B. Explore Scala-Spark Variables

#1. In a terminal window, start the Scala Spark Shell: $ spark2-shell

```
[hk2874@login-2-1 ~]$ spark2-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLeve
l(newLevel).
Spark context Web UI available at http://login-2-1.local:4040
Spark context available as 'sc' (master = yarn, app id = application_16043463923
76_2093).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.3.0.cloudera4
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_162)
Type in expressions to have them evaluated.
Type :help for more information.
```

#2-3

- Create an immutable variable named exchangeRate with explicit type Double and assign to it the value 0.55. val exchangeRate:Double = 0.55
- Create an immutable variable named dollars with explicit type Int and assign to it the value 100.00. val dollars: Int = 100.00

```
[scala> val exchageRate=0.55
exchageRate: Double = 0.55

[scala> val dollars: Int=100.00
<console>:23: error: type mismatch;
 found   : Double(100.0)
 required: Int
       val dollars: Int=100.00
                        ^
```

#4- 8

- Correct step 3 to get rid of the error. val dollars: Double = 100.00
- Create a mutable variable named euros with implicit type Double initialized to zero. var euros = 0.0
- Assign to 'euros' the result of converting dollars to euros using exchangeRate as the conversion factor. euros = dollars * exchangeRate
- Assign to dollars a new value: 500 dollars = 500
- Note the error in step 7. Fix the error in step 7 and set dollars to 500. var dollars = 500

```
scala> val dollars: Double = 100.00
dollars: Double = 100.0

scala> var euros = 0.0
euros: Double = 0.0

scala> euros = dollars * exchangeRate
euros: Double = 55.00000000000001

scala> dollars = 500
<console>:25: error: reassignment to val
       dollars = 500
               ^

scala> dollars=500
<console>:25: error: reassignment to val
       dollars=500
              ^

scala> var dollars = 500
dollars: Int = 500

scala> dollars = 500.00
<console>:25: error: type mismatch;
 found   : Double(500.0)
 required: Int
       dollars = 500.00
                 ^
```

#9-13

- Now set dollars to 500.00. You should see an error because dollars expects an Int, not a Double. dollars = 500.00
- Create a new mutable variable, eurosInt, of type Int and assign to it 0. var eurosInt: Int = 0
- Assign to eurosInt the result of converting dollars to euros using exchangeRate. eurosInt = dollars * exchangeRate
- Use toInt with exchangeRate to remove the error in step 11. eurosInt = dollars * exchangeRate.toInt
- What is the result in step 12? Is it a useful result? What happened? Result is 0, this is not what we want.

```
scala> dollars = 500.00
<console>:25: error: type mismatch;
 found   : Double(500.0)
 required: Int
       dollars = 500.00
                 ^

scala> var eurosInt: Int = 0
eurosInt: Int = 0

scala> eurosInt = dollars * exchangeRate
<console>:29: error: type mismatch;
 found   : Int
 required: ?{def *(x$1: ? >: Double): ?}
Note that implicit conversions are not applicable because they are ambiguous:
 both method int2long in object Int of type (x: Int)Long
 and method int2float in object Int of type (x: Int)Float
 are possible conversion functions from Int to ?{def *(x$1: ? >: Double): ?}
       eurosInt = dollars * exchangeRate
                  ^
<console>:29: error: overloaded method value * with alternatives:
  (x: Int)Int <and>
  (x: Char)Int <and>
  (x: Short)Int <and>
  (x: Byte)Int
 cannot be applied to (Double)
       eurosInt = dollars * exchangeRate
                            ^

scala> eurosInt = dollars * exchangeRate.toInt
eurosInt: Int = 0
```

C. Explore Scala-Spark Computation

In the previous exercise you worked with integers and doubles. Notice that using toInt may not give the expected result.

#1. Try using toInt in a different way to achieve the desired eurosInt result of 275.
eurosInt = (dollars * exchangeRate).toInt

```
scala> eurosInt = (dollars * exchangeRate).toInt
eurosInt: Int = 275
```

#2. Use getClass to verify the types of the three variables. eurosInt.getClass dollars.getClass exchangeRate.getClass

```
scala> eurosInt.getClass; dollars.getClass; exchangeRate.getClass;
res0: Class[Double] = double
```

#3. Output the result using the println command: println("FA19 - $" + dollars + " = " + eurosInt + " Euros")

```
scala> println("FA19 - $" + dollars + " = " + eurosInt + " Euros")
FA19 - $500 = 275 Euros
```

#4. Enter: 27/3.0 and note the result variable name, e.g. res3. Use the result variable in an expression: res3 * 2

```
scala> 27/3.0
res3: Double = 9.0


scala> res3 * 2
res5: Double = 18.0
```

#5. Assign the value 22.5 to res3 - why didn't this work? Because result variables are immutable.

```
scala> res3 =22.5
<console>:25: error: reassignment to val
       res3 =22.5
```

#6-7.

- Import scala.math.pow and raise 2 to the third power. import scala.math.pow pow(2, 3)
- Import scala.math.sqrt and take the sqrt (square root) of 64. import scala.math.sqrt sqrt(64)

```
scala> import scala.math.pow
import scala.math.pow

scala> pow(2, 3)
res6: Double = 8.0

scala> import scala.math.sqrt
import scala.math.sqrt

scala> sqrt(64)
res7: Double = 8.0
```

D. Explore Scala-Spark Strings

#1-4

- Create an immutable variable called record and assign to it the following string:
  2017-01-08:10:00:00, 12345678-aaaa-1000-gggg-000111222333, 58, TRUE, enabled, disabled, 37.819722,-122.478611
- Use record.length to determine the number of characters in record.
  record.length (answer is 109)
- Use the contains method to search for the word "disabled" in record: record.contains("search term")
  record.contains("disabled") (answer is true)
- Use indexOf to find the index of the first occurrence of "17" in record.
  record.indexOf("17") (answer is 2)

```
scala> val record = "2017-01-08:10:00:00, 12345678-aaaa-1000-gggg-000111222333,
58, TRUE,enabled, disabled, 37.819722, -122.478611"
record: String = 2017-01-08:10:00:00, 12345678-aaaa-1000-gggg-000111222333, 58,
TRUE,enabled, disabled, 37.819722, -122.478611

scala> record.length
res0: Int = 109

scala> record.contains("disabled")
res1: Boolean = true

scala> record.indexOf("17")
res2: Int = 2
```

#5-6

- Convert record to lower case using toLowerCase and then use chaining with indexOf to find the start of substring "true". record.toLowerCase.indexOf("true") res83: Int = 63
- Verify that step 5. did not modify the variable named record. record (should show some upper case characters)

```
scala> record.toLowerCase.indexOf("true")
res3: Int = 63

scala> record
res4: String = 2017-01-08:10:00:00, 12345678-aaaa-1000-gggg-000111222333, 58, TR
UE,enabled, disabled, 37.819722, -122.478611
```

#7-10

- Create a new variable called record2 and assign to it the contents of record. var record2 = record
- Test whether record == record2 record == record2 (answer is true)
- Set record2 = "no match" record2 = "no match"
- Test whether record == record2 record == record2 (answer is false)

```
scala> var record2 = record
record2: String = 2017-01-08:10:00:00, 12345678-aaaa-1000-gggg-00011122233
led, disabled, 37.819722, -122.478611

scala> record == record2
res6: Boolean = true

scala> record2 = "no match"
record2: String = no match

scala> record == record2
res7: Boolean = false
```

-