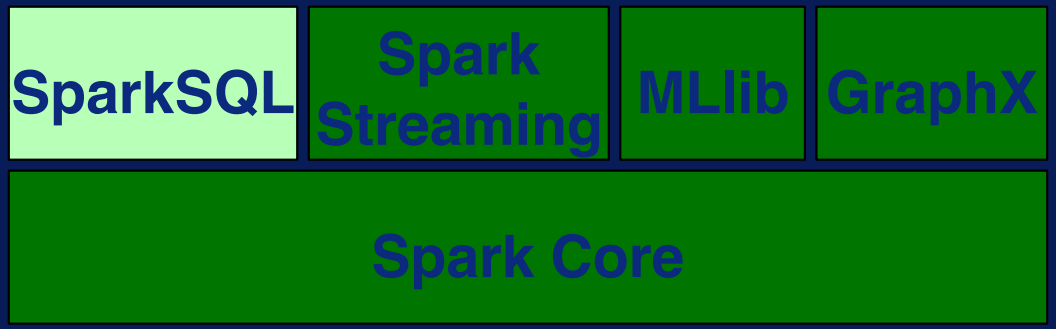
Spark SQL

After this video you will be able to..

- Process structured data using Spark's SQL module
- Explain the numerous benefits of Spark SQL



Spark SQL

- Enables querying structured and unstructured data through Spark
- Provides a common query language
- Has APIs for Scala, Java and Python to convert results into RDDs — can connect to many data sources

Relational Operations

Perform Relational Processing such as Declarative Queries

Embed SQL queries inside Spark Programs

- Spark SQL gives a mechanism for SQL users to deploy SQL queries on Spark

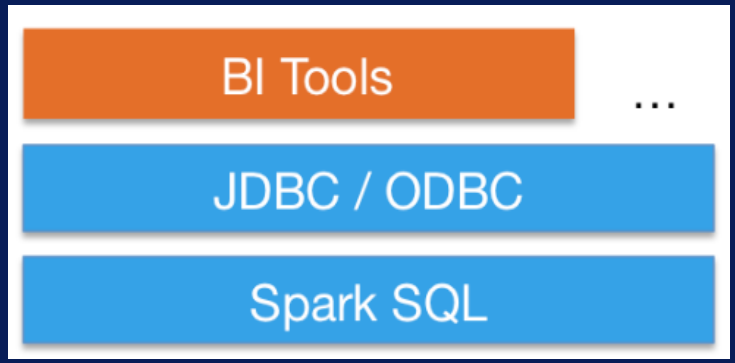
Business Intelligence Tools

Spark SQL connects to all BI tools that support JDBC or ODBC standard

enables

business intelligence

standard connection protocols



http://spark.apache.org/

DataFrames

Distributed Data organized as named columns

Look just like a table in relational databases

- APIs to convert query data into DataFrames to hold distributed data

How to go Relational in Spark ?

first step to run SQL Spark:

Step 1: Create a SQLContext

```
from pyspark.sql import SQLContext
sqlContext = SQLContext(sc)
```

- once you have an SQL context, leverage it to create a DataFrame so you can deploy complex operations on the data set

How to go Relational in Spark ?

Create a DataFrame from

- an existing RDD
- a Hive table
- data sources

JSON → DataFrame

```
# Read
df = sqlContext.read.json("/filename.json")

# Display
df.show()
```

file can be read and converted into DataFrame with single command

displays DataFrame in Spark show

RDD of Row objects → DataFrame

conversion requires more work

```
# Read
from pyspark.sql import SQLContext, Row
sqlContext = SQLContext(sc)

# Load a text file and convert each line to a Row.
lines = sc.textFile("filename.txt")
cols = lines.map(lambda l: l.split(","))
data = cols.map(lambda p: Row(name=p[0], zip=int(p[1])))

# Create DataFrame
df = sqlContext.createDataFrame(data)

# Register the DataFrame as a table
df.registerTempTable("table")

# Run SQL
Output = sqlContext.sql("SELECT * FROM table WHERE …")
```

1.

2.

once is a dataframe, you can perform all sorts of transformation operations on it

DataFrames are just like tables

http://spark.apache.org/

```
# Show the content of the DataFrame
df.show()

# Print the schema
df.printSchema()

# Select only the "X" column
df.select("X").show()

# Select everybody, but increment the discount by 5%
df.select(df["name"], df["discount"] + 5).show()

# Select people height greater than 4.0 ft
df.filter(df["height"] > 4.0).show()

# Count people by zip
df.groupBy("zip").count().show()
```

Spark SQL

spark allows to:

run

Relational on Spark

queries

Connect to variety of databases

Deploy business intelligence tools over Spark