# Fundamentos de Big Data

## ¿Qué es Spark SQL?

Módulo de Apache Spark diseñado para trabajar con datos estructurados.

#### **Características:**

Compatibilidad con SQL estándar

**Uso de DataFrames y Datasets** 

**Optimización con Catalyst** 

Compatibilidad con múltiples fuentes de datos

Interoperabilidad con RDDs



```
Tempils Tensorflowal etry Keras
11 Indleants Luth 1 CNN
2 AABE Casses Laf)
  1 Neccleatsacible Freem serving and the est Caylic chat)
3 Lest Verris, Painer collentage)
   4 Corsitation, Cantel metimate/Thetom and Keray,
    6 Lent Tingla Restion()
    7 Chok timet where conffileatersente) =
    8 last fernpounter on <fighrslahteschapallt, Milerary Fucom Fortle, Convertible(/11)
    3. Prast—Flong-Stncker Reston()
    2 rehieal.
    3 Ill conect, electyeniflaion ));
    6 Lins tent seretler of Atine State on efcherce),
     6 Creatt -- -- =
     7 CNN for sich a electadgy reherel and guter and the attaps from!)
    8 irf tions, wat steach, chatolactoment are gnarylates.
     4 /elercrsionte).
    13 Varteal (nistroicy derka):
        iff the provyleyenate/fine /lesolsate - muting).
       1rf slagresion: the intoôt, Potall wartiong del isservial)
       Settistom Stew the Kenate iantal(lappfferts it gett wattel contenteer))
       iff moitrite ' ([lanstegyVant on thel wankes of gaiverled)
    17 Inwaill, fits - Inlestate Heron Flal laser:ting.

☑ Godor:ice-Tipe: 9 gdents ]
```

#### DataFrame

Colección distribuida de datos

Ejecución diferida (lazy evaluation)

Creación a partir de listas, diccionarios, CSV, JSON, Parquet o RDD

## Formatos de Archivo: JSON y Parquet

#### **JSON**

- Formato ligero y basado en texto.
- Auto-descriptivo y legible.
- . Compatible con la mayoría de los lenguajes de programación.

Ejemplo de lectura y escritura en JSON:

#### **Parquet**

- Formato de almacenamiento columnar.
- . Optimizado para consultas rápidas y compresión eficiente.
- . Recomendado para grandes volúmenes de datos.

Ejemplo de lectura y escritura en Parquet:

```
df_json = spark.read.json("datos.json")
df_json.write.json("salida.json")
```

```
df_parquet = spark.read.parquet("datos.parquet")
df_parquet.write.parquet("salida.parquet")
```

### Lectura y Escritura Distintas Fuentes

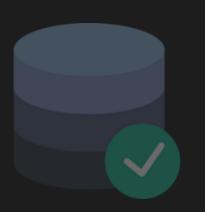
Lectura desde una base de datos MySQL

2

Escritura en una base de datos PostgreSQL

```
df_mysql = spark.read.format("jdbc") \
    .option("url", "jdbc:mysql://localhost:3306/mi_base") \
    .option("dbtable", "usuarios") \
    .option("user", "root") \
    .option("password", "1234") \
    .load()
df_mysql.show()
```

```
df.write.format("jdbc") \
    .option("url", "jdbc:postgresql://localhost:5432/mi_base") \
    .option("dbtable", "usuarios") \
    .option("user", "postgres") \
    .option("password", "1234") \
    .save()
```



#### Actividad Práctica Guiada

**Objetivo:** Implementar Spark SQL para procesar datos estructurados, usando DataFrames, consultas SQL, UDFs y Lectura/Escritura en distintos formatos.

#### **Requisitos:**

- 1. Requisitos previos: Instalación de Apache Spark, PySpark, tener los datos en formato JSON o CSV (revisar guía de estudio).
- 2. Configurar una sesión de Spark.
- 3. Cargar los datos en un DataFrame.
- 4. Explorar el esquema del DataFrame.
- 5. Registrar el DataFrame como una tabla SQL.
- 6. Crear una Función Definida por el Usuario (UDF).
- 7. Guardar los datos en formato Parquet.
- 8. Leer datos desde una base de datos.

## Preguntas

Sección de preguntas





#### Fundamentos de

## Big Data

Continúe con las actividades