SparkR basics

Introduction

This document explains how **SparkR** library can be used to leverage Spark capabilities.

Syntasa supports Spark code process in three distinct runtimes - Python, R and Scala. A user can write R code and to perform queries, data transformation, ETL etc. using Spark runtime in either GCP or AWS or on-prm environment.

The code gets submitted as a `spark-submit` command. For example, `spark-submit test_script.R` . When it comes to productionizing the workflow that contains this process, user can use variables/parameters to configure the script/code.

As in any other environment, the entry point in the code is `sparkR.session()` . This session connects your R progam to a Spark cluster. Additional configurations such as `driver.memory` , `executor.memory` can be passed.

Below code loads the SparkR library and launches Spark session (**NOTE**: On Local machine from R Studio, you may need to set SPARK_HOME but it is not needed on Google DataProc or AWS EMR)

```
Sys.setenv(SPARK_HOME ='/Users/pmehta/spark/spark-2.1.0-bin-hadoop2.7' )
.libPaths(c(file.path(Sys.getenv('SPARK_HOME') , 'R', 'lib'), .libPaths()))
library(SparkR)
sparkR.session()
```

Reading data from Hive tables

Once spark session is ready, we can read the data from Hive tables in Syntasa. Below example shows how to read a Hive table in Spark session in **4.0 QA environment**.

```
df <- sql("select visitor_id from dev_pranay_eventstore.atb_success limit 10")
createOrReplaceTempView(df, "df")
head(df)

                                visitor_id
1 -1001024707886664113-8216421554741634351
2  -1013501061646903335265795263611045927 5
3  -1015007571009945978103317900591824802 1
4  -1015677126577342708532357830933462363 1
5 -1015917284104657576-4591914585702054177
6 -1016308368085134159-7766967906350841878
```

Convert Spark Dataframe to R Dataframe

Whenever you read data using Spark session, the `type` returned is a `SparkDataFrame` - which is distributed dataframe implementation.

```
class(df)
[1] "SparkDataFrame"
attr(,"package")
[1] "SparkR"
```

To convert this to an `R` DataFrame you can use -

```
R_df <- as.data.frame(df)

class(R_df) # This is an R dataframe (not distributed anymore)
[1] "data.frame"
```

Writing data back to Hive tables

You can write results of your SparkDataFrame back to Hive. `sql()` method helps us create tables and push results into these tables.

Below example creates a table and pushes 10 visitor_id's in that table

```
# Create a table in Hive
sql("create table if not exists dev_pranay_eventstore.temp_table (visitor_id STRING)")
# Push the results
sql("insert overwrite table dev_pranay_eventstore.temp_table select visitor_id from df")
```

For more details, see https://spark.apache.org/docs/latest/sparkr.html (https://spark.apache.org/docs/latest/sparkr.html)