# Tecnoloxías de computación para datos masivos [P4181103] [2021/2022]



## Tema 5 - Práctica: Descripción y entrega

**Pendente:** Luns, 17 de Xaneiro de 2022, 23:59

---

### Actividad: Programación en Apache PySpark

Se propone la realización de 5 scripts. Para aprobar esta práctica es necesario, como mínimo, completar los 3 primeros.

Los script deben ejecutarse en el CESGA utilizando spark-submit. Para el desarrollo, podéis utilizar el contenedor Zeppelin en vuestro PC o, si preferís, un notebook Jupyter que se ejecute en el CESGA.
Para ello es necesario que inicies la VPN del CESGA y que os conectéis por ssh a hadoop3.usc.es. Una vez dentro ejecutais los siguientes comandos:

```
module load anaconda3
PYSPARK_DRIVER_PYTHON=jupyter PYSPARK_DRIVER_PYTHON_OPTS="notebook --no-browser --ip=IP_del_nodo" pyspark &
```

donde la IP_del_nodo es la IP de la máquina a la que estáis conectador, que podéis obtener ejecutando ifconfig.
Una vez iniciado, os conectáis a la URL que se indica y ya os abre un Jupyter con el que podéis trabajar. Todas las tareas que le lanzen en ese notebook se ejecutarán en el cluster del CESGA.

NOTA: A través del portal https://bigdata.cesga.es/ (con la VPN activada) podéis ver como avanzan vuestras aplicaciones en YARN.

**Normas**:

- Al inicio del script debe incluirse comentarios en los que se indiquen claramente como hay que ejecutarlo:
  - Seguid las indicaciones de esta guía y el formato indicado en el ejemplo de la práctica 1.
- La salida de los scripts debe seguir el formato indicado en cada una de las prácticas (**incluyendo el nombre y orden de las columnas**).
- Entregar un fichero comprimido con todos los scripts Python debidamente comentados.

#### Parte obligatoria (75% de la nota de esta práctica)

1. (10% de la práctica) 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:

```
+--------+------+
|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 de concesión (columna GYEAR), descartando el resto de campos del fichero. Ese DataFrame debe tener la siguiente forma:

```
+--------+----+----+
|NPatente|Pais|Anho|
+--------+----+----+
| 3070801| BE| 1963|
| 3070802| US| 1963|
| 3070803| US| 1963|
| 3070804| US| 1963|
| 3070805| US| 1963|
```

**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
```

  Y para ejecutarlo en el CESGA:

```
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
```

- Ejemplo:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from __future__ import print_function,division
from pyspark.sql import SparkSession
import pyspark.sql.functions as F
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|Pais|Anho|
#   +--------+----+----+
#   | 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("Usar: p1.py cite75_99.txt apat63_99.txt dfCitas.parquet dfInfo.parquet")
        exit(-1)

    spark = SparkSession\
        .builder\
        .appName("Practica 1 de Tomas")\
        .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.
**Nota2**: Si en el CESGA tarda mucho en entrar el trabajo, podéis lanzarlo a la cola **urgent** especificándola con la opción --queue

```
spark-submit --master yarn --num-executors 8 --driver-memory 4g --queue urgent p1.py path_a_cite75_99.txt_en_HDFS
path_a_apat63_99.txt_en_HDFS dfCitas.parquet dfInfo.parquet
```

2. (35% de la práctica) 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 país y, para cada país, por año

   Ejemplo de salida:

```
+------------------+----+----------+---------+------------------+--------+
|Pais              |Anho|NumPatentes|TotalCitas|MediaCitas        |MaxCitas|
+------------------+----+----------+---------+------------------+--------+
|Algeria           |1963|2         |7        |3.5               |4       |
|Algeria           |1968|1         |2        |2.0               |2       |
|Algeria           |1970|1         |2        |2.0               |2       |
|Algeria           |1972|1         |1        |1.0               |1       |
|Algeria           |1977|1         |2        |2.0               |2       |
|Andorra           |1987|1         |3        |3.0               |3       |
|Andorra           |1993|1         |1        |1.0               |1       |
|Andorra           |1998|1         |1        |1.0               |1       |
|Antigua and Barbuda|1978|1         |6        |6.0               |6       |
|Antigua and Barbuda|1979|1         |14       |14.0              |14      |
|Antigua and Barbuda|1991|1         |8        |8.0               |8       |
|Antigua and Barbuda|1994|1         |19       |19.0              |19      |
|Antigua and Barbuda|1995|2         |12       |6.0               |11      |
|Antigua and Barbuda|1996|2         |3        |1.5               |2       |
|Argentina         |1963|14        |35       |2.5               |7       |
|Argentina         |1964|20        |60       |3.0               |8       |
|Argentina         |1965|10        |35       |3.5               |10      |
|Argentina         |1966|16        |44       |2.75              |9       |
|Argentina         |1967|13        |60       |4.615384615384615 |14      |
```

   **Requisitos**
   - El DataFrame obtenido se debe guardar **en un único fichero CSV sin comprimir y con cabecera**.
   - IMPORTANTE: El fichero country_codes.txt **no** debe residir en HDFS, sino en el disco local. Debemos usarlo como **una variable de broadcast**, siguiendo el ejemplo de los apuntes.
   - El script deber aceptar argumentos en línea de comandos, es decir, para su ejecución deberíamos poder indicar la ruta a los directorios de entrada en HDFS, la ruta al al fichero country_codes.txt (en el disco local) y el nombre del directorio de salida. Por ejemplo, para la ejecución en el CESGA

```
spark-submit --master yarn --num-executors 8 --driver-memory 4g --queue urgent p2.py dirNcitas dirInfo
path_a_country_codes_en_local.txt p2out
```

3. (30% de la práctica) A partir del fichero apat63_99.txt obtener el número de patentes por país y año **usando RDDs** (no uséis DataFrames).
   - El RDD obtenido debe ser un RDD clave/valor, en el cual la clave es un país y el valor una lista de tuplas, en la que cada tupla esté formada por un año y el número de patentes de ese país en ese año.
   - Adicionalmente, el RDD creado debe estar ordenado por el código del país y, para cada país, la lista de valores ordenados por año.

   Ejemplo de par clave/valor de salida

```
(u'PA', [(u'1963', 2), (u'1964', 2), (u'1965', 1), (u'1966', 1), (u'1970', 1), (u'1971', 1), (u'1972', 6), (u'1974', 3),
(u'1975', 5), (u'1976', 3), (u'1977', 2), (u'1978', 2), (u'1980', 2), (u'1982', 1), (u'1983', 1), (u'1985', 2), (u'1986',
1), (u'1987', 2), (u'1988', 1), (u'1990', 1), (u'1991', 2), (u'1993', 1), (u'1995', 1), (u'1996', 1), (u'1999', 1)])
```

   **Requisitos**
   - Usad 8 particiones al leer el fichero apat63_99.txt.
   - Deben eliminarse las comillas dobles del código del país.
   - El script debe aceptar argumentos en línea de comandos, es decir, para su ejecución deberíamos poder indicar la ruta al fichero de entrada (apat63_99.txt) y el nombre del directorio de salida.
   - La salida debe estar en formato texto sin comprimir.

## Parte opcional (25% de la nota de esta práctica)

4. (15% de la práctica) Obtener a partir de los fichero Parquet creados en la practica 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 rango (posición de la patente en esa país/año según las citas obtenidas

La ejecución y salida del script debe ser como sigue

```
$ spark-submit ... p4.py dfCitas.parquet dfInfo.parquet FR,ES outdir
....
+----+----+--------+------+-----+
|Pais|Anho|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.

5. (10% de la práctica) Obtener a partir del fichero Parquet con la información de (Npatente, Pais 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 una década). 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:

```
+----+------+---------+----+
|Pais|Decada|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.

## Estado da entrega

| | |
|---|---|
| **Estado da entrega** | Sen intentos |
| **Estado das cualificacións** | Sen cualificar |
| **Tempo restante** | 18 días 6 horas |
| **Última modificación** | - |
| **Comentarios a entrega** | ▸ Comentarios (0) |

Engadir entrega

Vostede aínda non fixo ningunha entrega

◄ Grabación clase 15/11/21

Ir a...

English (en)
English (ja)
English (United States) (en_us)
Español - Internacional (es)
Euskara (eu)
Français (fr)
Galego (gl)
Italiano (it)
Nederlands (nl)
Norsk (no)
Polski (pl)
Português - Portugal (pt)
Română (ro)
Slovenščina (sl)
Svenska (sv)
Türkçe (tr)
Тоҷикӣ (tg)
Українська (uk)
עברית (he)
日本語 (ja_old)
正體中文 (zh_tw)
简体中文 (zh_cn)