**INSTITUTO TECNOLÓGICO DE TIJUANA**

**SUBDIRECCIÓN ACADÉMICA**

**DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN**

**SEMESTRE AGOSTO-DICIEMBRE 2018**

**CARRERA:**

**Ing. Informática**

**MATERIA Y SERIE**

**Datos Masivos BDD-1704 IF9A**

**TÍTULO**

**Ejercicio 4**

**UNIDAD A EVALUAR**

**Unidad 1**

**NOMBRE Y # DE CONTROL DEL ALUMNO:**

**Beltrán Bustos Claudio Israel 14212317**

**GRUPO**

**BDD-1704 IF9A**

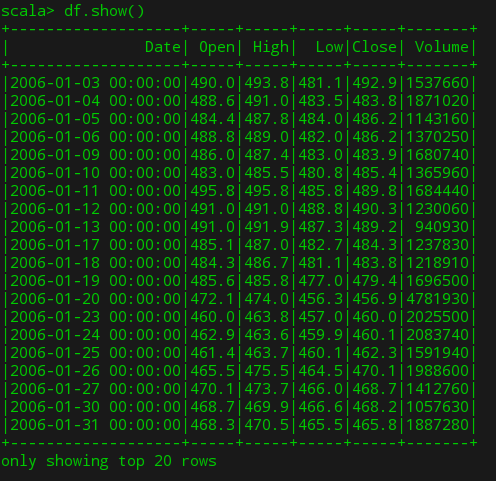
**NOMBRE DEL MAESTRO**

**José Christian Romero**

A continuación, se mostrarán imágenes y descripciones de 15 funciones utilizados para dataframes hechos en scala usando apache spark.

**1-- .show()**

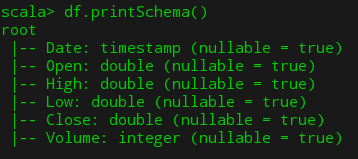
|  |
| --- |
| **df.show()** |



La función .show() permite ver los registros existentes en el dataframe, incluyendo aquellos registros con datos nulos. Aquí nos muestra los primeros 20 registros. Esto es muy útil para ver el contenido de nuestro dataframe, es decir, las variables cualitativas que usaremos para futuras operaciones y reportes. Existen más funciones que permiten hacer ciertos filtros bajo ciertas condiciones, lo cual hace la consulta mucho más concreta y más específica, el .show() se puede poner al final de los select() de tal manera que nos muestre el contenido filtrado que nosotros queremos.

**2-- .printSchema()**

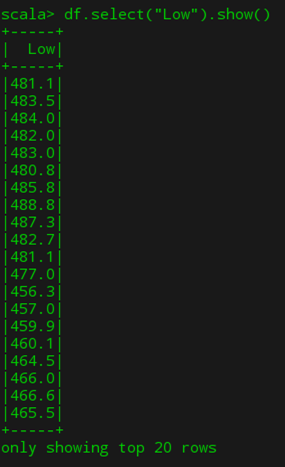
|  |
| --- |
| **df.printSchema()** |



Ésta función nos muestra las columnas; el nombre de las columnas, el tipo de datos y si el campo permite registros nulos o no. Esto nos puede ayudar a descifrar el volumen de las tablas para futuras operaciones y lo más importante, analizar el tipo de datos de dichas columnas para futuras operaciones de prueba (inserciones o modificaciones, por ejemplo).

**3-- .select()**

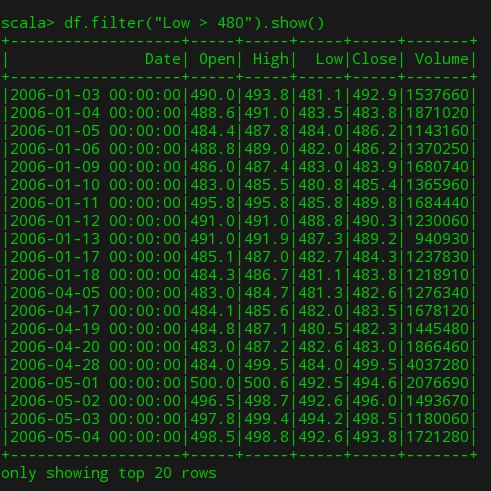
|  |
| --- |
| **df.select()** |



El select(), así como en SQL funciona de manera similar: nos permite consultar los datos a lo que nosotros queremos. Esta función acepta argumentos como funciones, objetos u otro tipo de dato para realizar dicha consulta, al final se escribe .show() para mostrar el resultado de la consulta.

**4-- .filter()**

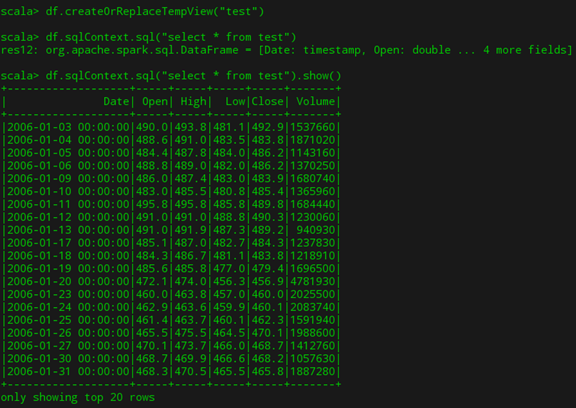
|  |
| --- |
| **df.filter("Low > 480").show()** |



Se usa .filter() para poder hacer consultas bajo ciertas restricciones, es decir, traer los datos bajo una condición. En éste caso, se muestran aquellos registros en donde el valor de la columna “Low” sea mayor a 480. Como se mencionó antes, .filter() se puede usar con los select() para comprimir la consulta. Esto permite hacer combinaciones de consultas en donde se permita guardar en las famosas vistas.

**5-- .createOrReplaceTempView(), 6-- .sql()**

|  |
| --- |
| **df.createOrReplaceTempView("test") df.sqlContext.sql("select \* from test").show()** |



La función createOrReplaceTempView() crea una vista temporal a partir de nuestro de dataframe, esto permite referenciarla para futuras operaciones con otras tablas (joins), sin la preocupación de modificar el nombre original del dataframe (si es que tiene una), por ende el “TempView”. La vista temporal es eliminada al finalizar el contexto de sql.

Se usa .sql() para usar sentencias tipo sql. En éste caso, hacemos un ‘select \* from test’, es decir, consultamos todos los elementos de la vista temporal ‘test’ que acabamos de crear.

**7-- .first()**

|  |
| --- |
| **df.first()** |



Ésta función nos regresa el primer registro del dataframe. En éste caso, nos regresa el primer registro en el dataframe de la crisis económica (nuestro archivo).

**8-- .count()**

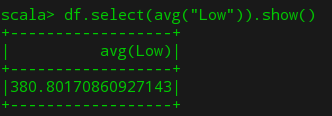
|  |
| --- |
| **df.count()** |



Ésta función nos regresa el número de registros contenido en el dataframe, es una función útil porque así podemos ver la cantidad de datos con los que estamos trabajando.

**9-- .avg()**

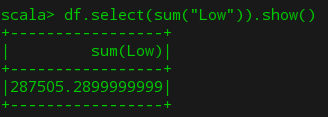
|  |
| --- |
| **df.select(avg("Low")).show()** |



La función avg() nos regresa el promedio de un conjunto de datos. En éste caso, nos regresa el promedio de la columna ‘Low’. Y como se había mencionado antes, se puede poner como argumento dentro de un select().

**10-- .sum()**

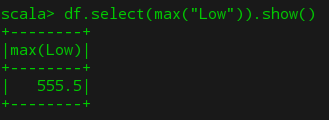
|  |
| --- |
| **df.select(sum("Low")).show()** |



La función sum() nos regresa la sumatoria de los datos, bajo la premisa de i = 0 hasta n, donde n = al total de los datos de dicho conjunto e i es un iterador para pasar a través de cada uno de ellos.

**11-- .max()**

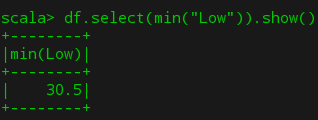
|  |
| --- |
| **df.select(max("Low")).show()** |



La función max() nos regresa el máximo de un conjunto de datos, es decir, el número de mayor valor. En éste caso, queremos que nos regrese el mayor valor de la columna ‘Low’.

**12-- .min()**

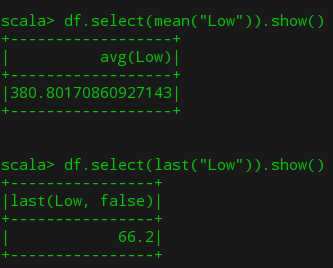
|  |
| --- |
| **df.select(min("Low")).show()** |



La función min() nos regresa el mínimo de un conjunto de datos, es decir, el número de mínimo valor. En éste caso, queremos que nos regrese el mínimo valor de la columna ‘Low’.

**13-- .mean(), 14-- .last()**

|  |
| --- |
| **df.select(mean("Low")).show() df.select(last("Low")).show()** |

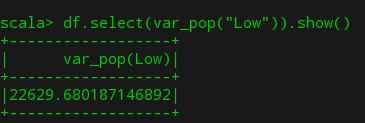


La función mean() nos regresa la media de un conjunto de datos., esto se requiere para hacer cálculos estadísticos como la varianza, así como otro tipo de operaciones que se deseen realizar.

La función last() nos regresa el último valor contenido en el dataframe, es decir, la última fila. En éste caso, se usó la columna ‘Low’ para sacar la media y obtener el último registro de dicha columna.

**15-- .var\_pop()**

|  |
| --- |
| **df.select(var\_pop("Low")).show()** |



La función var\_pop() nos regresa la varianza poblacional de un conjunto de datos, en éste caso, queremos que nos calculara la varianza poblacional de la columna ‘Low’. Ésta es una función más que se usa en los cálculos probabilísticos y estadísticos.