INTRODUCTION TO

APACHE SPARK

WHAT IS APACHE SPARK?

- https://databricks.com/spark/about
- http://spark.apache.org

RDD

- Immutable Distributed Collection of Objects.
- Ways to Create a RDD :
  - Use external Dataset
    - lines = sc.textFile(“filename”)
  - Parallelize a collection
    - lines = sc.parallelize([1,2,3,4,5,6])

TRANSFORMATION

- Always return new RDD's.
- Lazy Evaluation
- http://spark.apache.org/docs/latest/programming-guide.html#transformations
- x = sc.parallelize([1,2,3,4,5])
- x10 = x.map(lambda n : n*10)

Table 3-2. Basic RDD transformations on an RDD containing {1, 2, 3, 3}

Function name	Purpose	Example	Result
`map()`	Apply a function to each element in the RDD and return an RDD of the result.	`rdd.map(x => x + 1)`	`{2, 3, 4, 4}`
`flatMap()`	Apply a function to each element in the RDD and return an RDD of the contents of the iterators returned. Often used to extract words.	`rdd.flatMap(x => x.to(3))`	`{1, 2, 3, 2, 3, 3, 3}`
`filter()`	Return an RDD consisting of only elements that pass the condition passed to `filter()`.	`rdd.filter(x => x != 1)`	`{2, 3, 3}`
`distinct()`	Remove duplicates.	`rdd.distinct()`	`{1, 2, 3}`
`sample(withReplacement, fraction, [seed])`	Sample an RDD, with or without replacement.	`rdd.sample(false, 0.5)`	Nondeterministic

ACTIONS

- Actual Computation happens.
- http://spark.apache.org/docs/latest/programming-guide.html#actions

- sum = x10.reduce(lambda n,m : n+m)

Table 3-4. Basic actions on an RDD containing {1, 2, 3, 3}

Function name	Purpose	Example	Result
`collect()`	Return all elements from the RDD.	`rdd.collect()`	`{1, 2, 3, 3}`
`count()`	Number of elements in the RDD.	`rdd.count()`	`4`
`countByValue()`	Number of times each element occurs in the RDD.	`rdd.countByValue()`	`{(1, 1), (2, 1), (3, 2)}`

Function name	Purpose	Example	Result
`take(num)`	Return num elements from the RDD.	`rdd.take(2)`	`{1, 2}`
`top(num)`	Return the top num elements the RDD.	`rdd.top(2)`	`{3, 3}`
`takeOrdered(num)(ordering)`	Return num elements based on provided ordering.	`rdd.takeOrdered(2)(myOrdering)`	`{3, 3}`
`takeSample(withReplacement, num, [seed])`	Return num elements at random.	`rdd.takeSample(false, 1)`	Nondeterministic
`reduce(func)`	Combine the elements of the RDD together in parallel (e.g., sum).	`rdd.reduce((x, y) => x + y)`	9
`fold(zero)(func)`	Same as `reduce()` but with the provided zero value.	`rdd.fold(0)((x, y) => x + y)`	9

KEY VALUE PAIRS

- pairs = lines.map(lambda line: (line.split(“ ”)(0),x))

LOADING AND SAVING DATA

- Text Files :
  - sc.textFile(“file”) or sc.wholeTextFiles(“files”)
  - sc.saveAsTextFile(“outputFile”)