(https://databricks.com)

Práctica Big Data Processing con scala en notebook de Databricks

ZÚÑIGA ASPAS, Alba

```
//Los .csv los he subido a DBFS
//Cargamos los dataset
val df1 = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/alzuaskeepcoding@gmail.com/wo
val df2 = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/alzuaskeepcoding@gmail.com/wo

df1.show(5)
df2.show(5)
```

```
+------------+------------------+------------+-----------------------------+------------+------------+--------------------
-+--------------+-----------------------+---------------------------+----------+-------------------------+----------------
--------+--------------------------------+---------------------------+-----------------------------------+--------------
-------------------------+-----------------------+-----------------------+-----------------------------------+------------------+
|Country name|Regional indicator|Ladder score|Standard error of ladder score|upperwhisker|lowerwhisker|Logged GDP per capit
a|Social support|Healthy life expectancy|Freedom to make life choices|Generosity|Perceptions of corruption|Ladder score in
Dystopia|Explained by: Log GDP per capita|Explained by: Social support|Explained by: Healthy life expectancy|Explained by:
Freedom to make life choices|Explained by: Generosity|Explained by: Perceptions of corruption|Dystopia + residual|
+------------+------------------+------------+-----------------------------+------------+------------+--------------------
-+--------------+-----------------------+---------------------------+----------+-------------------------+----------------
--------+--------------------------------+---------------------------+-----------------------------------+--------------
-------------------------+-----------------------+-----------------------+-----------------------------------+------------------+
|     Finland|    Western Europe|       7.842|                        0.032|       7.904|       7.780|               10.77
5|         0.954|                 72.000|                      0.949|    -0.098|                    0.186|
2.430|                          1.446|                      1.106|                                 0.741|
0.691|                   0.124|                                 0.481|              3.253|
|     Denmark|    Western Europe|       7.620|                        0.035|       7.687|       7.552|               10.93
3|         0.954|                 72.700|                      0.946|     0.030|                    0.179|
2.430|                          1.502|                      1.108|                                 0.763|
0.686|                   0.208|                                 0.485|              2.868|
| Switzerland|    Western Europe|       7.571|                        0.036|       7.643|       7.500|               11.11
```

```
//1. ¿Cuál es el país más “feliz” del 2021 según la data?
import org.apache.spark.sql.functions._

val happiestCountry2021 = df1.orderBy(desc("Ladder score")).select("Country name", "Ladder score").first()
println(s"El país más feliz de 2021 es ${happiestCountry2021(0)} con un puntaje de ${happiestCountry2021(1)}")
```

```
El país más feliz de 2021 es Finland con un puntaje de 7.842
import org.apache.spark.sql.functions._
happiestCountry2021: org.apache.spark.sql.Row = [Finland,7.842]
```

```
//2. ¿Cuál es el país más “feliz” del 2021 por continente según la data?
import org.apache.spark.sql.expressions.Window

// Agregamos la columna "Max Ladder score" a df1
val windowSpec = Window.partitionBy("Regional indicator").orderBy(desc("Ladder score"))
val df1WithRank = df1.withColumn("rank", rank().over(windowSpec))

// Filtramos los países que tienen el puntaje más alto por continente
val happiestCountryByContinent2021 = df1WithRank.filter($"rank" === 1)
  .select("Regional indicator", "Country name", "Ladder score")

happiestCountryByContinent2021.show()
```

```
+--------------------+--------------------+------------+
|  Regional indicator|        Country name|Ladder score|
+--------------------+--------------------+------------+
|Central and Easte...|      Czech Republic|       6.965|
|Commonwealth of I...|          Uzbekistan|       6.179|
|           East Asia|Taiwan Province o...|       6.584|
|Latin America and...|          Costa Rica|       7.069|
|Middle East and N...|              Israel|       7.157|
|North America and...|         New Zealand|       7.277|
|          South Asia|               Nepal|       5.269|
|      Southeast Asia|           Singapore|       6.377|
|  Sub-Saharan Africa|           Mauritius|       6.049|
|      Western Europe|             Finland|       7.842|
+--------------------+--------------------+------------+

import org.apache.spark.sql.expressions.Window
windowSpec: org.apache.spark.sql.expressions.WindowSpec = org.apache.spark.sql.expressions.WindowSpec@39e5402d
df1WithRank: org.apache.spark.sql.DataFrame = [Country name: string, Regional indicator: string ... 19 more fields]
happiestCountryByContinent2021: org.apache.spark.sql.DataFrame = [Regional indicator: string, Country name: string ... 1 more
field]
```

```
//3. ¿Cuál es el país que más veces ocupó el primer lugar en todos los años?
import org.apache.spark.sql.expressions.Window

val windowSpec = Window.partitionBy("year").orderBy(desc("Life Ladder"))
val rankDf = df2.withColumn("rank", rank().over(windowSpec))
val firstPlaceCount = rankDf.filter($"rank" === 1).groupBy("Country name").count().orderBy(desc("count"))

firstPlaceCount.show(1)
```

```
+------------+-----+
|Country name|count|
+------------+-----+
|     Denmark|    7|
+------------+-----+
only showing top 1 row

import org.apache.spark.sql.expressions.Window
windowSpec: org.apache.spark.sql.expressions.WindowSpec = org.apache.spark.sql.expressions.WindowSpec@72f5df2f
rankDf: org.apache.spark.sql.DataFrame = [Country name: string, year: string ... 10 more fields]
firstPlaceCount: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Country name: string, count: bigint]
```

```
//4. ¿Qué puesto de Felicidad tiene el país con mayor GDP del 2020?
val gdp2020 = df2.filter($"year" === 2020).orderBy(desc("Log GDP per capita")).select("Country name").first()
val countryWithHighestGdp2020 = gdp2020(0)

val happinessRank2020 = df2.filter($"year" === 2020).orderBy(desc("Life Ladder"))
  .withColumn("rank", rank().over(Window.partitionBy("year").orderBy(desc("Life Ladder"))))
  .filter($"Country name" === countryWithHighestGdp2020)
  .select("Country name", "rank")

happinessRank2020.show()
```

```
+------------+----+
|Country name|rank|
+------------+----+
|    Bulgaria|  56|
+------------+----+

gdp2020: org.apache.spark.sql.Row = [Bulgaria]
countryWithHighestGdp2020: Any = Bulgaria
happinessRank2020: org.apache.spark.sql.DataFrame = [Country name: string, rank: int]
```

```
//5. ¿En que porcentaje a variado a nivel mundial el GDP promedio del 2020 respecto al 2021? ¿Aumentó o disminuyó?
// Filtramos los datos para el año 2020 y 2021
val df2020 = df2.filter($"year" === 2020)
val df2021 = df1 // Datos de 2021 ya están en df1

// Calculamos el promedio de Log GDP per capita para 2020
val avgGdp2020 = df2020.agg(avg("Log GDP per capita")).first().getDouble(0)

// Calculamos el promedio de Logged GDP per capita para 2021
val avgGdp2021 = df2021.agg(avg("Logged GDP per capita")).first().getDouble(0)

// Calculamos la variación porcentual
val percentageChange = ((avgGdp2021 - avgGdp2020) / avgGdp2020) * 100

if (percentageChange > 0) {
  println(f"El GDP promedio a nivel mundial aumentó un $percentageChange%.2f%% del 2020 al 2021")
} else {
  println(f"El GDP promedio a nivel mundial disminuyó un $percentageChange%.2f%% del 2020 al 2021")
}
```

```
El GDP promedio a nivel mundial disminuyó un -3.27% del 2020 al 2021
df2020: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Country name: string, year: string ... 9 more fields]
df2021: org.apache.spark.sql.DataFrame = [Country name: string, Regional indicator: string ... 18 more fields]
avgGdp2020: Double = 9.751329545454546
avgGdp2021: Double = 9.432208053691273
percentageChange: Double = -3.2725946782511013
```

```
El país con mayor expectativa de vida en 2021 es Singapore con una expectativa de vida de 76.953 años.
En 2019, la expectativa de vida de Singapore era de 77.1 años.
df1: org.apache.spark.sql.DataFrame = [Country name: string, Regional indicator: string ... 18 more fields]
df2: org.apache.spark.sql.DataFrame = [Country name: string, year: int ... 9 more fields]
import org.apache.spark.sql.functions._
highestLifeExpectancy2021: org.apache.spark.sql.Row = [Singapore,76.953]
countryWithHighestLifeExpectancy2021: String = Singapore
highestLifeExpectancyValue2021: Double = 76.953
df2019: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Country name: string, year: int ... 9 more fields]
lifeExpectancy2019: Double = 77.1
```