scark-cli Documentation

Release 1.1.0

Lukas Heumos

Sep 02, 2019

Contents:

CHAPTER 1

scark-cli

A tool for submitting SQL queries to a Spark Cluster. Currently MariaDB is supported out of the box. scark-cli is designed to interoperate with spark-service.

1.1 Features

- Submit SQL queries to a MariaDB database locally or through a spark network

1.2 Documentation

The documentation for scark-cli is hosted here.

1.3 Authors

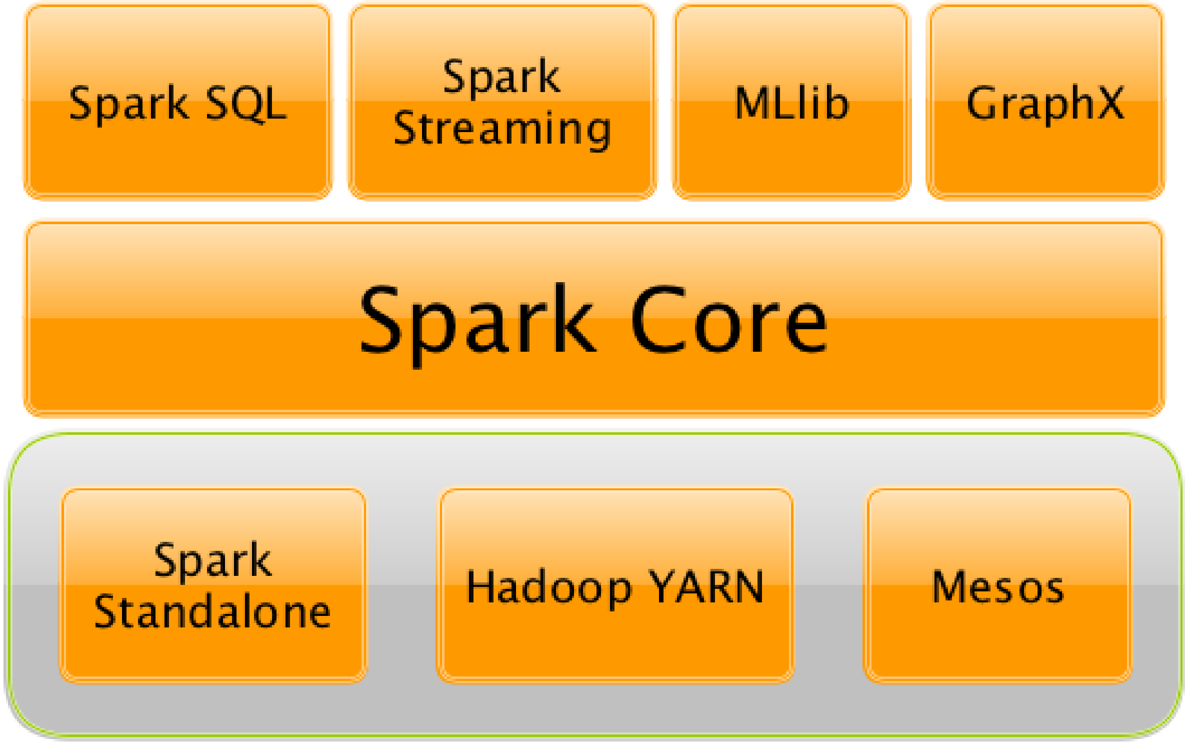
scark-cli was designed and implemented by Lukas Heumos.

CHAPTER 2

Spark

2.1 About Apache Spark

Apache Spark is a unified analytics engine for large-scale data processing. Basically, it's a cluster computing framework offering a stack of libraries for SQL and data frames (Spark SQL), machine learning (MLlib), graphs (GraphX) and streaming (Spark Streaming).



Although, Spark is written in Scala, applications can also be submitted in Java, Python, R and SQL. Spark can either be run in its standalone cluster mode or on EC2, on Hadoop Yarn, Mesos or on Kubernetes.

2.2 Comparing Apache Spark with Apache Hadoop

Apache Spark differs from Apache Hadoop in their different approaches to processing. Spark is able to do it in-memory, whereas Hadoop's MapReduce has to read from and write to a disk. As a result, the processing speed differs significantly. Commonly, Spark is said to be up to 100 times faster.

However, the volume of data processed also differs: Hadoop's MapReduce is able to work with far larger datasets than Spark. Incase the dataset is larger than available RAM, Hadoop MapReduce may outperform Spark. Due to Hadoop's worse performance, it should only be used if no immediate results are expected e.g. if processing can be done overnight. Spark's iterative processing approach is especially useful, when data is processed multiple times. Spark's Resilient Distributed Datasets (RDDs) enable multiple map operations in memory, while Hadoop MapReduce has to write interim results to a disk.

CHAPTER 3

Installation

The following sections will guide you through building, testing and obtaining the latest release of scark-cli.

3.1 Obtaining the latest release

If you're not a developer and not interested in building the code itself, simply download the latest release. Next, refer to usage for detailed instructions about how to submit SQL queries locally and to the spark network.

3.2 Getting the code

The code is hosted on github. Clone the repository via:

```
git clone https://github.com/qbicsoftware/scark-cli
```

3.3 Building

scark-cli is build with sbt. Furthermore, the sbt assembly plugin is configured allowing for fat jars to be build:

```
sbt assembly
```

will build the fat jar. The result will be written to

```
/target/$scala-version/$name-assembly-$version.jar
```

3.4 Testing

Run tests *inside the sbt console* from the root project directory using:

```
test
```

Next, refer to usage for detailed instructions about how to submit SQL queries locally and to the spark network.

CHAPTER 4

Usage

The following sections will guide you through running scark-cli, all possible parameter options and submitting SQL queries to Spark networks.

4.1 Running

```
java -jar scark-cli-1.1.0.jar
```

You should now see the help menu.

4.2 Options

```
Usage: Benchmark [-h] -q[=<sqlQuery>] -c=<configFilePath>
Benchmark Tool for evaluating the performance of a Spark Cluster. Run custom
SQL Queries inside Spark!
-s, --spark                 run with spark support
-l, --local                 run spark in local mode - requires -s option to be in␣
↪effect
-t, --table[=<tables>]      list of tables to execute SQL query in, mandatory if␣
↪running with spark support
-d, --driver[=<driver>]     driver to access Database, e.g. org.mariadb.jdbc.Driver,␣
↪mandatory if running with spark support
-q, --query[=<sqlQuery>]    SQL query to execute
-c, --config[=<configFilePath>]
                            database config file path
-h, --help                  display a help message
```

Required parameters are:

```
-c, --config=<configFilePath>
-t  --table[=<table>]
-q, --query[=<sqlQuery>]
```

Queries are optionally interactive. You can either use `-q` to get a prompt for your query or supply a full query when running the tool: `--q[=<sqlQuery>]`.

4.3 Configuration file

The configuration file should look as follows:

host: jdbc:mysql://<URL>/<database>

port: “<port>”

name: “<database>”

pw: “<password>”

user: “<username>”

4.4 Spark

Note that for scark-cli to be run inside a containerized spark network, such as one set up by spark-service, it has to be available in the container.

Refer to the documentation of your spark container about how to mount volumes and provide access to scark-cli. A query can be submitted to spark via:

```
/spark/bin/spark-submit --master spark://spark-master:7077 \
/opt/spark-apps/scark-cli-1.1.0.jar \
-c /opt/spark-data/database_properties.txt \
-s \
-t <table> \
-q <"query"> \
-d org.mariadb.jdbc.Driver
```

4.5 Example Query

```
/spark/bin/spark-submit --master spark://spark-master:7077 \
/opt/spark-apps/scark-cli-1.1.0.jar \
-c /opt/spark-data/database_properties.txt \
-s \
-t Consequence \
-q "SELECT id FROM Consequence" \
-d org.mariadb.jdbc.Driver
```

4.6 Complex Query

```
/spark/bin/spark-submit --master spark://spark-master:7077 \
/opt/spark-apps/scark-cli-1.1.0.jar \
-c /opt/spark-data/database_properties.txt \
-s \
-t Consequence Variant Variant_has_Consequence \
-q "select * from Variant INNER JOIN Variant_has_Consequence ON Variant.id = Variant_
↪has_Consequence.Variant_id INNER JOIN Consequence on Variant_has_Consequence.
↪Consequence_id = Consequence.id" \
-d org.mariadb.jdbc.Driver
```

CHAPTER 5

Known issues

1. Due to a bug in the MariaDB connector and Spark, mariadb in the jdbc URL has to be replaced with mysql. Please refer to: #9.

2. java.io.IOException: No FileSystem for scheme: file #13 when running scark-cli in local mode as a jar Please refer to: #13.

CHAPTER 6

History

- 09.07.2019: [1.0.0]
- 16.07.2019: [1.1.0]

CHAPTER 7

Contributing

Please submit open issues and pull requests to scark-cli.

CHAPTER 8

Authors

spark-cli was designed and implemented by Lukas Heumos.

CHAPTER 9

Indices and tables

- search