 3106762 | 3 | 7 |
| ES | 1963 | 3088009 | 3 | 7 |
| ES | 1963 | 3087842 | 2 | 8 |
| ES | 1963 | 3078145 | 2 | 8 |
| ES | 1963 | 3094806 | 2 | 8 |
| ES | 1963 | 3073124 | 2 | 8 |
| ES | 1963 | 3112201 | 2 | 8 |
| ES | 1963 | 3102971 | 1 | 9 |
| ES | 1963 | 3112703 | 1 | 9 |
| ES | 1963 | 3095297 | 1 | 9 |
| ES | 1964 | 3129307 | 11 | 1 |
| ES | 1964 | 3133001 | 10 | 2 |
| ES | 1964 | 3161239 | 8 | 3 |
| ... | ... | ... | ... | ... |
| FR | 1963 | 3111006 | 35 | 1 |
| FR | 1963 | 3083101 | 22 | 2 |
| FR | 1963 | 3077496 | 16 | 3 |
| FR | 1963 | 3072512 | 15 | 4 |
| FR | 1963 | 3090203 | 15 | 4 |
| FR | 1963 | 3086777 | 14 | 5 |
| FR | 1963 | 3074344 | 13 | 6 |
| FR | 1963 | 3096621 | 13 | 6 |
| FR | 1963 | 3089153 | 13 | 6 |
| ... | ... | ... | ... | ... |

## Requisitos

- El DataFrame debe de estar ordenado por código del país y año (ascendente) y número de citas (descendente).
- Utilizad funciones de ventana para obtener el rango.
- NO reemplazar el código del país por su nombre completo.
- La salida debe guardarse en un único fichero CSV sin comprimir y con cabecera.
- Como en los casos anteriores, el script debe aceptar argumentos en línea de comandos, es decir, para su ejecución deberíamos poder indicar la ruta a los directorios de entrada creados en la práctica 1, la lista de países a analizar (separados por coma) y el nombre del directorio de salida.

# Ejercicio 4

Obtener a partir del fichero Parquet con la información de (Npatente, País y Año) un DataFrame que nos muestre el número de patentes asociadas a cada país por cada década (entendemos por década los años del 0 al 9, es decir de 1970 a 1979 es la década de los 70s). Adicionalmente, debe mostrar el aumento o disminución del número de patentes para cada país y década con respecto al la década anterior.

El DataFrame generado tiene que ser como este:

| País | Década | NPatentes | Dif |
|---|---|---|---|
| AD | 1980 | 1 | 0 |
| AD | 1990 | 5 | 4 |
| AE | 1980 | 7 | 0 |
| AE | 1990 | 11 | 4 |
| AG | 1970 | 2 | 0 |
| AG | 1990 | 7 | 5 |
| AI | 1990 | 1 | 0 |
| AL | 1990 | 1 | 0 |
| AM | 1990 | 2 | 0 |
| AN | 1970 | 1 | 0 |
| AN | 1980 | 2 | 1 |
| AN | 1990 | 5 | 3 |
| AR | 1960 | 135 | 0 |
| AR | 1970 | 239 | 104 |
| AR | 1980 | 184 | -55 |
| AR | 1990 | 292 | 108 |
| ... | ... | ... | ... |

## Requisitos

- El DataFrame debe de estar ordenado por código del país y año.
- NO reemplazar el código del país por su nombre completo.
- La salida debe guardarse en un único fichero CSV sin comprimir y con cabecera.
- Como en los casos anteriores, el script debe aceptar argumentos en línea de comandos, es decir, para su ejecución deberíamos poder indicar la ruta al directorio de entrada creado en la práctica 1 y el nombre del directorio de salida.