Data Analysis using The Spark Shell

Once you have installed Spark, you can launch the Spark Shell from the commandline

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

This starts up an REPL for Scala

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

This REPL is just the same as any Scala REPL

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

The only difference is that you can also use Spark functions within this REPL

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

You can configure a Jupyter/ iPython notebook to use Spark Shell as a kernel

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

Use Apache Toree to configure Jupyter notebooks for Scala

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

This will give you an interactive environment where you can save your session and results as a notebook

```
> spark-shell
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

```
scala>
```

By default, this launches Spark in a local non-distributed mode

```
> spark-shell --master local[2]
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

By using the master option you can launch Spark in distributed mode and plug in different cluster managers

```
> spark-shell --master local[2]
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

This tells Spark to launch the shell with multiple threads on the local machine

```
> spark-shell --master local[2]
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

This is the number of threads (ideal is the number of cores on your machine)

```
> spark-shell --master yarn-client
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

If you have a Hadoop cluster running, you can plug in YARN as the cluster manager

```
> spark-shell --master yarn-client
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

When the shell is launched it initializes a SparkContext (sc)

```
> spark-shell --master yarn-client
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

The SparkContext represents a connection to the Spark Cluster

```
> spark-shell --master yarn-client
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

This can be used to load data into memory from a specified source

```
> spark-shell --master yarn-client
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

The SparkContext object is required to create Resilient Distributed Datasets(RDDs)

```
> spark-shell --master yarn-client
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

All our data analysis will be through operations on RDDs

```
> spark-shell --master yarn-client
```

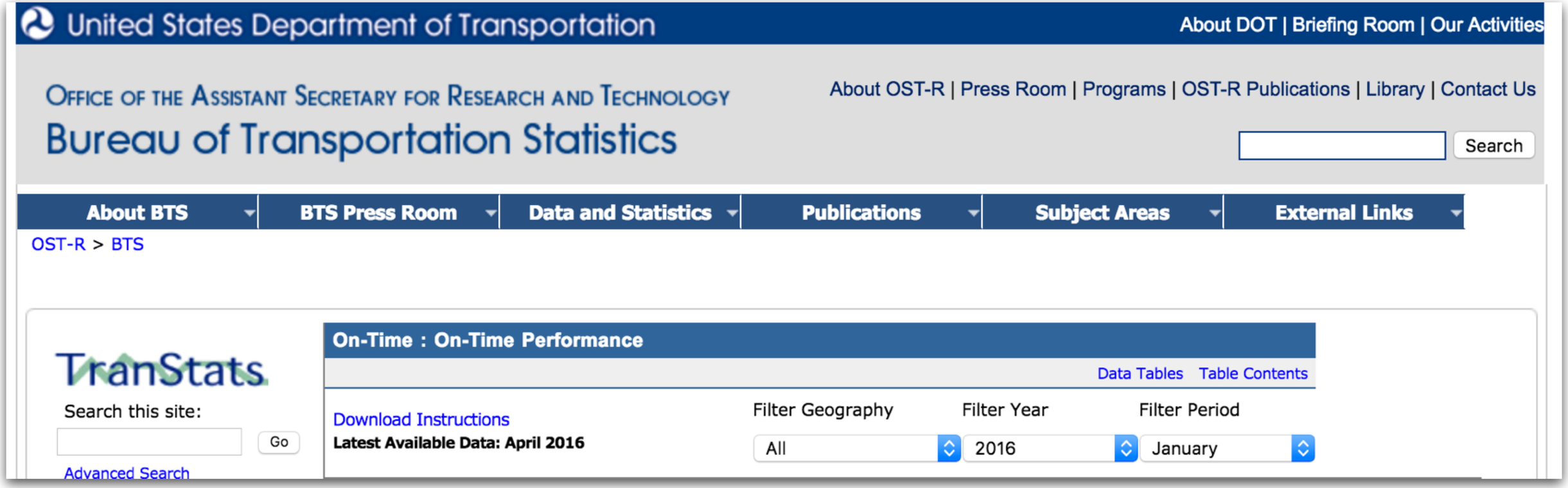
```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_60)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
```

Before we understand more about RDDs, let's see an example of data analysis using Spark and Scala

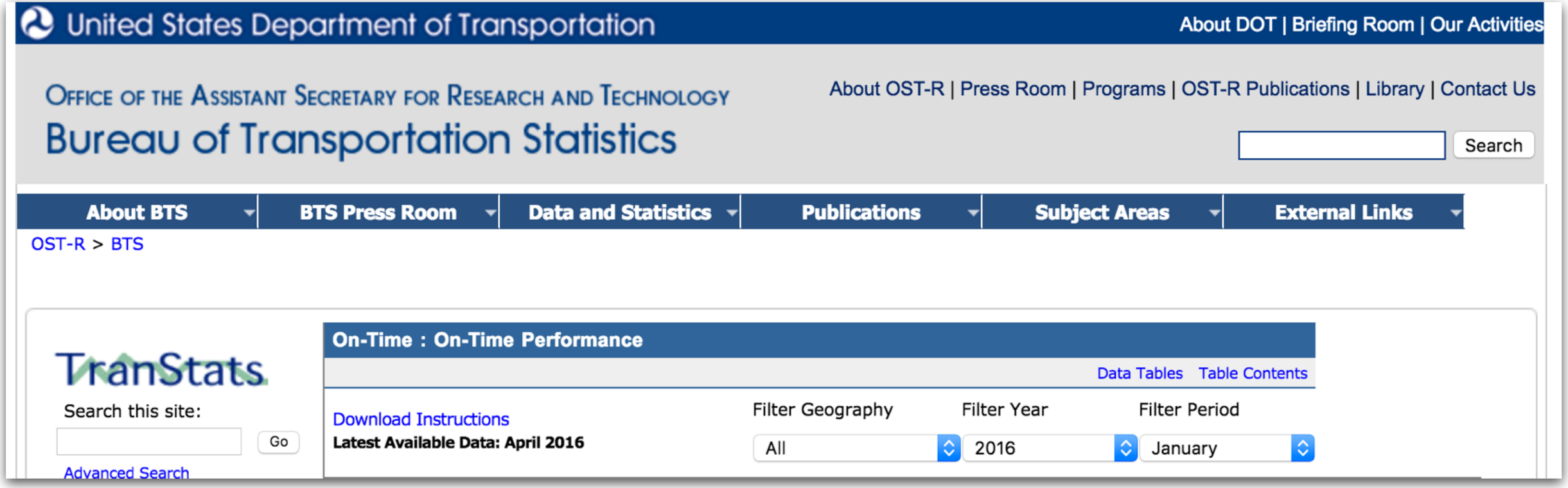
Exploring
Airline delays data in the Spark Shell

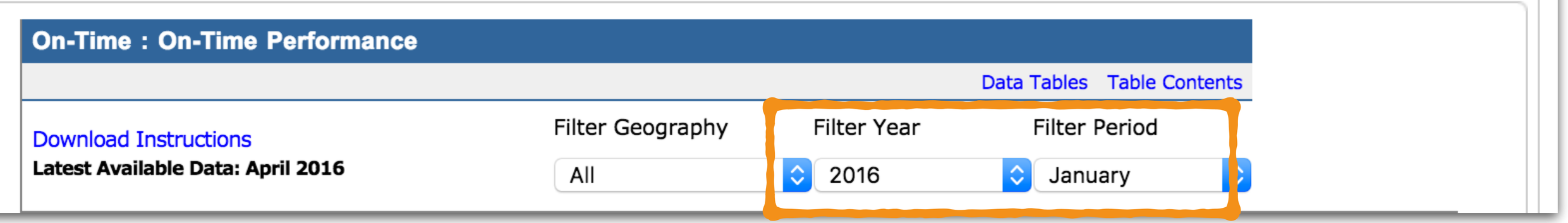
Part 1

The US department of transportation publishes Flight related information on their website



This data is free and available for anyone to analyze





From here we can get flight arrival and departure times, delays for all flights taking off in a certain period

There are 3 files

flights.csv	Flight id, airline, airport, departure, arrival , delay
airlines.csv	airline id, airline name
airports.csv	airport id, airport name

```
// Data location
val airlinesPath="hdfs:///user/swethakolalapudi/flightDelaysData/airlines.csv"
val airportsPath="hdfs:///user/swethakolalapudi/flightDelaysData/airports.csv"
val flightsPath="hdfs:///user/swethakolalapudi/flightDelaysData/flights.csv"
```

airlines.csv

airports.csv

flights.csv

Spark can read these files from the local file system or from HDFS

```
// Data location
val airlinesPath="hdfs:///user/swethakolalapudi/flightDelaysData/airlines.csv"
val airportsPath="hdfs:///user/swethakolalapudi/flightDelaysData/airports.csv"
val flightsPath="hdfs:///user/swethakolalapudi/flightDelaysData/flights.csv"
```

airlines.csv

airports.csv

flights.csv

If the files are in HDFS, you need to precede the file path with hdfs:///

```
// Data location
val airlinesPath="hdfs:///user/swethakolalapudi/flightDelaysData/airlines.csv"
val airportsPath="hdfs:///user/swethakolalapudi/flightDelaysData/airports.csv"
val flightsPath="hdfs:///user/swethakolalapudi/flightDelaysData/flights.csv"
```

airlines.csv

airports.csv

flights.csv

Now we can use the SparkContext initialized by Spark to load the data

```
// Load one dataset
val airlines=sc.textFile(airlinesPath)
```

```
// Load one dataset
val airlines=sc.textFile(airlinesPath)
```

This is the SparkContext variable
i.e. the connection to Spark

```
// Load one dataset
val airlines=sc.textFile(airlinesPath)
```

We are telling SparkContext to load data from a text file into memory

```
// Load one dataset
val airlines=sc.textFile(airlinesPath)
```

The path to the text file, either on local or in HDFS

```
// Load one dataset
val airlines=sc.textFile(airlinesPath)
```

airlines is a Resilient Distributed Dataset (RDD)

```
airlines
```

```
hdfs:///user/swethakolalapudi/flightDelaysData/airlines.csv MapPartitionsRDD[2] at textFile at <console>:23
```

Resilient Distributed Dataset (RDD)

```
irlines

dfs:///user/swethakolalapudi/flightDelaysData/airlines.csv MapPartitionsRDD[2] at textFile at <console>:23
```

We have created our first RDD!

```
airlines
```

```
hdfs:///user/swethakolalapudi/flightDelaysData/airlines.csv MapPartitionsRDD[2] at textFile at <console>:23
```

Let's get a quick glimpse of the data from the RDD

```
// Get the first line
airlines.first()
```

```
Code,Description
```

A quick glimpse of the data

```
// Get the first line
airlines.first()
```

```
Code,Description
```

```
// View a few lines
airlines.take(10)
```

```
Array(Code,Description, "19031","Mackey International Inc.: MAC", "19032","Munz Northern Airlines Inc.: XY", "1903
3","Cochise Airlines Inc.: COC", "19034","Golden Gate Airlines Inc.: GSA", "19035","Aeromech Inc.: RZZ", "19036","Gol
den West Airlines Co.: GLW", "19037","Puerto Rico Intl Airlines: PRN", "19038","Air America Inc.: STZ", "19039","Swif
t Aire Lines Inc.: SWT")
```

A quick glimpse of the data

```
// View a few lines
airlines.take(10)
```

```
Array(Code,Description, "19031","Mackey International Inc.: MAC", "19032","Munz Northern Airlines Inc.: XY", "1903
3","Cochise Airlines Inc.: COC", "19034","Golden Gate Airlines Inc.: GSA", "19035","Aeromech Inc.: RZZ", "19036","Gol
den West Airlines Co.: GLW", "19037","Puerto Rico Intl Airlines: PRN", "19038","Air America Inc.: STZ", "19039","Swif
t Aire Lines Inc.: SWT")
```

```
// View the entire dataset
airlines.collect()
```

```
Array(Code,Description, "19031","Mackey International Inc.: MAC", "19032","Munz Northern Airlines Inc.: XY", "1903
3","Cochise Airlines Inc.: COC", "19034","Golden Gate Airlines Inc.: GSA", "19035","Aeromech Inc.: RZZ", "19036","Gol
den West Airlines Co.: GLW", "19037","Puerto Rico Intl Airlines: PRN", "19038","Air America Inc.: STZ", "19039","Swif
t Aire Lines Inc.: SWT", "19040","American Central Airlines: TSF", "19041","Valdez Airlines: VEZ", "19042","Southeast
 Alaska Airlines: WEB", "19043","Altair Airlines Inc.: AAR", "19044","Chitina Air Service: CHI", "19045","Marco Islan
d Airways Inc.: MRC", "19046","Caribbean Air Services Inc.: OHZ", "19047","Sundance Airlines: PRO", "19048","Seair Al
aska Airlines Inc.: SAI", "19049","Southeast Airlines Inc.: SLZ", "19050","A...
```

Array(Code,Description, "19031","Mackey International Inc.: MAC", "19032","Munz Northern Airlines Inc
3","Cochise Airlines Inc.: COC", "19034","Golden Gate Airlines Inc.: GSA", "19035","Aeromech Inc.: RZ
den West Airlines Co.: GLW", "19037","Puerto Rico Intl Airlines: PRN", "19038","Air America Inc.: STZ
t Aire Lines Inc.: SWT", "19040","American Central Airlines: TSF", "19041","Valdez Airlines: VEZ", "1
 Alaska Airlines: WEB", "19043","Altair Airlines Inc.: AAR", "19044","Chitina Air Service: CHI", "190
d Airways Inc.: MRC", "19046","Caribbean Air Services Inc.: OHZ", "19047","Sundance Airlines: PRO", "
aska Airlines Inc.: SAI", "19049","Southeast Airlines Inc.: SLZ", "19050","A...

Each line consists of an airline code and description

```
Array(Code,Description, "19031","Mackey International Inc.: MAC", "19032","Munz Northern Airlines Inc.: XY", "1903
3","Cochise Airlines Inc.: COC", "19034","Golden Gate Airlines Inc.: GSA", "19035","Aeromech Inc.: RZZ", "19036","Gol
den West Airlines Co.: GLW", "19037","Puerto Rico Intl Airlines: PRN", "19038","Air America Inc.: STZ", "19039","Swif
t Aire Lines Inc.: SWT", "19040","American Central Airlines: TSF", "19041","Valdez Airlines: VEZ", "19042","Southeast
 Alaska Airlines: WEB", "19043","Altair Airlines Inc.: AAR", "19044","Chitina Air Service: CHI", "19045","Marco Islan
d Airways Inc.: MRC", "19046","Caribbean Air Services Inc.: OHZ", "19047","Sundance Airlines: PRO", "19048","Seair Al
aska Airlines Inc.: SAI", "19049","Southeast Airlines Inc.: SLZ", "19050","A...
```

Here's how we can filter out the header row

```
airlines.filter(x => !x.contains("Description"))
```

```
airlines.filter(x => !x.contains("Description"))
```

The filter operation takes a function that returns a Boolean

It calls that function for each record in the RDD and only keeps records that return True

```
airlines.filter(x => !x.contains("Description"))
```

The function can be anything!

```
airlines.filter(x => !x.contains("Description"))
```

You can pass in any function as long as the function input type matches the type of records of the RDD

```
airlines.filter(x => !x.contains("Description"))
```

In this case, the function should work for Strings

```
airlines.filter(x => !x.contains("Description"))
```

This particular syntax is a way to define functions on the fly

```
airlines.filter(x => !x.contains("Description"))
```

The function is usually applied to each member of a collection (List, Array, RDD etc)

```
airlines.filter(x => !x.contains("Description"))
```

This represents each record in the RDD

```
airlines.filter(x => !x.contains("Description"))
```

Since we loaded the data from a textFile, each record is a String

```
airlines.filter(x => !x.contains("Description"))
```

This designates that the expression(s) on the right has to be evaluated for each x

```
airlines.filter(x => !x.contains("Description")
```

This is the expression that's evaluated for every x

```
airlines.filter(x => !x.contains("Description")
```

For a filter operation this expression must always return a Boolean

```
airlines.filter(x => !x.contains("Description"))
```

We can use contains because all String objects have this method

```
airlines.filter(x => !x.contains("Description"))
```

```
MapPartitionsRDD[3] at filter at <console>:25
```

Something interesting happens when we try to execute this operation

```
airlines.filter(x => !x.contains("Description"))
```

```
MapPartitionsRDD[3] at filter at <console>:25
```

Unlike the results of first(), take() or collect()

```
airlines.filter(x => !x.contains("Description"))
```

```
MapPartitionsRDD[3] at filter at <console>:25
```

Unlike the results of first(), take() or collect()

```
// View a few lines
airlines.take(10)
```

```
Array(Code,Description, "19031","Mackey I
3","Cochise Airlines Inc.: COC", "19034",
den West Airlines Co.: GLW", "19037","Pue
t Aire Lines Inc.: SWT")
```

```
airlines.filter(x => !x.contains("Description"))
```

```
MapPartitionsRDD[3] at filter at <console>:25
```

The filter operation returns another RDD

```
airlines.filter(x => !x.contains("Description"))
```

```
MapPartitionsRDD[3] at filter at <console>:25
```

This is the key to understanding RDDs and Spark's core functionality

```
filter
```

The filter operation tells Spark to change the dataset in some way

Transformation

```
take(10)
```

The take operation actually asks for a result - ten rows from the dataset

Action

Transformation

```
filter
```

Action

```
take(10)
```

All operations on RDDs are either Transformations or Actions

Transformation

```
filter
```

Action

```
take(10)
```

Spark will not immediately execute a transformation

Transformation

Action

```
filter
```

```
take(10)
```

It will just make a record of this transformation request by creating a new RDD

Transformation

Action

```
filter
```

```
take(10)
```

The user may define a chain of transformations on the dataset

Transformation

Action

```
filter
```

```
take(10)
```

In the airline example:

1. Filter the header
2. Split by comma
3. Convert airline code to integer

Transformation

Action

```
filter
```

```
take(10)
```

Spark will wait until a result is requested before executing any of these transformations

Transformation

```
filter
```

Action

```
take(10)
```

When created, an RDD just holds metadata

1. A transformation
2. It's parent RDD

AirlineFiltered RDD

Transformation

filter

Action

take(10)

AirlineFiltered RDD

filter

airlines

This way, every RDD knows where it came from

ie. RDDs know their lineage

Transformation

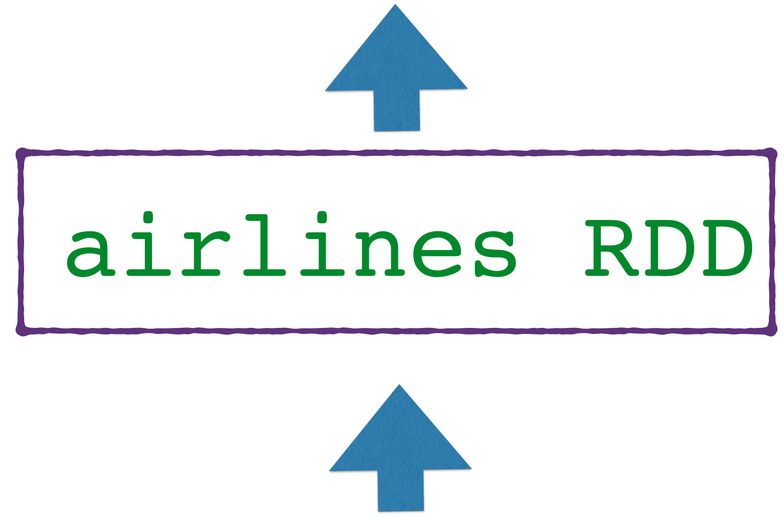
Action

```
take(10)
```

This lineage can be traced back to the original file that held the data

Recall: Reading a file created an RDD

AirlineFiltered RDD



Transformation

Action

filter

take(10)

File read is also like a transformation

RDDs are materialized only when a result is requested

Transformation

filter

Action

```
take(10)
```

A result is requested

using an action

Transformation

Action

```
filter
```

```
take(10)
```

Ex: the first 10 rows
a count
a sum
the entire dataset

Transformation

```
filter
```

Action

```
take(10)
```

To summarize, data is processed only when the user requests a result

Transformation

```
filter
```

Action

```
take(10)
```

This is known as

Lazy Evaluation

Lazy Evaluation

This is what allows Spark to perform **fast computations** on large amounts of data

Lazy Evaluation

Spark will keep a record of the series of transformations requested by the user

Lazy Evaluation

When called upon to execute,
Spark takes these operations
and groups them in
an efficient way

More on this later...