Apache Spark 2 using SQL - Getting Started

1. Introduction

Apache Spark SQL is a module in Spark for:

- Executing SQL queries
- Integrating with structured/semi-structured data (Parquet, ORC, JSON, CSV)
- Working with **DataFrames** and **Datasets**
- Optimizing queries using the **Catalyst optimizer**

It allows you to:

- Use both **programmatic API** and **pure SQL syntax** inside Spark applications.
- Mix SQL with DataFrame operations.
- Read/write data from multiple formats.

2. Key Concepts

Term	Meaning
SparkSession	Entry point for Spark SQL. Manages DataFrame and SQL query execution.
DataFrame	Distributed collection of data organized into named columns.
Temporary View	In-memory table created from a DataFrame to run SQL queries.
Global Temporary View	View accessible across sessions, tied to the Spark application.
Catalog	Metadata store of databases, tables, and functions.

3. Step-by-Step: Getting Started with Spark SQL in Scala

3.1 Setup

```
import org.apache.spark.sql.SparkSession
object SparkSQLExample {
  def main(args: Array[String]): Unit = {

    // 1. Create SparkSession
    val spark = SparkSession.builder()
      .appName("Spark SQL Getting Started")
      .master("local[*]")
      .getOrCreate()
    spark.sparkContext.setLogLevel("ERROR")
    // 2. Import implicits for DF to DS conversion
    import spark.implicits._
    // 3. Load sample data
    val data = Seq(
      (1, "Anjali", 3000),
      (2, "Ram", 4000),
      (3, "Chitra", 5000)
    )
    val df = data.toDF("id", "name", "salary")
    // 4. Create Temporary View
    df.createOrReplaceTempView("employees")
    // 5. Run SQL Query
    val result = spark.sql("SELECT name, salary FROM employees WHERE salary > 3500")
    // 6. Show Results
    result.show()
    // 7. Stop Spark Session
    spark.stop()
  }
}
```

Output:

```
+-----+------+
| name|salary|
+-----+------+
|  Bob|  4000|
|Charlie| 5000|
+-----+------+
```

4. Key Functions in Spark SQL

Function	Description	Example
spark.sql()	Executes a SQL query string	spark.sql("SELECT * FROM table")
createOrReplaceTempView()	Creates session-level temp view	df.createOrReplaceTempView("view1")
createGlobalTempView()	Creates global temp view	df.createGlobalTempView("view2")
spark.catalog.listTables()	Lists tables in catalog	spark.catalog.listTables().show()

5. Common Data Formats in Spark SQL

```
// Reading JSON
val jsonDF = spark.read.json("data/employees.json")

// Reading CSV
val csvDF = spark.read.option("header", "true").csv("data/employees.csv")

// Reading Parquet
val parquetDF = spark.read.parquet("data/employees.parquet")
```

6. Example – SQL + DataFrame API Together

scala
CopyEdit

```
// DataFrame API
val highPaid = df.filter($"salary" > 3500)
// SQL
df.createOrReplaceTempView("emp")
```

```
val sqlResult = spark.sql("SELECT name FROM emp WHERE salary > 3500")
highPaid.show()
sqlResult.show()
```

Here, both **API** and **SQL** give the same result.

7. GitHub Repositories

- 🔹 Spark SQL Examples – SparkByExamples
- 🔹 Apache Spark Official Examples
- 🔹 Scala + Spark SQL Demo Projects

8. Quick Quiz

Q1. What is the main entry point for Spark SQL?
a) SQLContext
b) SparkSession
c) HiveContext
d) DataFrame

Q2. Which method creates a session-scoped temporary view?
a) createGlobalTempView()
b) createOrReplaceTempView()
c) createTable()
d) cacheTable()

Q3. True or False: Spark SQL can only query data stored in Parquet format.

Q4. Which optimizer does Spark SQL use internally?
a) Volcano
b) Catalyst
c) Cost-based Optimizer
d) Rule-based Parser

Q5. Fill in the blank:
To run SQL queries in Spark, we use __________ method.