Article
Benjamin De Boe · Jan 31, 2018 4m read

Introducing the InterSystems IRIS Connector for Apache Spark

With the release of InterSystems IRIS, we're also making available a nifty bit of software that allows you to get the best out of your InterSystems IRIS cluster when working with Apache Spark for data processing, machine learning and other data-heavy fun. Let's take a closer look at how we're making your life as a Data Scientist easier, as you're probably already facing tough big data challenges already, just from the influx of job offers in your inbox!

What is Apache Spark?

Together with the technology itself, we're also launching an exciting new learning package called the InterSystems IRIS Experience, an immersive combination of fast-paced courses, crisp videos and hands-on labs in our hosted learning environment. The first of those focuses exclusively on our connector for Apache Spark, so let's not reinvent the introduction wheel here and refer you to the course for a broader introduction. It's 100% free, after all!

In (very!) short, Apache Spark offers you an abstract object representation of a potentially massive dataset. As a developer, you just call methods on this Dataset interface like filter(), map() and orderBy() and Spark will make sure they are executed efficiently, leveraging the servers in your Spark cluster by parallelizing the work as much as possible. It also comes with a growing set of libraries for Machine Learning, streaming and analyzing graph data that leverage this dataset paradigm.

Why combine Spark with InterSystems IRIS?

Spark isn't just good at data processing and attracting large crowds at open source conferences, it's also good at allowing smart database vendors to participate in this drive to efficiency through its Data Source API. While the Dataset object abstracts the *user* from any complexities of the underlying data store (which could be pretty crude in the case of a file system), it does offer the *database vendors* on the other side of the object a chance to provide information on what those complexities (which we call features!) may be leveraged by Spark to improve overall efficiency. For example, a filter() call on the Dataset object could be forwarded to an underlying SQL database in the form of a WHERE clause. This **predicate pushdown** mechanism means the compute work is pushed closer to the data, allowing greater overall efficiency and throughput and part of building a connector for Apache Spark means registering all core functions that can be pushed down into the data source.

Besides this predicate pushdown, which many other databases support for Spark, our InterSystems IRIS Connector offers a few other unique advantages:

- The connector is designed to work well with sharding, our new option for horizontal scalability. More specifically, if you're building a Dataset based on a sharded table, we'll make sure **Spark slaves connect directly to the shard servers** to read the data in parallel, rather than stand in line to pipe the data through the shard master.
- As part of our new container deployment options, we're also offering a container that has both InterSystems IRIS and Apache Spark included. This means that when setting up a (sharded) cluster of these using ICM, you'll automatically have a Spark slave running alongside each data shard, allowing them to **exploit data locality when reading or writing sharded tables**, avoiding any network overhead. Note that if you set up your Spark and InterSystems IRIS clusters manually, with a Spark slave running on each server that has an InterSystems IRIS instance, you'll also benefit from this.
- When reading data, the connector can **implicitly partition the data being read** by exploiting the same mechanism our SQL query optimizer uses in %PARALLEL mode. Hereby, multiple connections to the same InterSystems IRIS instance are opened to read the data in parallel, increasing throughput. With the basics

in place already, you'll see further speedups coming up in InterSystems IRIS 2018.2.
- Also starting with InterSystems IRIS 2018.2, you'll be able to **export predictive models built with** SparkML to InterSystems IRIS with a single iscSave() method call. This will automatically generate a PMML class on the database side with native code to run the model in InterSystems IRIS in real time or batch scenarios.

Getting started with the InterSystems IRIS Connector is easy, as it's plug-compatible with the default JDBC connector that ships with Spark. So any Spark program that started with

```
var dataset = spark.read.format("jdbc")
                        .option("dbtable", "BigData.MassiveSales")
```

can now become

```
import com.intersys.spark._
var dataset = spark.read.format("iris")
                        .option("dbtable","BigData.MassiveSales")
```

Now that's just the bare essentials to get you started wit InterSystems IRIS as the data store behind your Apache Spark cluster. The rest will only be constrained by your Data Science imagination, coffee supply and the 24h in a typical day. Come and try it yourself in the InterSystems IRIS Experience for Big Data Analytics!

#AI #Analytics #Big Data #Distributed Data Management #Java #Machine Learning #Sharding #InterSystems IRIS

Source URL: https://community.intersystems.com/post/introducing-intersystems-iris-connector-apache-spark