

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

**MAESTRÍA EN GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LA
COMUNICACIÓN**



Big Data

Alumnos:

**Ledezma Ibarra Juan Francisco
López Ramírez Flavio Hiram**

Evaluación 1

Octubre 2023, Tijuana Baja California, México

Evaluation Unit 1 [↗](#)

1.- Comienza una simple sesión Spark. [↗](#)

```
import org.apache.spark.sql.SparkSession
val spark = SparkSession.builder().getOrCreate()
```

```
val args: Array[String] = Array()
Loading Practices\evaluationPractice1.scala...
import org.apache.spark.sql.SparkSession
val spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@6b1293a5
```

2.- Cargue el archivo Netflix Stock CSV en dataframe llamado df, haga que Spark infiera los tipos de datos. [↗](#)

```
val df = spark.read.option("header", "true").option("inferSchema", "true").csv("Practices/Netflix_2011_2016.csv")
```

```
val df: org.apache.spark.sql.DataFrame = [Date: date, Open: double ... 5 more fields]
```

3. ¿Cuáles son los nombres de las columnas? [↗](#)

```
df.printSchema()
```

Imagen 1 - Archivo Readme en Github

```
README.md x evaluationPractice1.scala df.scala test.scala
ClaseBigData > README.md > # Evaluation Unit 1
405
406 ## 1.- Comienza una simple sesión Spark.
407 ```scala
408 import org.apache.spark.sql.SparkSession
409 val spark = SparkSession.builder().getOrCreate()
410 ```
411 ```sh
412 val args: Array[String] = Array()
413 Loading Practices\evaluationPractice1.scala...
414 import org.apache.spark.sql.SparkSession
415 val spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@6b1293a5
416 ```
417 ## 2.- Cargue el archivo Netflix Stock CSV en dataframe llamado df, haga que Spark infiera los tipos de datos.
418 ```scala
419 val df = spark.read.option("header", "true").option("inferSchema", "true").csv("Practices/Netflix_2011_2016.csv")
420 ```
421 ```sh
422 val df: org.apache.spark.sql.DataFrame = [Date: date, Open: double ... 5 more fields]
423 ```
```

Imagen 2 - Archivo Fuente de Readme (Visual Code)

Commits		
unit1		
Commits on Oct 22, 2023		
11 e).- Calculate column Close avg for every month	FLI04 committed 4 minutes ago	01d9fc4 <>
11 d).- Get column High max value groupBy Year	FLI04 committed 8 minutes ago	7fcdc94 <>
11 c).- Pearson corr between columns High & Volume	FLI04 committed 13 minutes ago	28d973d <>
11 b).- Calculate time avg of column High greater than 500	Flavio-Lopez committed 17 minutes ago	05597f9 <>
11 a).- Days from column Close under 600	Flavio-Lopez committed 49 minutes ago	eb57b4f <>
10.- Select Min & Max values of column Volume	FLI04 committed 1 hour ago	7b07f76 <>
9.- Description of Stock's Close position	FLI04 committed 1 hour ago	661460f <>
Select max value from Open column	Flavio-Lopez committed 1 hour ago	1264b66 <>
7.- Create HVRatio dataframe adding HV Ratio Column with Volume and H...	Flavio-Lopez committed 2 hours ago	a2df096 <>
6.- Describe df dataframe	Flavio-Lopez committed 2 hours ago	7cf4b62 <>

Imagen 3 - Historial de Commits Generados en Github

```

evaluationPractice1.scala X df.scala test.scala
BigData > Practices > evaluationPractice1.scala
1 // 1. Comienza una simple sesión Spark
2 import org.apache.spark.sql.SparkSession
3 val spark = SparkSession.builder().getOrCreate()
4
5 // 2. Cargue el archivo Netflix Stock CSV en dataframe llamado df, haga que Spark infiera los tipos de datos.
6 val df = spark.read.option("header", "true").option("inferSchema", "true").csv("Practices/Netflix_2011_2016.csv")
7
8
9 // 3. ¿Cuáles son los nombres de las columnas?
10 df.printSchema()
11
12 // 4. ¿Cómo es el esquema?
13 df.printSchema()
14
15 // 5. Imprime las primeras 5 renglones.
16 df.head(5)
17
18 // 6. Usa el método describe () para aprender sobre el DataFrame.
19 df.describe().show()
20
21 // 7. Crea un nuevo Dataframe a partir del df creado anteriormente llamdo "HVRatio" para crear una columna nueva llamada '
22 // entre el precio de la columna "High" frente a la columna "Volumen" de acciones negociadas por un día. Hint - es una o
23
24 val HVRatio = df.withColumn("HV Ratio", $"Volume" / $"High")
25 HVRatio.show()
26
27 // 8. ¿Qué día tuvo el pico más alto en la columna "Open"?
28
29 df.select(max("OPEN")).show()
30
31 // 10. ¿Cuál es el máximo y mínimo de la columna "Volumen"?
32
33 df.select(max("Volume"),min("Volume")).show()
34
35 // 11. Con Sintaxis Scala/Spark $ conteste lo siguiente:
36 // a) ¿Cuántos días fue la columna "Close" inferior a $ 600?
37 df.filter($"Close"<600).count()
38 // b) ¿Qué porcentaje del tiempo fue la columna "High" mayor que $ 500?
39 (df.filter($"High">500).count()*1.0/df.count())*100
40
41 // c) ¿Cuál es la correlación de Pearson entre columna "High" y la columna "Volumen"?

```

Imagen 4 - Código en Scala