Data Source Options

The Sparky data source constructor has two parameters:

- `path` - the file system path where to begin the search for HDF5 files
- `dataset` - the HDF5 path name of the dataset to be read (can also be used to get virtual table summaries)

The following options can be used to further configure the data source behavior.

Option	Comment	Default
`extension`	A comma-separated list of HDF5 file extensions	`h5`
`window size`	The maximal number of elements to be read from an HDF5 dataset in one read operation.	10,000
`recursion`	Recursively scan sub-directories for HDF5 files	`true`
`start`	The offset of `block`. A comma-separated list of non-negative integers whose length equals the rank of `dataset`.	-1
`block`	The dimensions of the block to be read. A comma-separated list of positive integers.	-1

Dataset input	Schema
`sparky://files`	"FileID": Integer, "FilePath": String, "FileSize": Long
`sparky://datasets`	"FileID": Integer, "DatasetPath": String, "ElementType": String, "Dimensions": Array(Long), "ElementCount": Long
`sparky://attributes`	"FileID": Integer, "ObjectPath": String, "AttributeName": String, "ElementType": String, "Dimensions": Array(Long)
`/path/to/dataset`	"FileID": Integer, "Index": Long, "Value": Datatype

Include the following:

```
/* Run with:
  time spark-shell -i test.scala --jars
lib/5parky_2.11-0.0.1-ALPHA.jar,lib/sis-jhdf5-batteries_included.jar
 **/

import org.hdfgroup.spark.hdf5._
import org.apache.spark.sql.functions._
val sqlContext = new org.apache.spark.sql.SQLContext(sc)
val files = "/path/to/files/"
val dataset = "/path/to/dataset"
// ADD CODE HERE
System.exit(0)
```

Extensions and Virtual Dataset examples:

```
val df = sqlContext.read.hdf5(files, "sparky://files")
val df = sqlContext.read.hdf5(files, "sparky://datasets")
val df = sqlContext.read.hdf5(files, "sparky://attributes")
```

Window size example:

```
val df = sqlContext.read.option("window size", "100000").hdf5(files,
dataset)
```

Recursion example:

```
val df = sqlContext.read.option("recursion", "false").hdf5(files, dataset)
```

Start/Block example:

```
val df = sqlContext.read.option("start", "1,1").option("block",
"50,50").hdf5(files, dataset)
```