Instituto Tecnológico de Tijuana

Ingeniería en Tecnologías de la Información y Comunicaciones

Nombre de la Materia:

DATOS MASIVOS

Actividad:
Examen 1

Profesor:

JOSE CHRISTIAN ROMERO HERNANDEZ

Alumno(s):

Ramos Rivera Manue Isai #17212931
López Higuera Saúl Alfredo #18210493

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TIJUANA

SUBDIRECCIÓN ACADÉMICA

Departamento de Sistemas y Computación

EXAMEN

Carrera: Ingeniería En Sistemas Computacionales/ Tecnologías de la información/ Informatíca Período: **Febreo-Junio 2022** Materia: Datos Masivos Grupo: BDD-1704SC9C Salón: Unidad (es) a evaluar: Unidad 1 Tipo de examen:Practico Fecha: Catedrático: Jose Christian Romero Hernandez Firma del maestro: Calificación:

Alumnos:

Ramos Rivera Manue Isai #17212931

López Higuera Saúl Alfredo #18210493

Instrucciones

Responder las siguientes preguntas con Spark DataFrames y Scala utilizando el "CSV" Netflix_2011_2016.csv que se encuentra el la carpeta de spark-dataframes.

1. Comienza una simple sesión Spark.

Examen1.scala

Spark_DataFrame > Examen1.scala

```
1. Start a simple Spark session
import org.apache.spark.sql.SparkSession

val session = SparkSession.builder().getOrCreate

```

```
scala> import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.SparkSession

scala> val session = SparkSession.builder().getOrCreate
session: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@134ec0f3
```

2. Cargue el archivo Netflix Stock CSV, haga que Spark infiera los tipos de datos.

```
2. Upload Netflix Stock CSV file, have spark infer data types
val df_netflix = session.read.option("header", "true").option("inferSchema", true).csv("Netflix_2011_2016.csv")
```

```
scala> val df_netflix = session.read.option("header", "true").option("inferSchema", true).csv("Netflix_2011_2016.csv")
df_netflix: org.apache.spark.sql.DataFrame = [Date: timestamp, Open: double ... 5 more fields]
```

3. ¿Cuáles son los nombres de las columnas?

```
3. What are the names of the columns?
df_netflix.columns
```

```
scala> df_netflix.columns
res0: Array[String] = Array(Date, Open, High, Low, Close, Volume, Adj Close)
```

4. ¿Cómo es el esquema?

```
4. How is the scheme?
df_netflix.printSchema()
```

```
scala> df_netflix.printSchema()
root
 |-- Date: timestamp (nullable = true)
 |-- Open: double (nullable = true)
 |-- High: double (nullable = true)
 |-- Low: double (nullable = true)
 |-- Close: double (nullable = true)
 |-- Volume: integer (nullable = true)
 |-- Adj Close: double (nullable = true)
```

5. Imprime las primeras 5 columnas.

```
5. Print the first 5 columns
df_netflix.head(5)
```

```
scala> df_netflix.head(5)
res2: Array[org.apache.spark.sql.Row] = Array([2011-10-24 00:00:00.0,119.100002,120.28000300000001,115.100004,118.839996,120460200,16.977142], [2011-10-25 00:00:00.0,74.899999,79.390001,74.249997,77.370002,315541800,11.052857000000001], [2011-10-26 00:00:00.0,78.73,81.420001,75.399997,79.400002,148733900,11.342857], [2011-10-27 00:00:00.0,82.179998,82.71999699999999,79.249998,80.86000200000001,71190000,11.551428999999999], [2011-10-28 00:00:00.0,80.280002,84.660002,79.599999,84.14000300000001,57769600,12.02])
```

6. Usa describe () para aprender sobre el DataFrame.

```
6. Use describe () to learn about the DataFrame
df_netflix.describe().show
```

```
scala> df_netflix.describe().show
+-------+------------------+-----------------+-----------------+-----------------+--------------------+------------------+
|summary|              Open|             High|              Low|            Close|              Volume|         Adj Close|
+-------+------------------+-----------------+-----------------+-----------------+--------------------+------------------+
|  count|              1259|             1259|             1259|             1259|                1259|              1259|
|   mean|230.39351086656092|233.97320872915006|226.80127876251044|  230.522453845909|2.5634836060365368E7|55.610540036536875|
| stddev|164.37456353264244| 165.9705082667129| 162.6506358235739|164.40918905512854| 2.306312683388607E7|35.186669331525486|
|    min|         53.990001|        55.480001|            52.81|            53.8|             3531300|          7.685714|
|    max|        708.900017|        716.159996|        697.569984|        707.610001|           315541800|         130.929993|
+-------+------------------+-----------------+-----------------+-----------------+--------------------+------------------+
```

7. Crea un nuevo dataframe con una columna nueva llamada “HV Ratio” que es la relación que existe entre el precio de la columna “High” frente a la columna “Volumen” de acciones negociadas por un día. Hint - es una operación

```
7. Create a new data frame with a new column called "HV Ratio" which is the relationship between the price in the "High" column versus the "Volume" column of shares traded for a
val df_netflix2 = df_netflix.withColumn("HV Ratio", df_netflix("High")/df_netflix("Volume"))
```

8. ¿Qué día tuvo el pico mas alto en la columna “Open”?

```
8. What day had the highest peak in the "Open" column?
df_netflix.select(max("Open")).show()
```

9. ¿Cuál es el significado de la columna Cerrar “Close” en el contexto de información financiera, expliquelo no hay que codificar nada?

```
9. What is the meaning of the "Close" column in the context of financial information, explain it, there is no need to code anything?

//The Close column refers to the company action price at the end of the day's closing.
```

10. ¿Cuál es el máximo y mínimo de la columna “Volumen”?

```
10. What is the maximum and minimum in the "Volume" column?
df_netflix.select(max("Volume")).show()
df_netflix.select(min("Volume")).show()
```

```
scala> df_netflix.select(mean("Open")).show()
+------------------+
|         avg(Open)|
+------------------+
|230.39351086656092|
+------------------+


scala> df_netflix.select(max("Volume")).show()
+-----------+
|max(Volume)|
+-----------+
|  315541800|
+-----------+
```

11. Con Sintaxis Scala/Spark $ conteste los siguiente:

a. ¿Cuántos días fue la columna “Close” inferior a $ 600?

```
a. How many days was the “Close” column less than $ 600?
val Day = df_netflix.where($"Close" < 600).count()
```

```
scala> val Day = df_netflix.where($"Close" < 600).count()
Day: Long = 1218
```

b. ¿Qué porcentaje del tiempo fue la columna “High” mayor que $ 500?

```
b. What percentage of the time was the “High” column greater than $ 500?
val Day = df_netflix.where($"High" > 500).count().toFloat
```

```
scala> val Day = df_netflix.where($"High" > 500).count().toFloat
Day: Float = 62.0
```

c. ¿Cuál es la correlación de Pearson entre la columna “High” y la columna “Volumen”?

```
c. What is the Pearson correlation between the “High” column and the “Volume” column?
df_netflix.select(corr("High", "Volume")).show()
```

```
scala> df_netflix.select(corr("High", "Volume")).show()
+--------------------+
|  corr(High, Volume)|
+--------------------+
|-0.20960233287942157|
+--------------------+
```

d. ¿Cuál es el máximo de la columna “High” por año?

```
d. What is the maximum in the “High” column per year?
df_netflix.groupBy(year($"Date")).max("High").show()
```

```
scala> df_netflix.groupBy(year($"Date")).max("High").show()
+----------+------------------+
|year(Date)|         max(High)|
+----------+------------------+
|      2015|        716.159996|
|      2013|        389.159988|
|      2014|        489.290024|
|      2012|        133.429996|
|      2016|129.28999299999998|
|      2011|120.28000300000001|
+----------+------------------+
```

e. ¿Cuál es el promedio de columna “Close” para cada mes del calendario?

```
e. What is the average in the “Close” column for each calendar month?
val df_netflix3 = df_netflix.groupBy(year($"Date"), month($"Date")).mean("Close"). toDF("Year","Month","Mean")
df_netflix3.orderBy($"Year",$"Month").show()
```

```
scala> df_netflix3.orderBy($"Year",$"Month").show()
+----+-----+------------------+
|Year|Month|              Mean|
+----+-----+------------------+
|2011|   10| 87.11500133333334|
|2011|   11| 79.76380923809522|
|2011|   12| 70.30428566666667|
|2012|    1| 97.75149895000001|
|2012|    2|119.92049895000002|
|2012|    3|113.00181809090908|
|2012|    4|100.88399985000001|
|2012|    5| 72.98772681818181|
|2012|    6| 65.75380899999999|
|2012|    7|  75.2542851904762|
|2012|    8|60.736521347826084|
|2012|    9| 56.57736921052631|
|2012|   10| 65.78095142857143|
```

Instrucciones de evaluación

- Tiempo de entrega 22 de marzo 2022
- Al terminar poner el codigo y la documentación con su explicación en el branch correspondiete de su github, asi mismo realizar su explicación de la solución en su google drive en documento de google (Portada, Introduccion, Desarrollo, etc).
- Finalmente defender su desarrollo en un video de 6-8 min explicando su solución y obsevaciones, este servira para dar su calificación de esta practica evaluatoria, este video debe subirse a youtube para ser compartido por un link publico (Utilicen google meet con las camaras encendidas y graben su defensa para eleborar el video).

Link del video:https://youtu.be/cYICYxvh2LU

Happy Coding :) !