

Resilient Distributed Datasets (RDDs)

Resilient Distributed Datasets (RDDs)

Primary abstraction that allow Spark to distribute data

- Fault tolerant – if they are destroyed they can be recreated by the driver and sent to a new worker
- Immutable – once created you cannot change them. Instead you perform transformations on them and create new RDDs.

Resilient Distributed Datasets (RDDs)

You can create an RDD in one of two ways:

- *Parallelizing* an existing collection in your driver program
- Referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat

Resilient Distributed Datasets (RDDs)

Parallelizing an existing collection in your driver program

```
>>> data = [1, 2, 3, 4, 5]           //Python List  
>>> distData = sc.parallelize(data)   // RDD
```

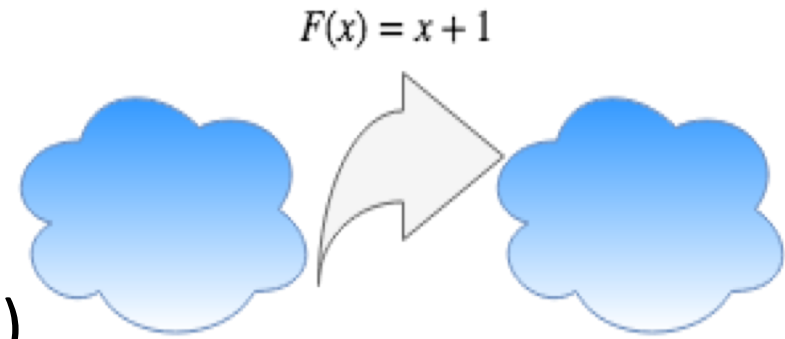
Referencing a dataset

```
>>> distFile = sc.textFile("data.txt") //RDD
```

Resilient Distributed Datasets (RDDs)

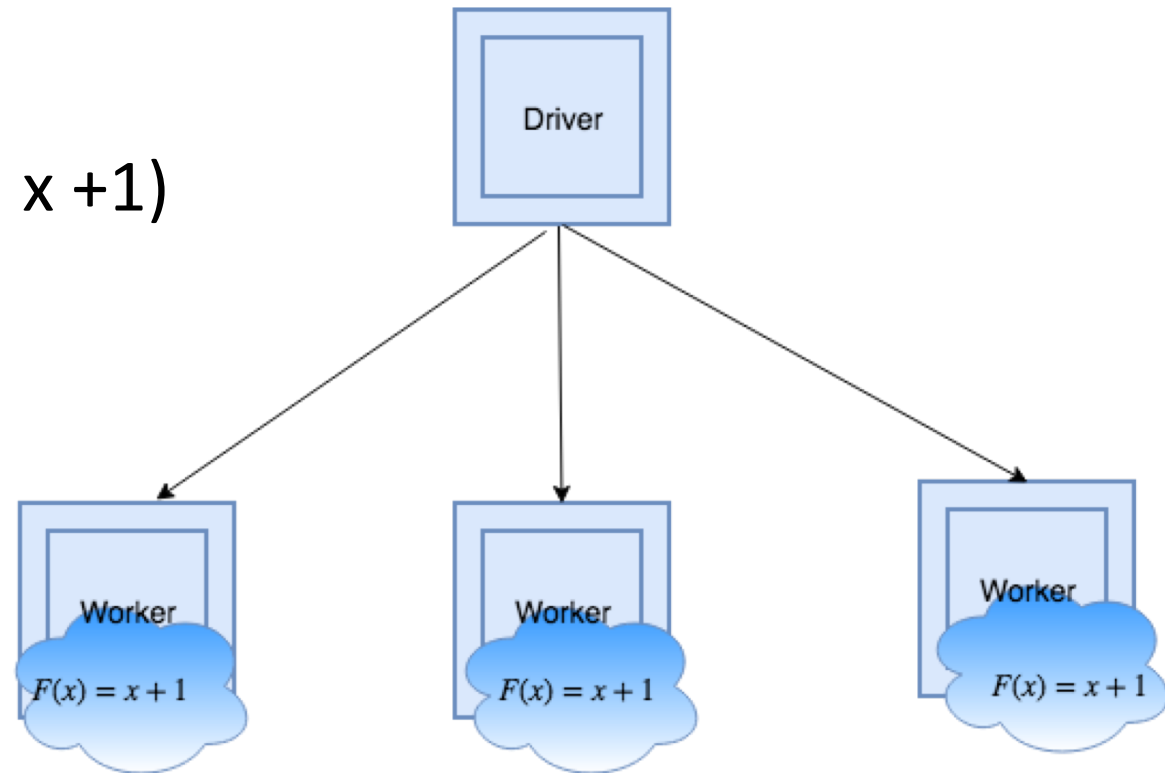
Given an RDD you can create a new RDD using ***transformations***.

```
>>> data = [1, 2, 3, 4, 5]
>>> rdd1 = sc.parallelize(data)
>>> rdd2 = rdd1.map(lambda x : x + 1)
```



Resilient Distributed Datasets (RDDs)

```
rdd2 = rdd1.map(lambda x : x + 1)
```



Resilient Distributed Datasets (RDDs)

Transformations on RDDs

map

flatMap

filter

fold

aggregate



Resilient Distributed Datasets (RDDs)

- To get the results of a transformation back to the driver, you must issue an ***action***

```
>>> data = [1, 2, 3, 4, 5]
```

```
>>> rdd1 = sc.parallelize(data)
```

```
>>> rdd2 = rdd1.map(lambda x : x + 1)
```

```
>>> rdd2.collect()
```


Resilient Distributed Datasets (RDDs)

```
In [1]: import pyspark
        sc = pyspark.SparkContext('local[*]')
```

```
In [2]: data = [1, 2, 3, 4, 5]
        rdd1 = sc.parallelize(data)
        rdd2 = rdd1.map(lambda x : x + 1)
        print(type(rdd2))
```

```
<class 'pyspark.rdd.PipelinedRDD'>
```

```
In [3]: mylist = rdd2.collect()
        print(type(mylist))
```

```
<class 'list'>
```

```
In [5]: print(mylist)
```

```
[2, 3, 4, 5, 6]
```

Resilient Distributed Datasets (RDDs)

- Actions
 - collect
 - count
 - reduce
 - take(n)

Reference : <https://spark.apache.org/docs/latest/rdd-programming-guide.html> - actions

Resilient Distributed Datasets (RDDs)

- Lazy Evaluation