R Sever & HDInsights Insights

slalom

HDInsight

What it is:

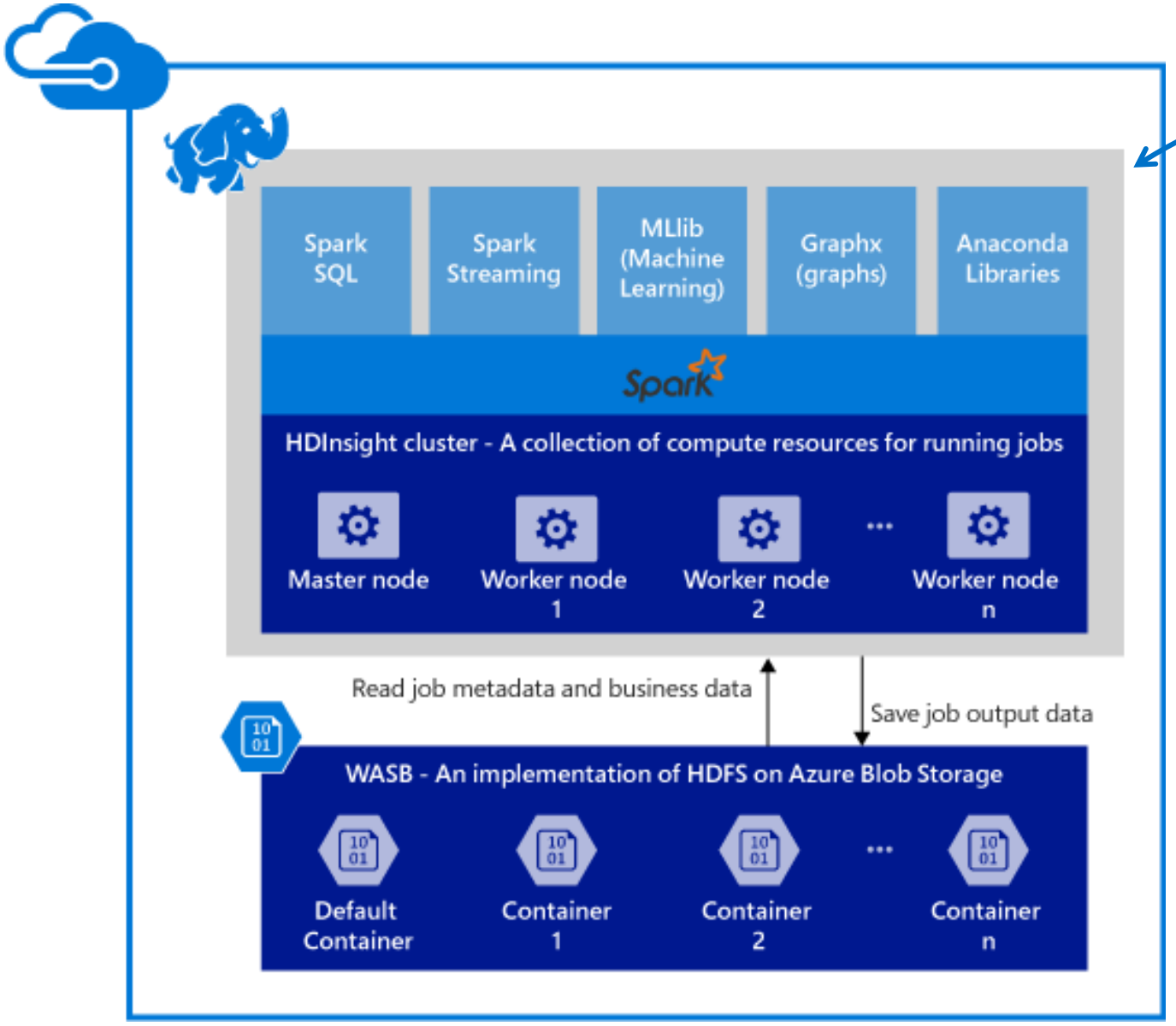
Microsoft's implementation of apache Hadoop (as a service) that uses Blobs for persistent storage

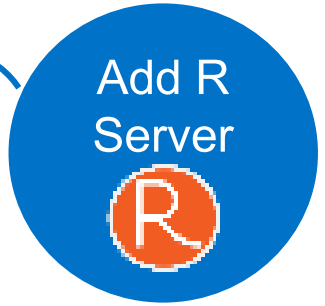
When to use it:

- When you need to process large scale data (PB+)
- When you want to use Hadoop or Spark as a service
- When you want to compute data and retire the servers, but retain the results
- When your team is familiar with the Hadoop Zoo

http://azure.microsoft.com/en-us/services/hdinsight/

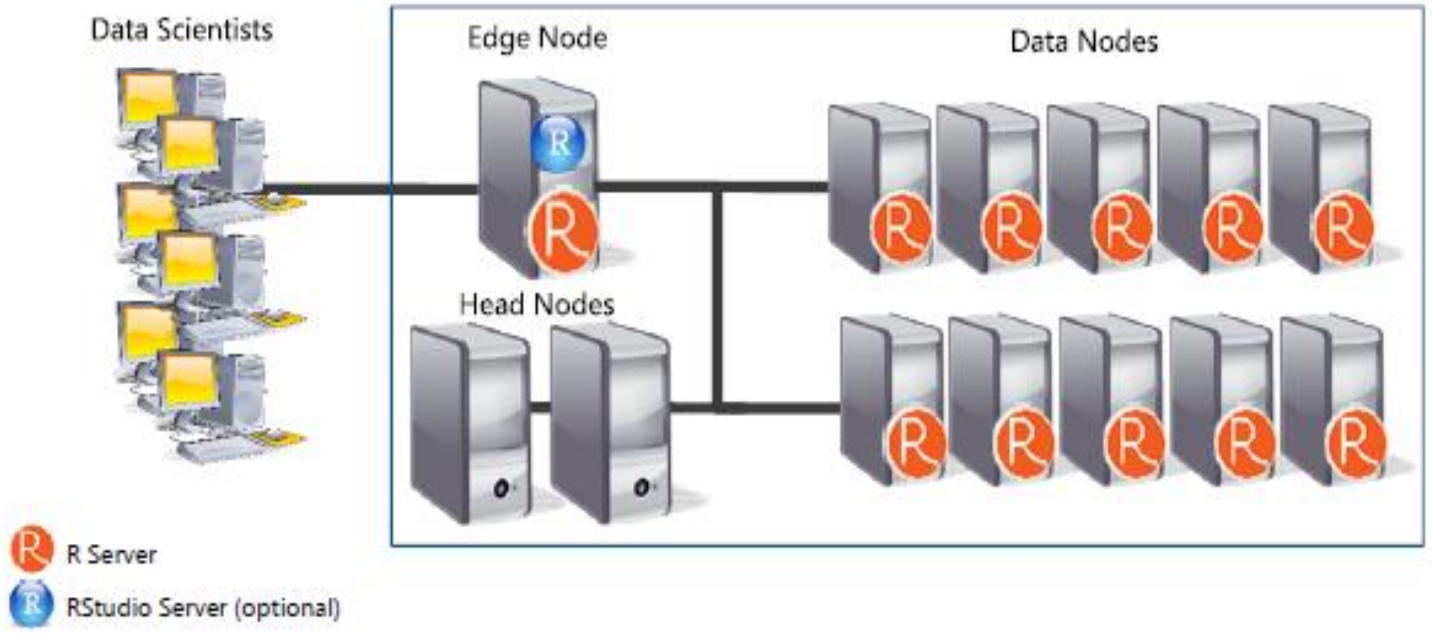
HDInsights = Hadoop and/or Spark clusters





- Provisions Azure compute resources with Spark 1.6 installed and configured.
- Data is stored in Azure Blob storage (WASB).

R Server HDInsights Architecture



R Server Distributed Processing:

Master R process on Edge node

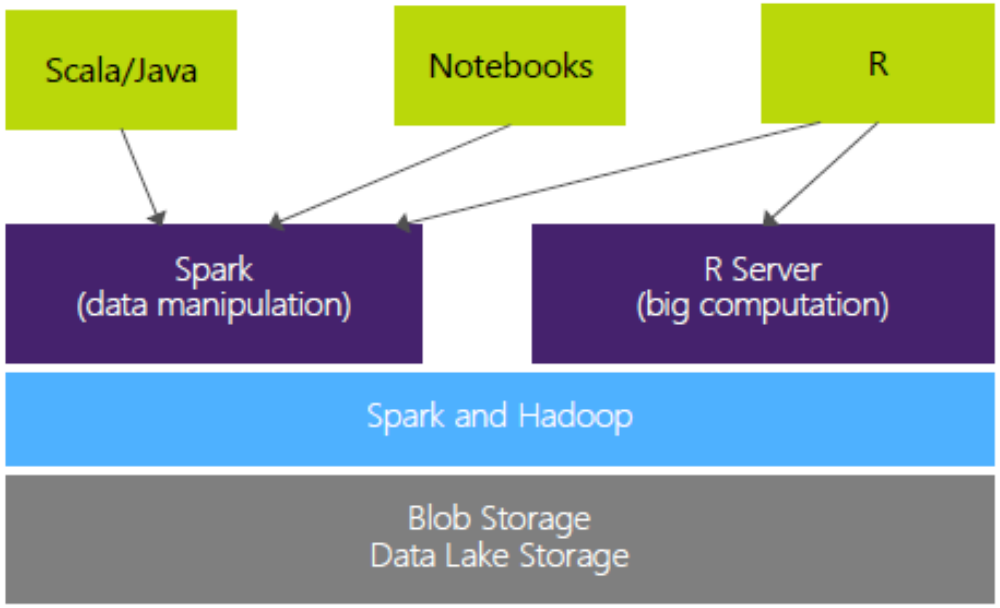
↓

Apache YARN and Spark

↓

Worker R processes on Data nodes

HDInsights w/R Server = managed Hadoop for advanced analytics

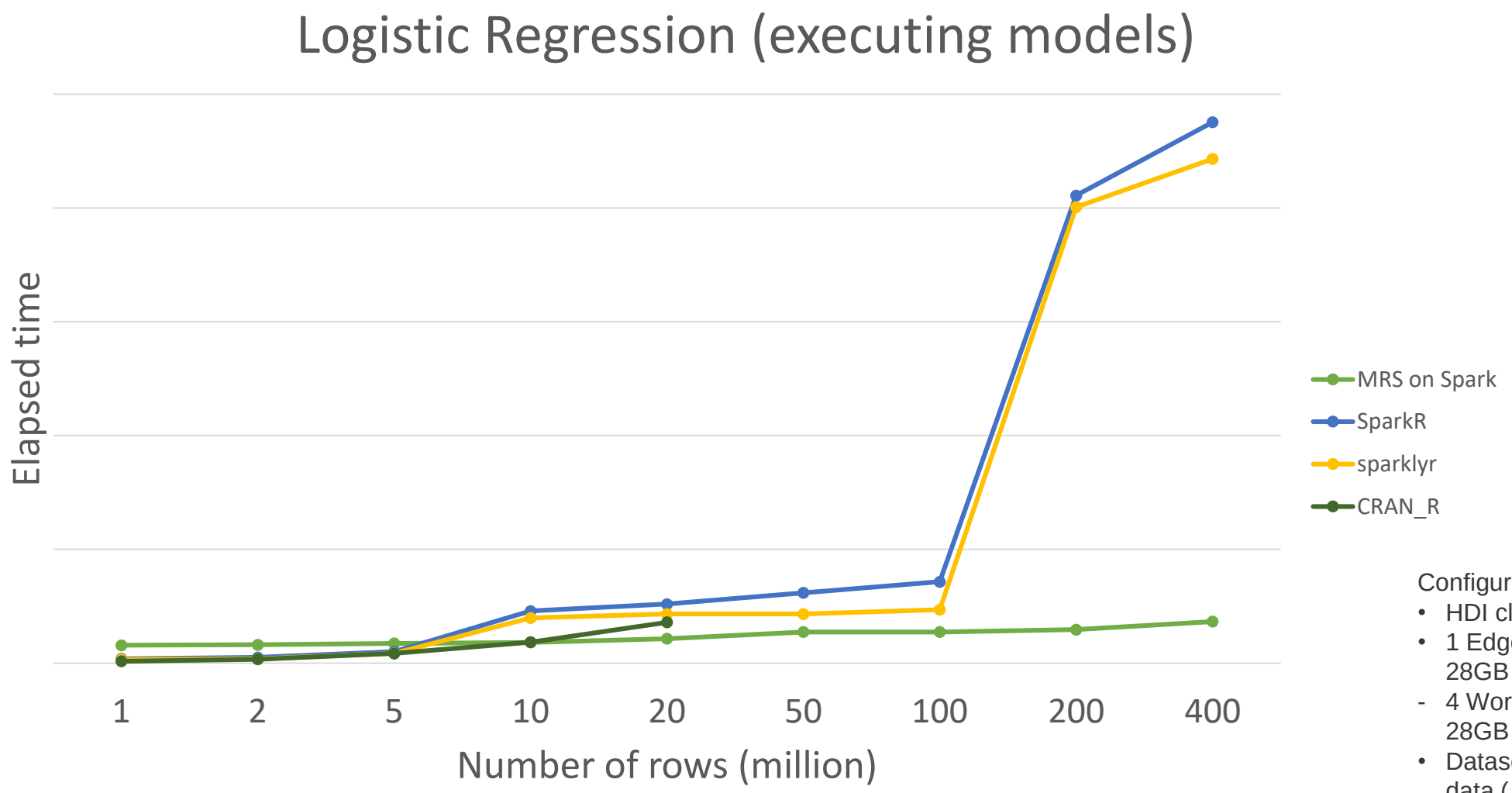


With HDInsights you can leverage SparkR and well as R Server.

Hadoop: lingua franca for BigData

- Spark (Standard)
  - Integrated notebooks experience
  - Upgraded to latest Version 1.6
- R Server (Premium)
  - Leverage R skills with massively scalable algorithms and statistical functions
  - Reuse existing R functions over multiple machines

R Server on Spark - substantially faster



Configuration:

- HDI cluster size: 7 nodes
- 1 Edge Node: 8 cores, 28GB
- 4 Worker Nodes: 8 cores, 28GB
- Dataset: Duplicated Airlines data (.csv)
- Number of columns: 26

Source: Scalable Data Analytics Using R: Single Machines to Spark Clusters, Microsoft presentation KDD 2016

There are 3 main steps to getting started with R on HDInsights

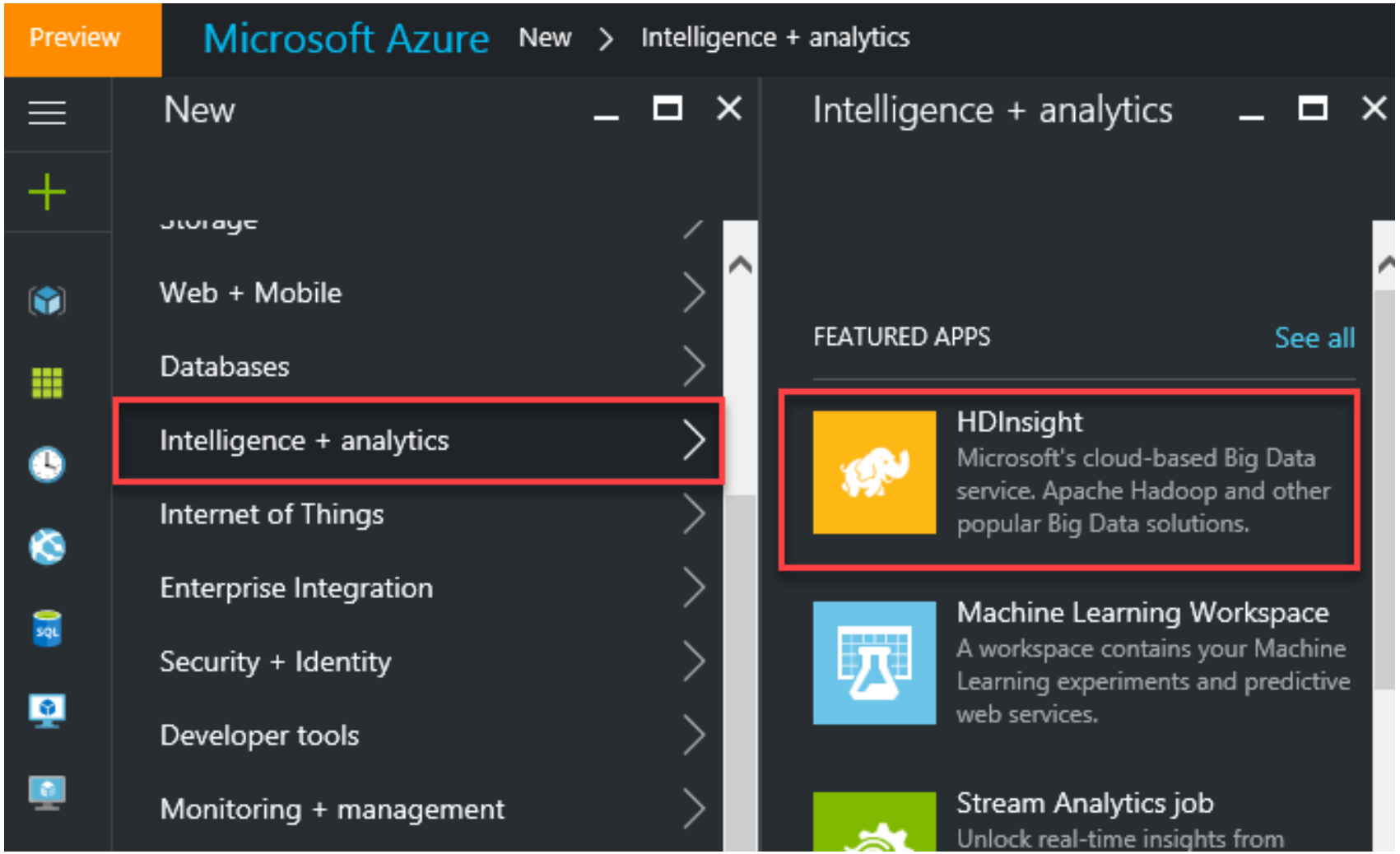
1. Provision a HDInsight cluster
2. Create a compute context
3. Run R code

The following slides will give you some guidance, code examples and useful links to get more details.

Provision an HDInsight cluster

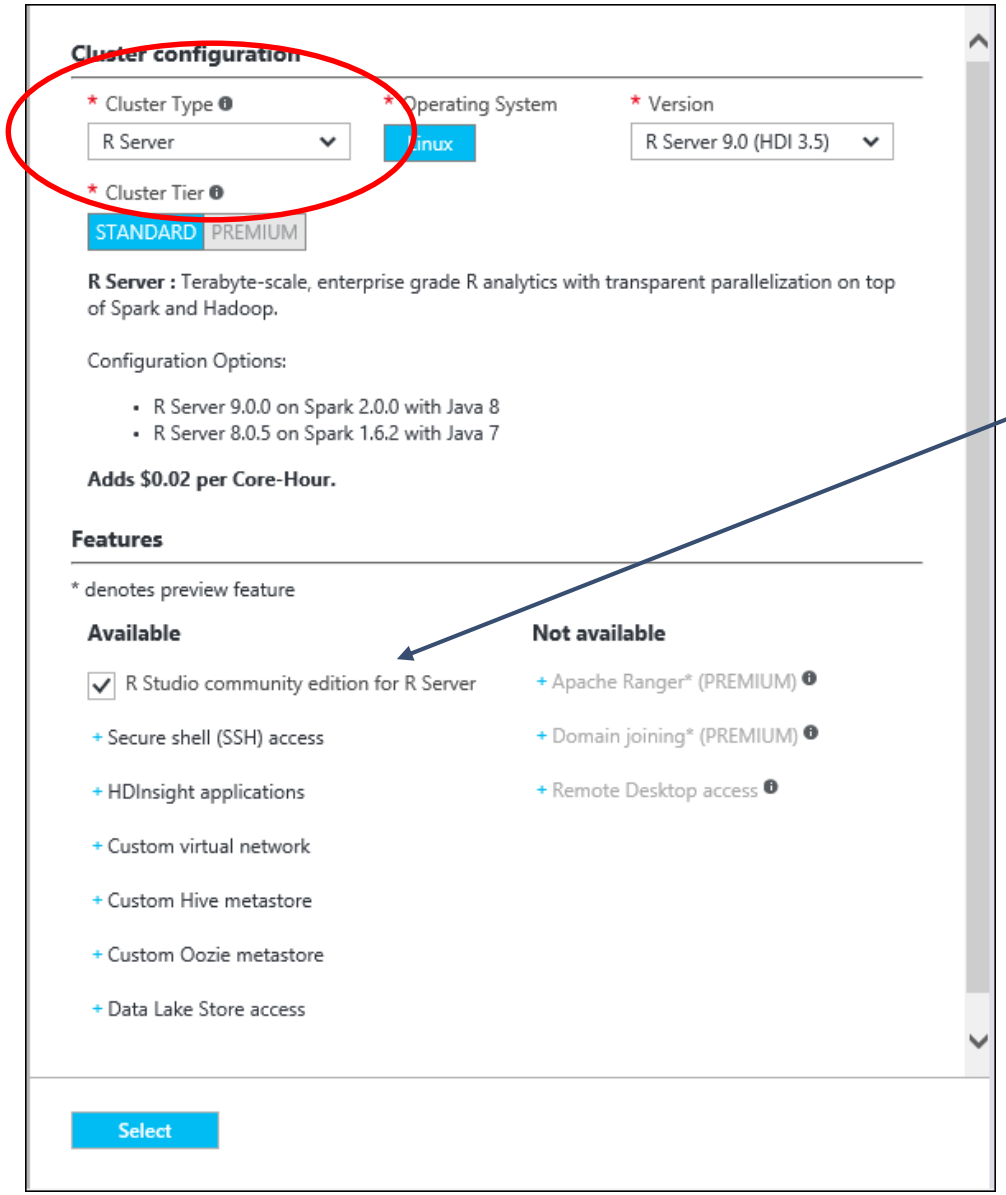
Run R scripts against data stored on a Hadoop file system or within Spark.

Provision a HDInsight cluster



Different types of clusters – select R Server

Provision a HDInsight cluster



RStudio Server enables you to provide a browser based interface to a version of R running on a remote Linux server. **R Studio community edition for R Server**: *this browser-based IDE* is installed by default on the edge/worker node. If you would prefer to not have it installed, then un-check the check box. If you choose to have it installed, then you'll find the URL for accessing the RStudio Server login on a portal application blade for your cluster once it's been created.

Create your compute context

Create a compute context 2

Connect to your cluster using R Studio Server on port 8787, e.g. http://<ip address>:8787/

- Hadoop MapReduce

  Create a compute context for Hadoop MapReduce

  a. Specify parameters:

```
myHadoopCluster <- RxHadoopMR(
                     hdfsShareDir = myHdfsShareDir
                   , shareDir     = myShareDir
                   , sshUsername  = mySshUsername
                   , sshHostname  = mySshHostname
                   , sshSwitches  = mySshSwitches)
```

  b. Specify context:

```
rxSetComputeContext(myHadoopCluster)
```

Note: there are default values for *shareDir* and *hdfsShareDir,* if you're running a node of the cluster using these defaults you don't need to specify them. You also don't need the host and user name if you're already connected to a node on the cluster. See the link above for more detail on how to use R client to remotely connect to a Spark/Hadoop cluster.

Create your compute context

- Spark

  [Create a compute context for Spark]

  a. Specify parameters:

```
myHadoopCluster <- RxSpark(
                       hdfsShareDir = myHdfsShareDir
                       , shareDir = myShareDir
                       , sshUsername = mySshUsername
                       , sshHostname = mySshHostname
                       , sshSwitches = mySshSwitches)
```

  b. Specify context:

```
rxSetComputeContext(myHadoopCluster)
```

Note: there are default values for shareDir and hdfsShareDir, if you're running a node of the cluster using these defaults you don't need to specify them. You also don't need the host and user name if you're already connected to a node on the cluster. See this link for more detail on how to use R client to remotely connect to a Spark/Hadoop cluster.

https://msdn.microsoft.com/microsoft-r/scaler-spark-getting-started#creating-a-compute-context-for-spark

Run R code

- Copy data into the cluster

```
bigDataDirRoot <- "/share" # define hdfs location of the example data

source <-system.file("SampleData/AirlineDemoSmall.csv",
package="RevoScaleR") # create pointer to a sample data set

inputDir <- file.path(bigDataDirRoot,"AirlineDemoSmall") # create
pointer to a directory location

rxHadoopMakeDir(inputDir) # make directory in hdfs

rxHadoopCopyFromLocal(source, inputDir) # copy sample data into the hdfs
```

(all that to copy a csv and store it!)

- Define the data source, e.g. hdfs, for R Server

```
hdfsFS <- RxHdfsFileSystem()
```

Run R code

- Read data into R Server

```
airDS <- RxTextData(file = inputDir
        , missingValueString = "M”
        , fileSystem = hdfsFS)
```

- Summarize

```
adsSummary <- rxSummary(~ArrDelay+CRSDepTime+DayOfWeek
        , data = airDS)
```

Some function names begin with rx and others with Rx. The Rx function name prefix is used to distinguish the class constructors such as data sources and compute contexts.

Useful links

- Azure Data Science Virtual Machines
- R Server on Hadoop
- R Server Compute Context on Hadoop
  - Spark (https://msdn.microsoft.com/en-us/microsoft-r/scaler-spark-getting-started)
  - MapReduce (https://msdn.microsoft.com/en-us/microsoft-r/scaler-hadoop-getting-started)

Creating compute contexts

https://msdn.microsoft.com/en-us/microsoft-r/scaler-hadoop-getting-started#create-a-compute-context-for-Hadoop-MapReduce

https://msdn.microsoft.com/microsoft-r/scaler-spark-getting-started#creating-a-compute-context-for-spark

Appendix

slalom

Example using sparklyr package

```
>library(sparklyr)
>sc <- spark_connect(master = "yarn-client")
>download.file(http://alizaidi.blob.core.windows.net/training/taxi_large.csv
                    , "taxi_large.csv")
>wasb_taxi <- "/NYCTaxi/sample"
>rxHadoopListFiles("/")
>rxHadoopMakeDir(wasb_taxi)
>rxHadoopCopyFromLocal("taxi_large.csv", wasb_taxi)
>rxHadoopCommand("fs -cat /NYCTaxi/sample/taxi_large.csv | head")
>taxi <- spark_read_csv(sc, path = wasb_taxi, "taxisample", header = TRUE)
```

Tutorial on R Server, Spark and SparkR