

Spark Data Sources

DataFrameReader

- ▶ `DataFrameReader` is the core construct for reading data from a data source into a `DataFrame`. Recommended format:

```
DataFrameReader.format(args).option("key", "value")  
    .schema(args).load()
```
- ▶ We can access a `DataFrameReader` through a `SparkSession` instance by using one either `SparkSession.read` or `SparkSession.readStream`.
- ▶ In general, we don't need a schema when reading a Parquet file as it is contained in its metadata. Parquet is the preferred data source because it is efficient, uses columnar storage, and employs a fast compression algorithm.

DataFrameReader Parameters

Method	Arguments	Description
<code>format()</code>	"parquet", "csv", "txt", "json", "jdbc", "orc", "avro" etc	Default is parquet or as specified in the config for <code>spark.sql.sources.default</code> .
<code>option()</code>	("mode", {PERMISSIVE FAILFAST DROPMALFORMED }) ("inferSchema", {true false}) ("path", "path_file_data_source")	default is PERMISSIVE mode and inferSchema is for JSON and CSV sources For JSON and CSV formats
<code>schema()</code>	DDL String or StructType	
<code>load()</code>	"path/to/data/source"	optional if already specified above

DataFrameWriter

- ▶ `DataFrameWriter` is the core construct for writing data from a `DataFrame` to external source a `DataFrame`. Recommended formats:

```
DataFrameWriter.format(args).option("key", "value")  
                        .bucketBy(args).partitionBy(args)  
                        .save(path)
```

```
DataFrameWriter.format(args).option("key", "value")  
                        .sortBy(args).saveAsTable(table)
```

- ▶ We can access a `DataFrameWriter` directly through a `DataFrame` instance by using either `DataFrame.write` or `DataFrame.writeStream`.

DataFrameWriter Parameters

Method	Arguments
<code>format()</code>	"parquet", "csv", "txt", "json", "jdbc", "orc", "avro" etc
<code>option()</code>	("mode", {append overwrite ignore error or errorifexists}) ("mode", {SaveMode.Overwrite SaveMode.Append SaveMode.Ignore} SaveMode.ErrorIfExists}) ("inferSchema", {true false}) ("path", "path_to_write_to")
<code>bucketBy()</code>	DDL String or StructType
<code>save()</code>	"path/to/data/source"
<code>saveAsTable()</code>	"table_name"

Overview of Data Source Types

- ▶ Text
- ▶ CSV
- ▶ JSON
- ▶ Parquet
- ▶ Avro: Used by Apache Kafka for serialization. Offers direct mapping to JSON, speed and efficiency, and bindings for many programming languages.
- ▶ ORC (Optimized Row Columnar) file format provides a highly efficient way to store Hive data.
- ▶ Image data
- ▶ Arbitrary binary data: The DataFrame has the following attributes: path, modificationTime, length, and content. Writing isn't supported.

External Data Sources

- ▶ Popular tools like Tableau and PowerBI can connect simply to spark engine using the Spark URL (like we do for connecting to the cluster).
- ▶ We can also easily connect to SQL data bases using the JDBC connector and appropriate jar file for PostgreSQL, MySQL, SQLite, Azure Cosmos DB, MS SQL Server.
- ▶ Connect to distributed databases, data lakes such as Cassandra, MongoDB, Snowflake, etc. For example, see:

<https://github.com/datastax/spark-cassandra-connector>