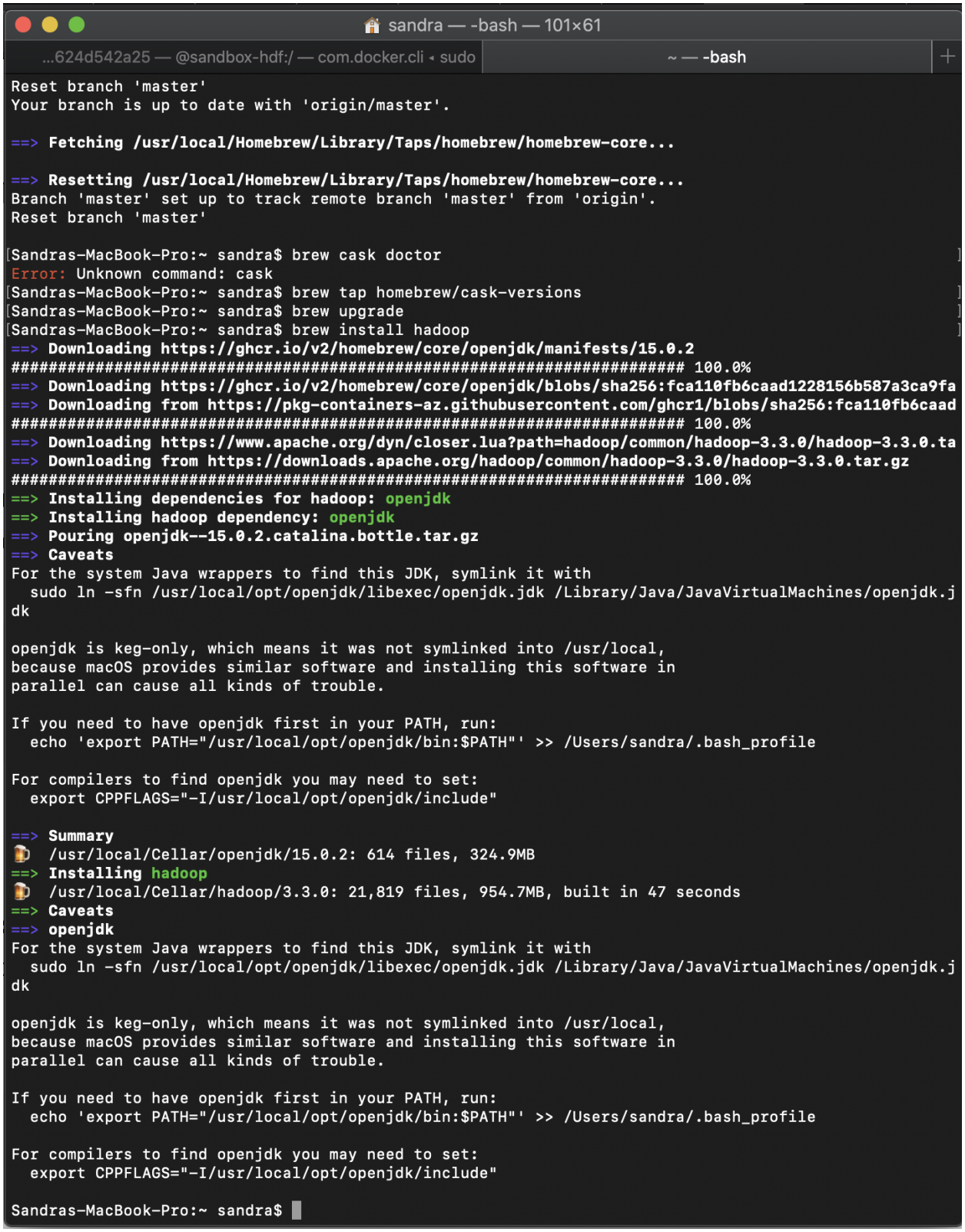
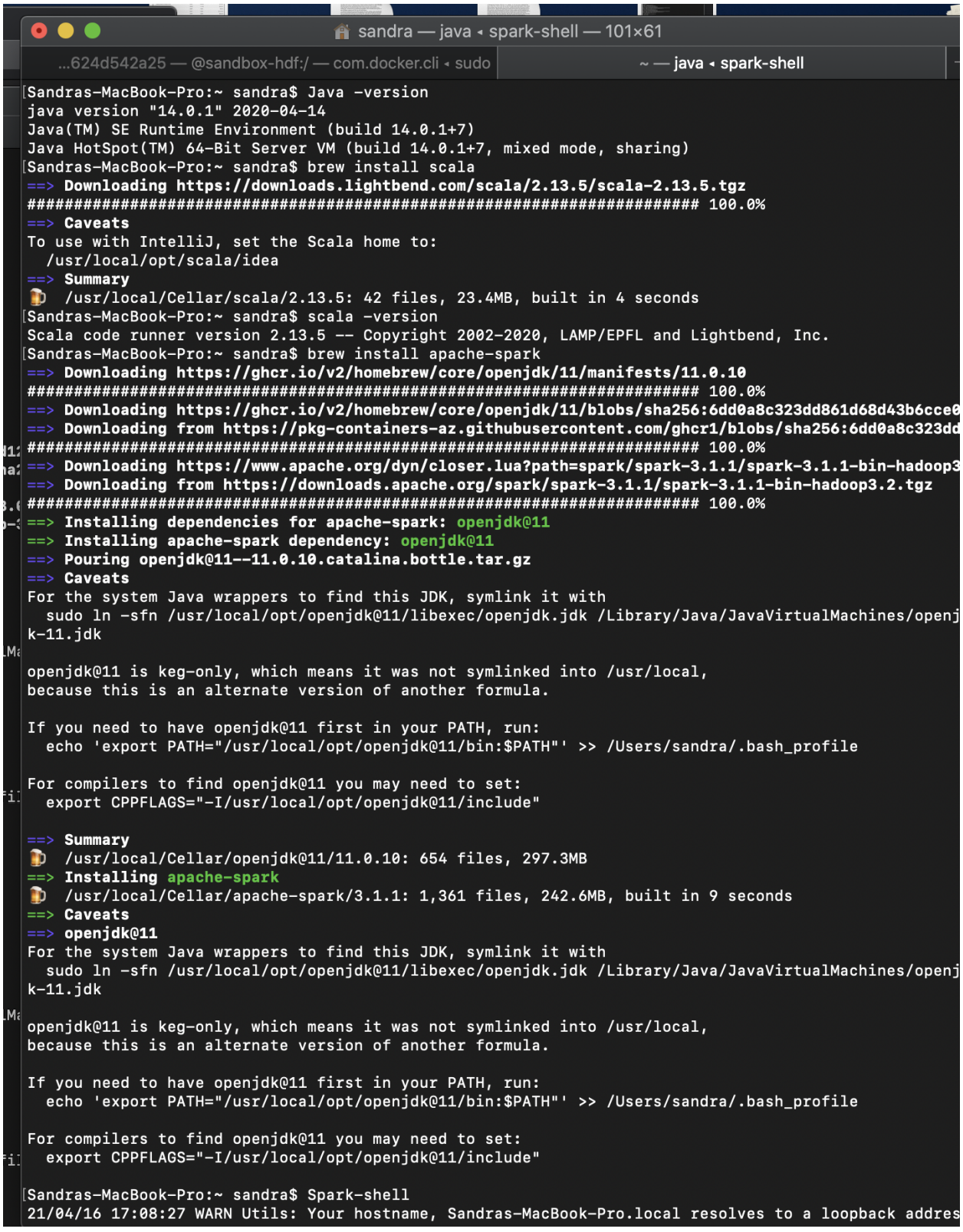
Chapter4 Project to be done in local machine - all steps taken

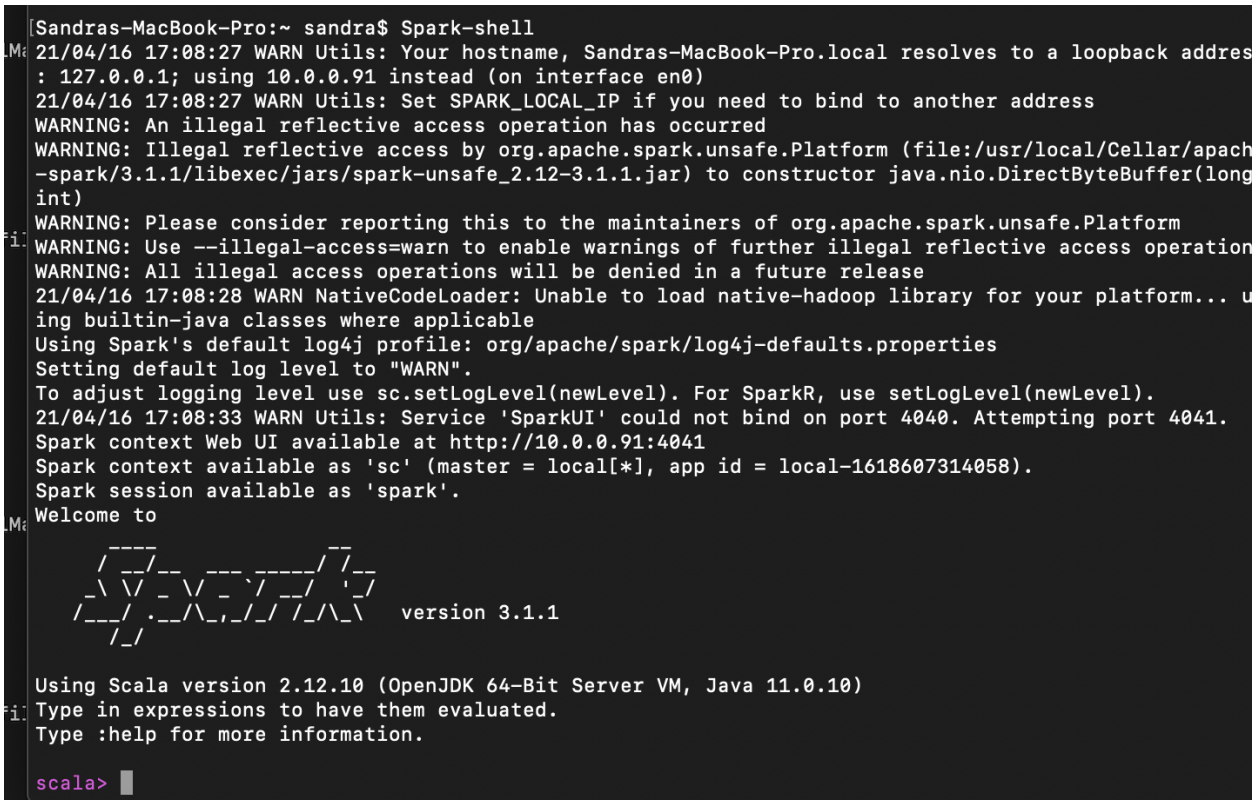
Installing hadoop local machine



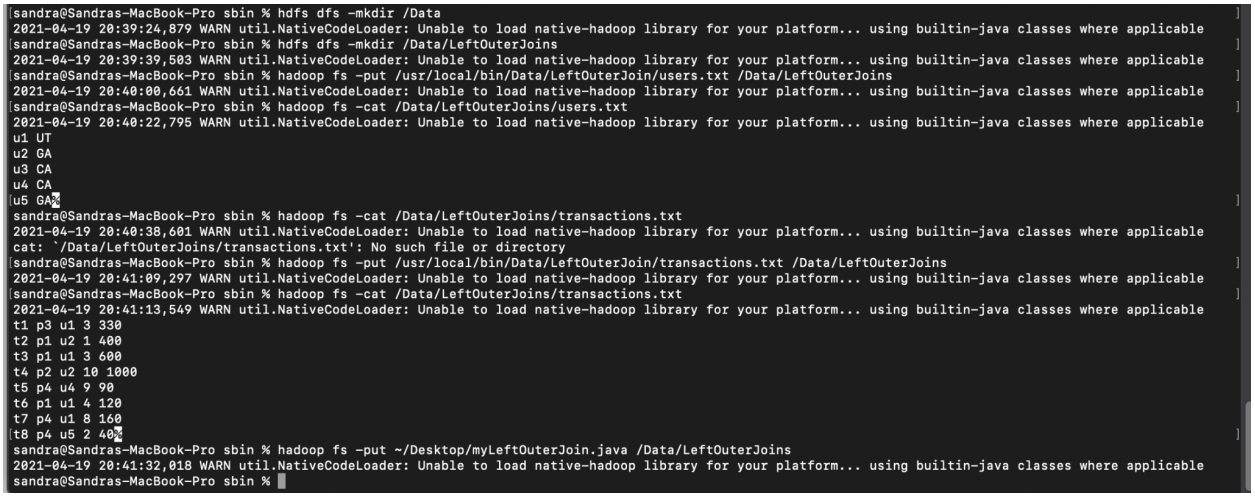
Installing spark in local machine



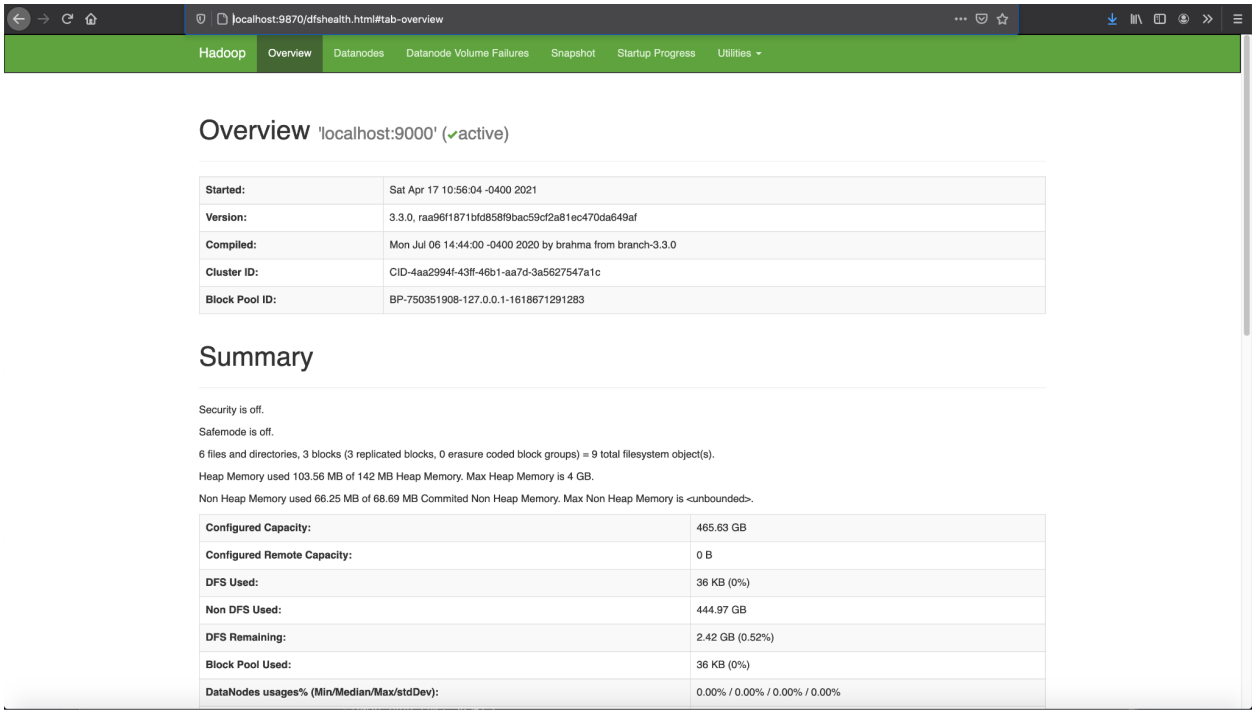
Accessing Spark local machine



Inserting Data in hadoop



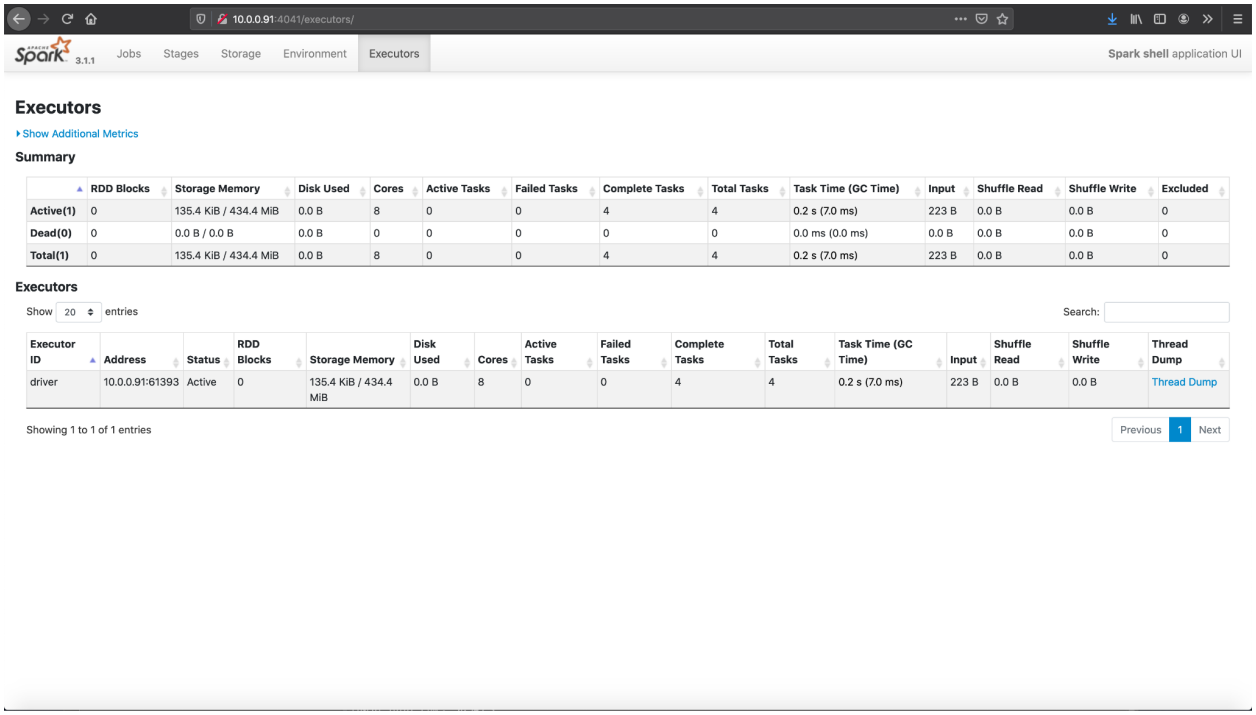
Hadoop localhost active

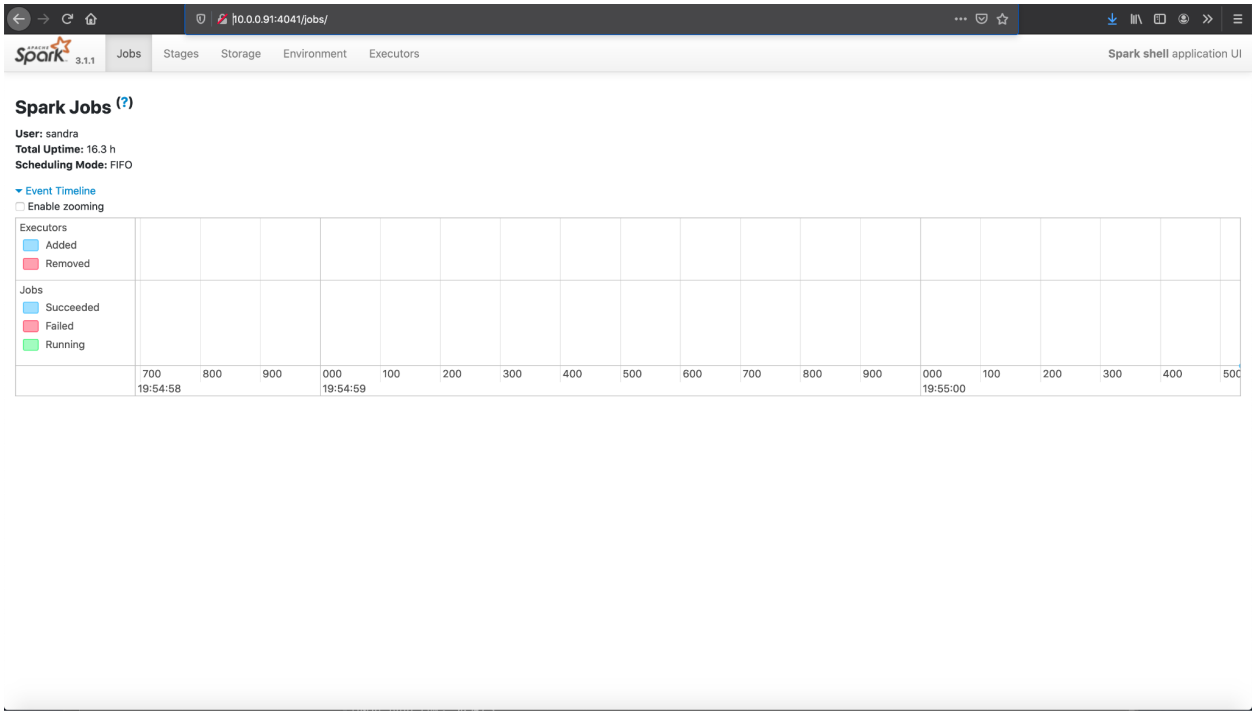


Inserting into spark scala

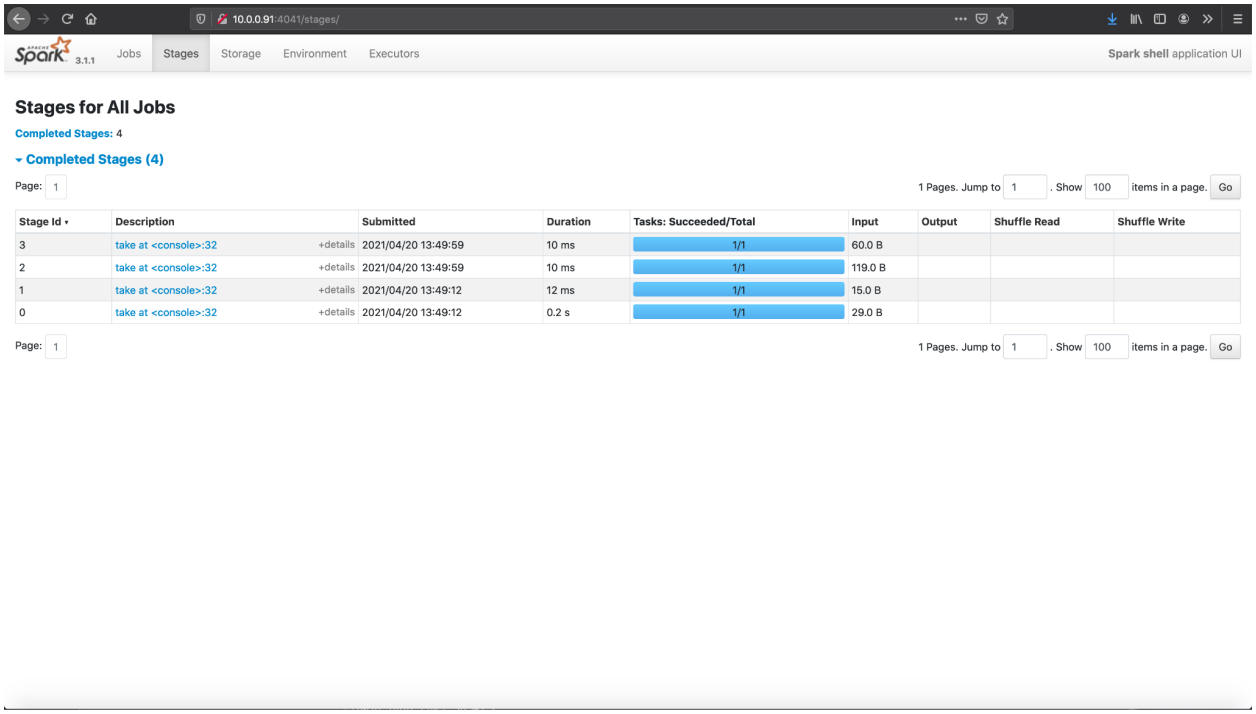
```
[scala> sc.textFile("users.txt")
res20: org.apache.spark.rdd.RDD[String] = users.txt MapPartitionsRDD[7] at textFile at <console>:31

[scala> res20.take(5)
res21: Array[String] = Array(u1 UT, u2 GA, u3 CA, u4 CA, u5 GA)

[scala> sc.textFile("transactions.txt")
res22: org.apache.spark.rdd.RDD[String] = transactions.txt MapPartitionsRDD[9] at textFile at <console>:31

[scala> res22.take(8)
res23: Array[String] = Array(t1 p3 u1 3 330, t2 p1 u2 1 400, t3 p1 u1 3 600, t4 p2 u2 10 1000, t5 p4 u4 9 90, t6 p1 u1 4 120, t7 p4 u1 8 160, t8 p4 u5 2 40)

scala>
```

Spark local host active :





Looks successful so far:



Java Code (attached too)

```
package myId.LeftOuterJoin;
//Step 1 import required classes and interfaces
import scala.Tuple2;
import java.util.Iterator;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaSparkContext;
import org.apache.spark.api.java.function.Function;
import org.apache.spark.api.java.function.PairFlatMapFunction;
import org.apache.spark.api.java.function.FlatMapFunction;
import org.apache.spark.api.java.function.PairFunction;
import java.util.Set;
import java.util.HashSet;
import java.util.Iterator;
import java.util.Arrays;
import java.util.List;
import java.util.ArrayList;
import java.util.Collections;


public class myLeftOuterJoin {
    public static void main(String[] args) {
            // TODO Auto-generated method stub
```

```
        //Step 2 Read input parameters
        if(args.length < 2) {
                System.err.println("Usage: LeftOuterJoin <users> <transactions>");
                System.exit(1);;
        }
        String usersInputFile = args[0]; //HDFS text file
        String transactionsInputFile = args[1]; //HDFS text file
        System.out.println("users="+ usersInputFile);
        System.out.println("transactions+"+ transactionsInputFile);

        //Step 3 create a JavaSparkContext object
        JavaSparkContext ctx = new JavaSparkContext();

        //Step 4 create a JavaRDD for users
        JavaRDD<String> users = ctx.textFile(usersInputFile, 1);

        //<K2, V2> JavaPairRDD<K2, V2> mapToPair(PairFunction<T, K2, V2> f)
        //Return a new RDD by applying a function to all elements of this RDD.
        //PairFunction<t, K, V> where T => Tuple2<K, V>
        JavaPairRDD<String, Tuple2<String, String>> usersRDD =
                                users.mapToPair(new PairFunction<
                                        String, //T
                                        String, //K
                                        Tuple2<String, String> //V
                                >(){
                                public Tuple2<String, Tuple2<String, String>> call(String s) {
                                        String[] userRecord = s.split("\t");
                                        Tuple2<String, String> location =
                                                new Tuple2<String, String>("L", userRecord[1]);
                                return new Tuple2<String, Tuple2<String,
String>>(userRecord[0], location);
                                }
                        });

        //Step 5 create a JavaRDD for transactions
        JavaRDD<String> transactions = ctx.textFile(transactionsInputFile, 1);

        //mapToPair
       //<K2, V2> JavaPairRDD<K2, V2> mapToPair(PairFunction<T, K2, V2> f)
        //Return a new RDD by applying a function to all elements of the RDD.
        //PairFunction<T, K, V>
        //T => Tuple2<K, V>
        JavaPairRDD<String, Tuple2<String, String>> transactionsRDD =
                                transactions.mapToPair(new PairFunction<
```

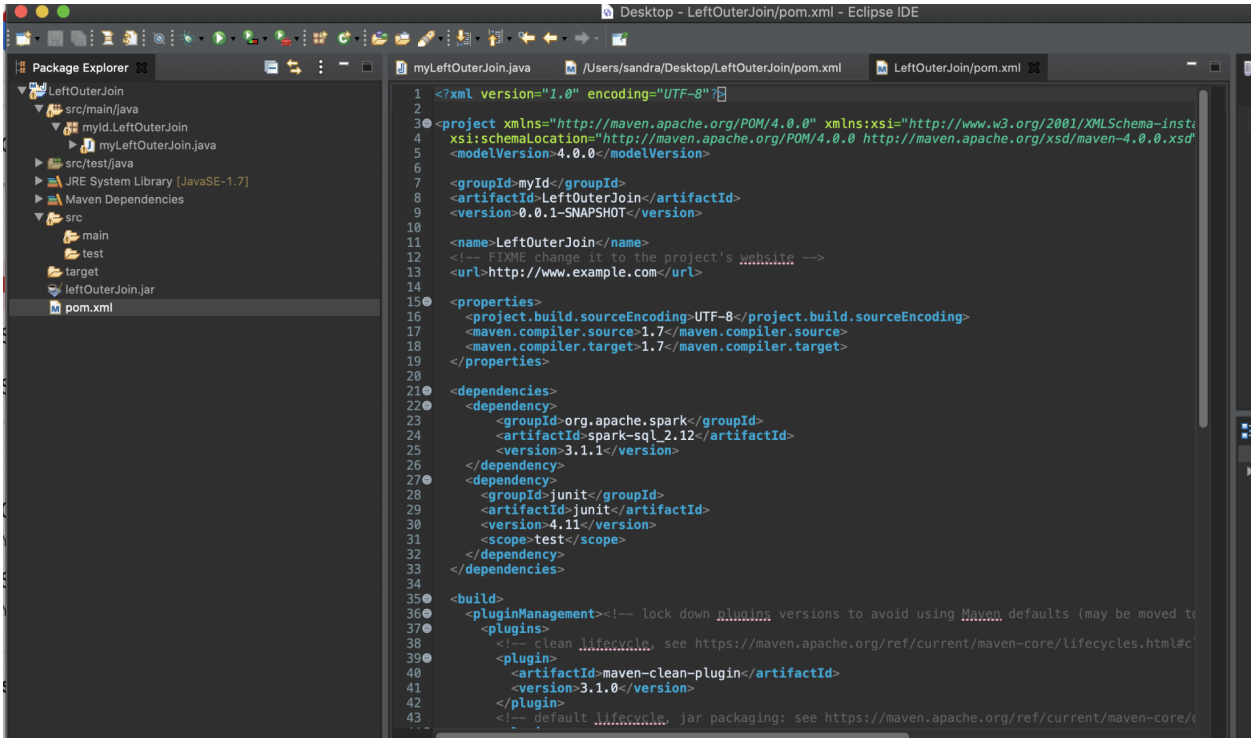
```
                                        String, //T
                                        String, //K
                                        Tuple2<String, String> //V
                                >(){
                                public Tuple2<String, Tuple2<String, String>> call(String s) {
                                        String[] transactionRecord = s.split("\t");
                                        Tuple2<String, String> product =
                                                new Tuple2<String, String>("P",
transactionRecord[1]);
                                        return new Tuple2<String, Tuple2<String,
String>>(transactionRecord[2], product);
                                        }
                                });

        //Step 6 create a union of the RDDs created in step 4 and step 5
        JavaPairRDD<String, Tuple2<String, String>> allRDD =
                        transactionsRDD.union(usersRDD);

        //Here we perform a union() on usersRDD and transactionsRdd
        //JavaPairRDD<String, Tuple2<String, String>> allRDD =
        //        usersRDD.union(transactionsRDD);

        //Step 7 create a JavaPairRDD (userId, List<T2>) by calling groupBY()
        //group allRDD by userID
        JavaPairRDD<String, Iterable<Tuple2<String, String>>> groupedRDD =
                        allRDD.groupByKey();
        //now the groupedRDD entries will be as follows:
        //<userIdi, List[T2("L", location),
                                        //T2("P", Pi1),
                                        //T2("P", Pi2),
                                        //T2("P", Pi3),...
                                        //]
        // >

        //Step 8 create a productLocationRDD as JavaPairRDD<String, String>
        //PairFlatMapFunction<T, K, V>
        //T => Iterable<Tuple2<K, V>>
        JavaPairRDD<String, String> productLocationsRDD =
                        groupedRDD.flatMapToPair(new PairFlatMapFunction<
                                        Tuple2<String, Iterable<Tuple2<String, String>>>, //T
                                        String, //K
                                        String>(){  //V
                public Iterator<Tuple2<String, String>>
                        call(Tuple2<String, Iterable<Tuple2<String, String>>> s){
```

```
                //String userID = s._1; //NOT needed
                Iterable<Tuple2<String, String>> pairs = s._2;
                String location = "UNKNOWN";
                List<String> products = new ArrayList<String>();
                for(Tuple2<String, String> t2: pairs) {
                        if(t2._1.equals("L")) {
                                location = t2._2;
                        }
                        else {
                                //t2._1.equals("P")
                                products.add(t2._2);
                        }
                }

                //now emit (K, V) pairs
                List<Tuple2<String, String>> kvList =
                        new ArrayList<Tuple2<String, String>>();
                for (String product : products) {
                        kvList.add(new Tuple2<String, String>(product, location));
                }
                //Note that edges must be reciprocal; that is,
                //every {source, destination} edge must have
                // a corresponding {destination, source}
        return (Iterator<Tuple2<String, String>>) kvList;
        }

});

//Step 9 find all locations for a product;
//result will be JavaPAirRDD <String, List<String>>
JavaPairRDD<String, Iterable<String>> productByLocations =
                productLocationsRDD.groupByKey();

//debug3
List<Tuple2<String, Iterable<String>>> debug3 = productByLocations.collect();
System.out.println("--- debug3 begin ---");
for(Tuple2<String, Iterable<String>> t2 : debug3){
        System.out.println("debug3 t2._1=" + t2._1);
        System.out.println("debug3 t2._2+"+ t2._2);
}
System.out.println("---debug3 end ---");

//Step 10 finalize output by changing "value" from List<String>
//to Tuple2<Set<String>, Integer>, where you have a unique
```

```
		//set of locations and their count
		JavaPairRDD<String, Tuple2<Set<String>, Integer>>
productByUniqueLocations =
				productByLocations.mapValues(
						new Function<Iterable<String>, //input
						Tuple2<Set<String>, Integer> //output
						>(){
			public Tuple2<Set<String>, Integer> call(Iterable<String> s){
				Set<String> uniqueLocations = new HashSet<String>();
				for(String locations : s) {
					uniqueLocations.add(locations);
				}
				return new Tuple2<Set<String>, Integer>(uniqueLocations,

	uniqueLocations.size());
			}

		});

		//Step 11 print the final result RDD
		//debug4
		System.out.println("=== Unique Locations and Counts ===");
		List<Tuple2<String, Tuple2<Set<String>, Integer>>> debug4 =
			productByUniqueLocations.collect();
		System.out.println("---debug4 begins---");
		for(Tuple2<String, Tuple2<Set<String>, Integer>> t2 : debug4) {
			System.out.println("debug4 t2._1="+t2._1);
			System.out.println("debug4 t2._2="+t2._2);
		}
		System.out.println("---debug4 end---");


   }

}
```

pom.xml

Adding Apache dependencies (creatingfile) (next page):



Updating #!/bin/bash

load-spark-env.sh

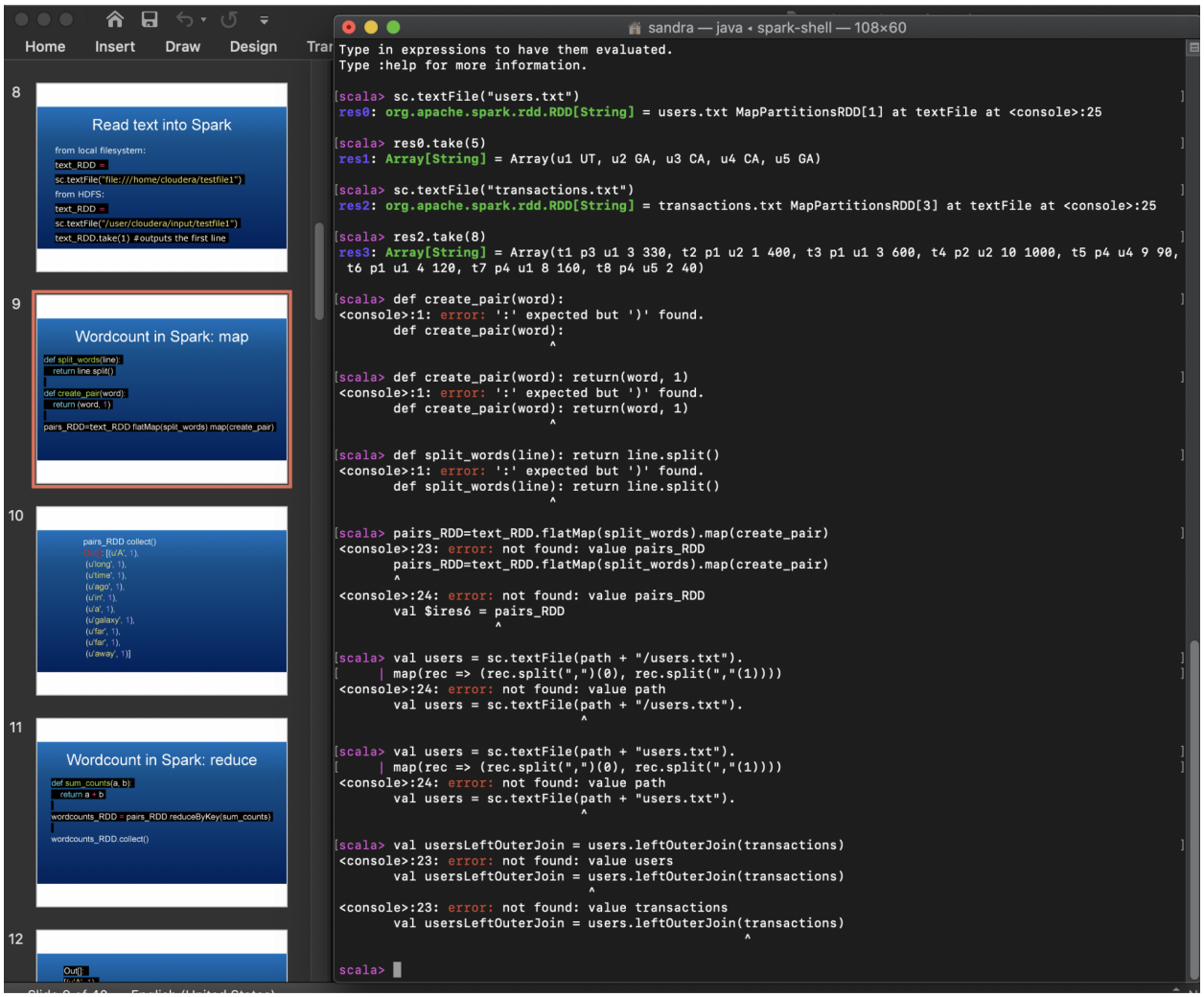
/usr/local/Cellar/apache-spark/3.1.1/bin/load-spark-env.sh (functions)

```
#!/bin/bash
#JAVA_HOME="/usr/local/opt/openjdk@11" exec "/usr/local/Cellar/apache-spark/3.1.1/libexec/bin/load-spark-env.sh"  "$@"
export JAVA_HOME=/usr/java/jdk7
export SPARK_HOME=/usr/local/spark-2.13.5   #1.1.0
export SPARK_MASTER=spark://http://10.0.0.91:4041 #myserver100:7077
export BOOK_HOME=/home/data-algorithms-book
export APP_JAR=$BOOK_HOME/dist/data_algorithms_book_jar
USERS=/data/leftouterjoins/users.txt
TRANSACTIONS=/data/leftouterjoins/transactions.txt

#Run on a Saprk cluster
prog=org.dataalgorithms.chap04.spark.SparkLeftOuterJoin
$SPARK_HOME/bin/spark-submit \
    - --class $prog \
    - --master $SPARK_MASTER \
    - --executor-memory @G \
    - --total-executor-cores 20 \
    $APP_JAR $USERS 4TRANSACTIONS
```

Creating .jar file Inserting java code into scala → error!

```
sandra@Sandras-MacBook-Pro bin % ./spark-submit --class LeftOuterJoin /Users/Sandra/Desktop/LeftOuterJoin/leftOuterJoin.jar
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/usr/local/Cellar/apache-spark/3.1.1/libexec/jars/spark-unsafe_2.12-3.1.1.jar) t
o constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
21/04/20 15:32:15 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Error: Failed to load class LeftOuterJoin.
log4j:WARN No appenders could be found for logger (org.apache.spark.util.ShutdownHookManager).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
sandra@Sandras-MacBook-Pro bin %
```

Other approach in Spark-shell (running commands but getting nowhere):



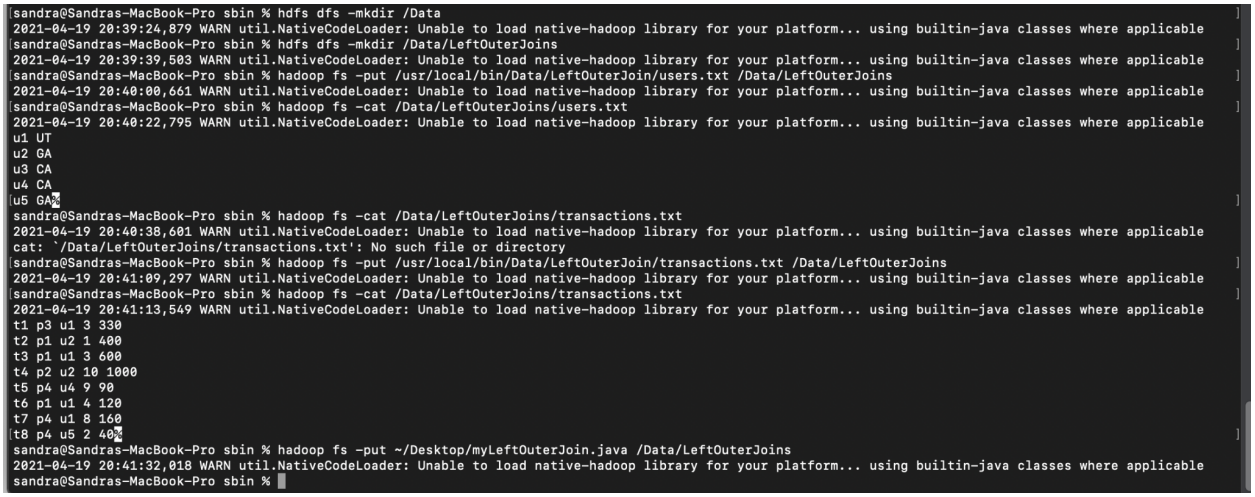

```
scala> val users = sc.textFile(path + users.txt).
     | map(rec => (rec.split(",")(0), rec.split(",")(1)))
<console>:24: error: not found: value path
       val users = sc.textFile(path + users.txt).
                               ^
<console>:24: error: value txt is not a member of org.apache.spark.rdd.RDD[(String, Array[String])]
       val users = sc.textFile(path + users.txt).
                                      ^

scala> val users = sc.textFile(path + hdfs:///Data/LeftOuterJoins/users.txt).
     | map(rec => (rec.split(",")(0), rec.split(",")(1)))
<console>:2: error: ')' expected but '(' found.
       map(rec => (rec.split(",")(0), rec.split(",")(1)))
          ^

scala> val users = sc.textFile(path + "/hdfs:///Data/LeftOuterJoins/users.txt").
     | map(rec => (rec.split(",")(0), rec.split(",")(1)))
<console>:24: error: not found: value path
       val users = sc.textFile(path + "/hdfs:///Data/LeftOuterJoins/users.txt").
                               ^

scala>
```

Unable to proceed or figure how to write the path for val users and val transactions so that i can try performing leftOuterJoin command
val usersLeftOuterJoin = users.leftOuterJoin(transactions)

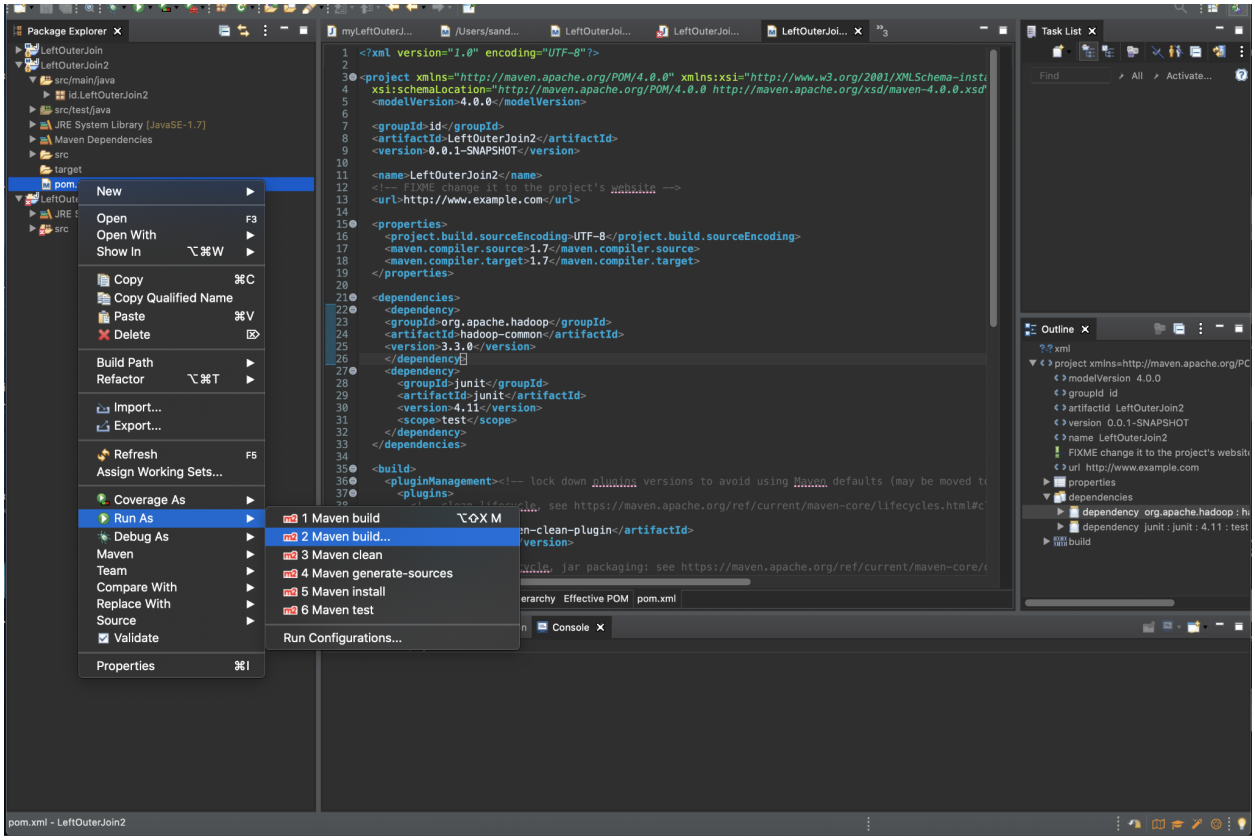
Attempt with Method hadoop for extra credit:

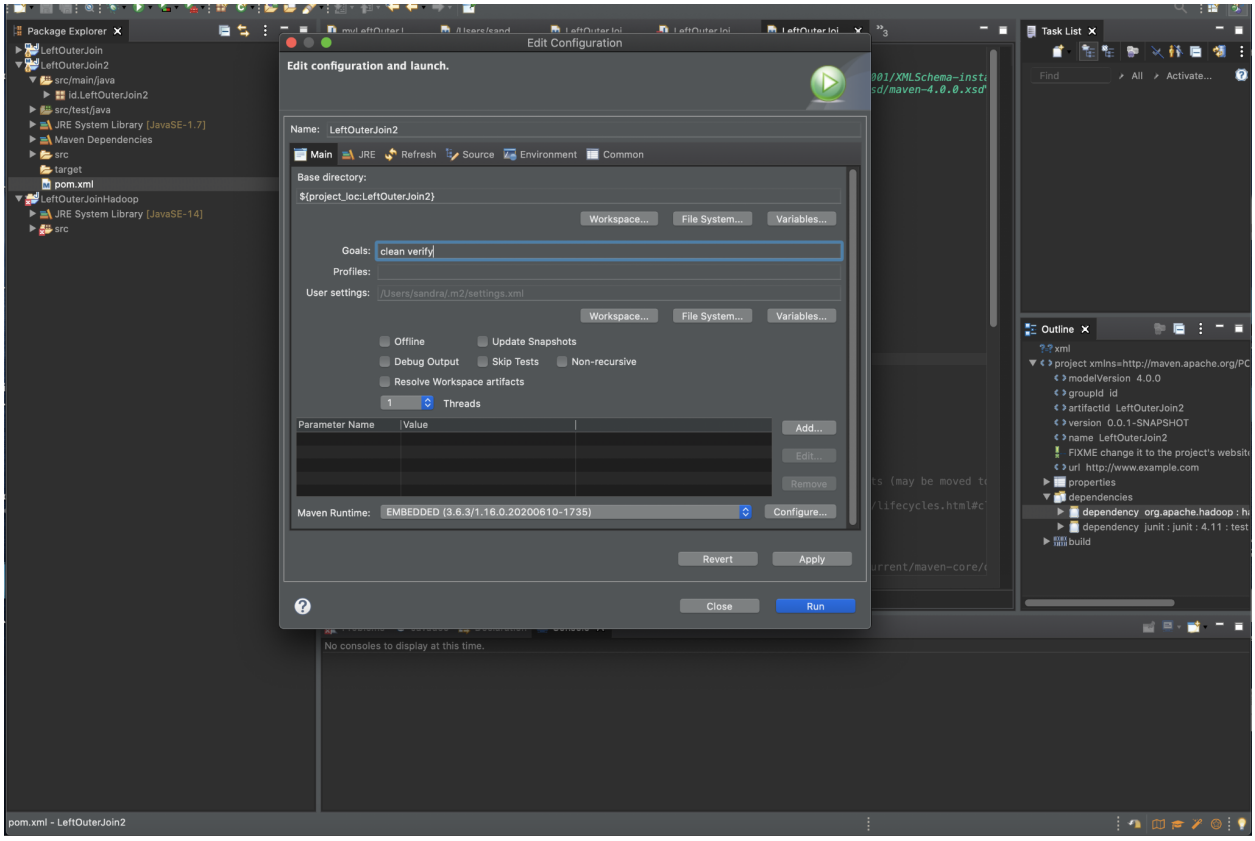
Java Code attached(LeftOuterJoinHadoop, LeftOuterJoinHadoopMapper, & LeftOuterJoinHadoopReducer)

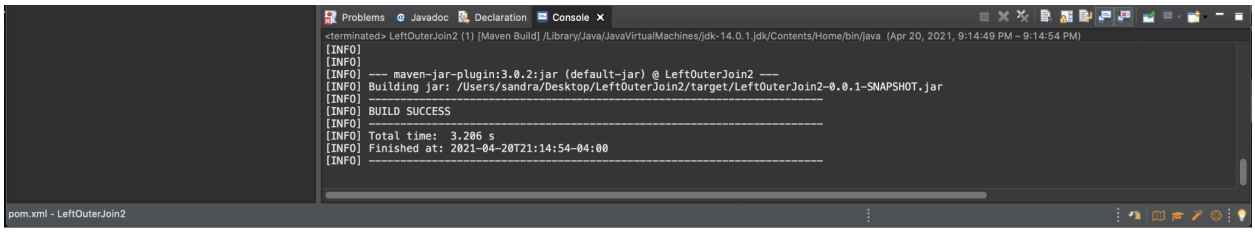
Data in place:



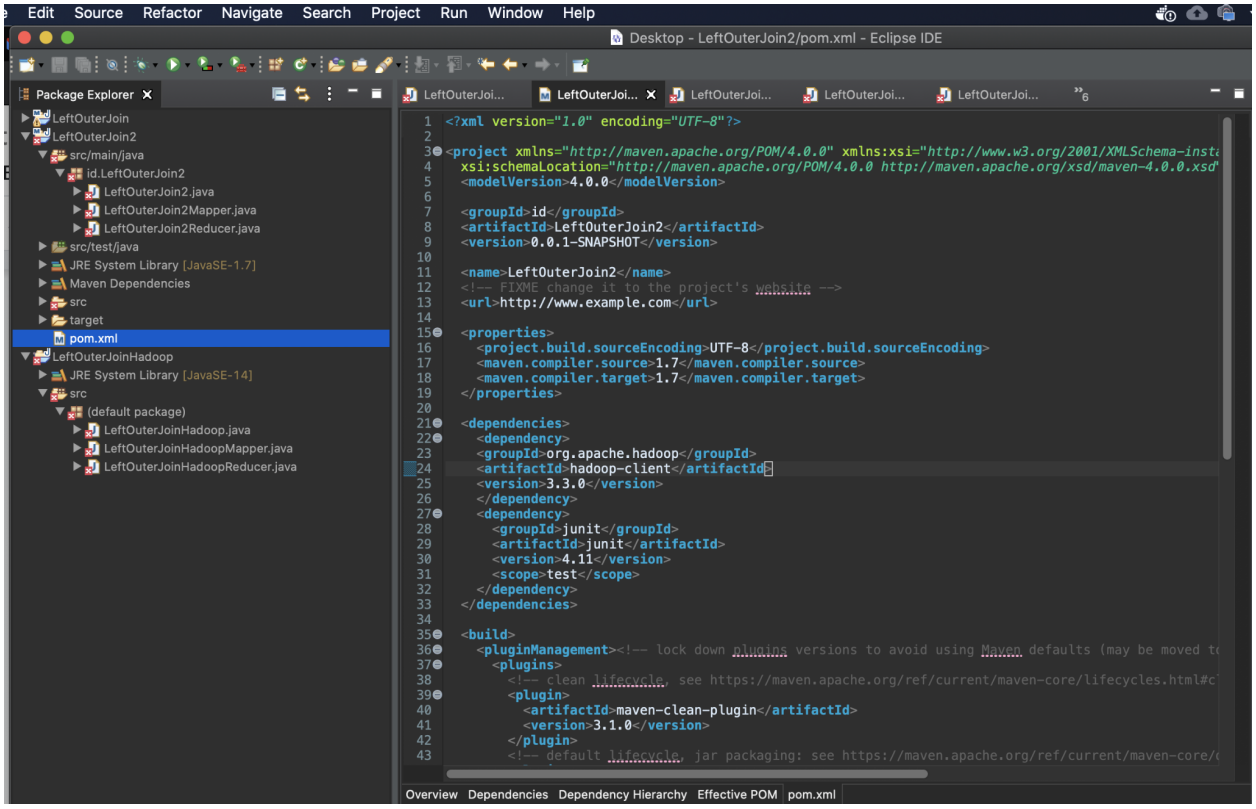
Adding hadoop dependency in pom.xml file and running maven build for success:


Package Explorer
LeftOuterJoin
LeftOuterJoin2
src/main/java
id.LeftOuterJoin2
src/test/java
JRE System Library [JavaSE-1.7]
Maven Dependencies
src
target
pom.
New
Open F3
Open With
Show In ⌥⌘W
Copy ⌘C
Copy Qualified Name
Paste ⌘V
Delete
Build Path
Refactor ⌥⌘T
Import...
Export...
Refresh F5
Assign Working Sets...
Coverage As
Run As
Debug As
Maven
Team
Compare With
Replace With
Source
Validate
Properties ⌘I
1 Maven build ⌥⇧X M
2 Maven build...
3 Maven clean
4 Maven generate-sources
5 Maven install
6 Maven test
Run Configurations...
myLeftOuterJ...
/Users/sand...
LeftOuterJoi...
LeftOuterJoi...
LeftOuterJoi...
Task List
Find
All
Activate...
Outline
project xmlns=http://maven.apache.org/PC
modelVersion 4.0.0
groupId id
artifactId LeftOuterJoin2
version 0.0.1-SNAPSHOT
name LeftOuterJoin2
FIXME change it to the project's websit
url http://www.example.com
properties
dependencies
dependency org.apache.hadoop : h
dependency junit : junit : 4.11 : test
build
Effective POM
pom.xml
Console
pom.xml - LeftOuterJoin2

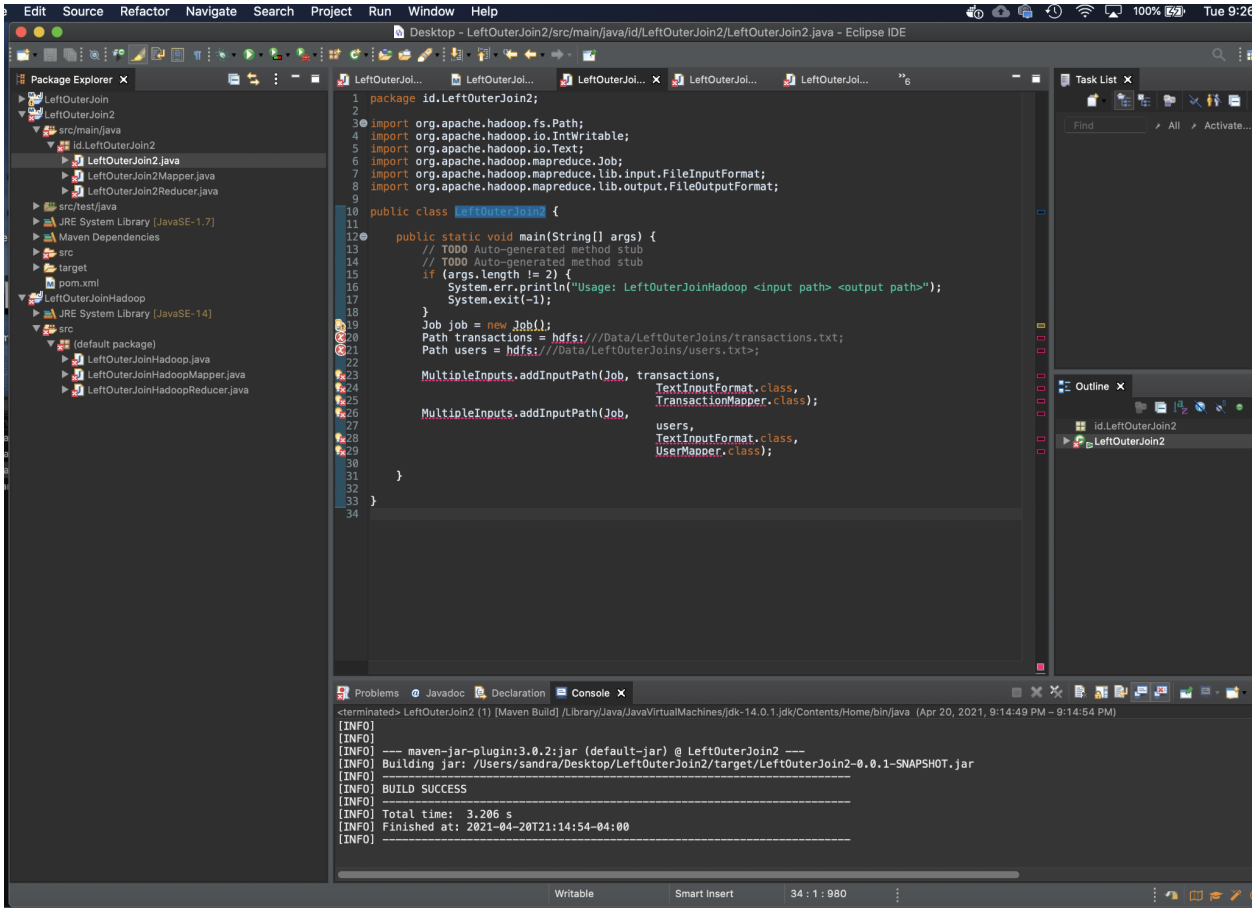




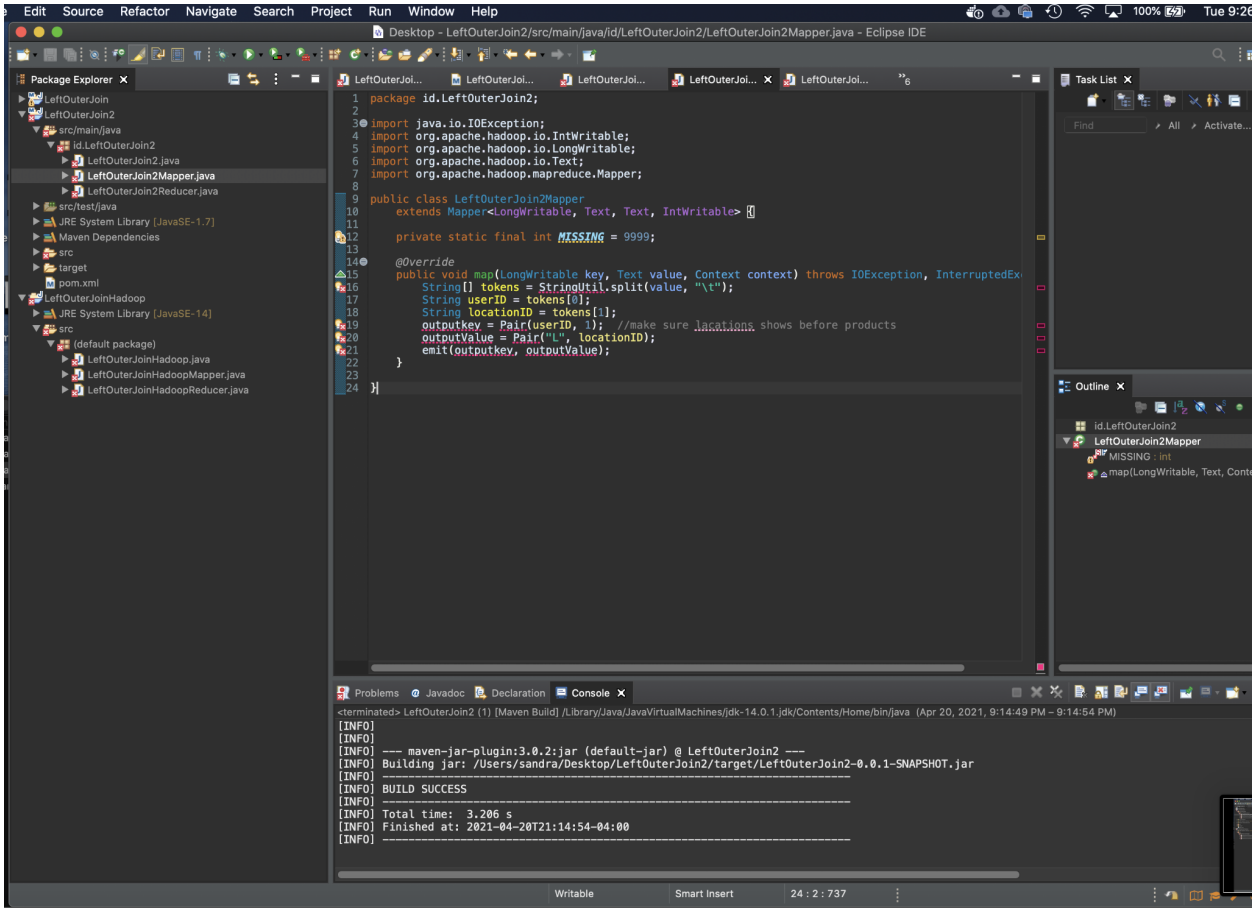
Java code presented error the import org.apache.hadoop.mapreduce cannot be resolved
So switch pom.xml to the following dependency to get rid of that.



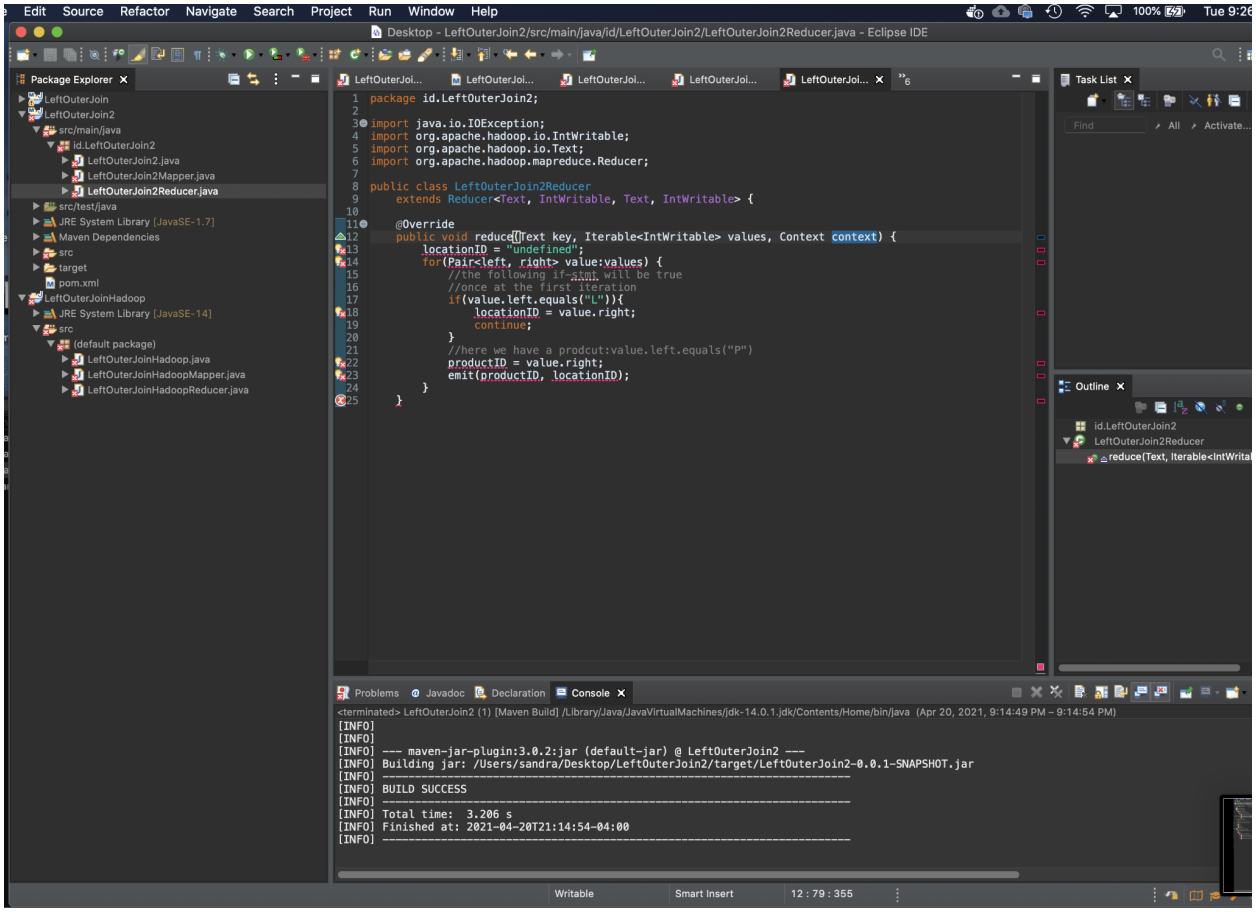
Java code still present with errors i wasn’t not able to fix


Edit Source Refactor Navigate Search Project Run Window Help
Desktop - LeftOuterJoin2/src/main/java/id/LeftOuterJoin2/LeftOuterJoin2.java - Eclipse IDE
Package Explorer
LeftOuterJoin
LeftOuterJoin2
src/main/java
id.LeftOuterJoin2
LeftOuterJoin2.java
LeftOuterJoin2Mapper.java
LeftOuterJoin2Reducer.java
src/test/java
JRE System Library [JavaSE-1.7]
Maven Dependencies
src
target
pom.xml
LeftOuterJoinHadoop
JRE System Library [JavaSE-14]
src
(default package)
LeftOuterJoinHadoop.java
LeftOuterJoinHadoopMapper.java
LeftOuterJoinHadoopReducer.java
package id.LeftOuterJoin2;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public class LeftOuterJoin2 {
public static void main(String[] args) {
// TODO Auto-generated method stub
// TODO Auto-generated method stub
if (args.length != 2) {
System.err.println("Usage: LeftOuterJoinHadoop <input path> <output path>");
System.exit(-1);
}
Job job = new Job();
Path transactions = hdfs:///Data/LeftOuterJoins/transactions.txt;
Path users = hdfs:///Data/LeftOuterJoins/users.txt>;
MultipleInputs.addInputPath(Job, transactions,
TextInputFormat.class,
TransactionMapper.class);
MultipleInputs.addInputPath(Job,
users,
TextInputFormat.class,
UserMapper.class);
}
}
Task List
Outline
id.LeftOuterJoin2
LeftOuterJoin2
Problems Javadoc Declaration Console
<terminated> LeftOuterJoin2 (1) [Maven Build] /Library/Java/JavaVirtualMachines/jdk-14.0.1.jdk/Contents/Home/bin/java (Apr 20, 2021, 9:14:49 PM – 9:14:54 PM)
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ LeftOuterJoin2 ---
[INFO] Building jar: /Users/sandra/Desktop/LeftOuterJoin2/target/LeftOuterJoin2-0.0.1-SNAPSHOT.jar
[INFO] --
[INFO] BUILD SUCCESS
[INFO] --
[INFO] Total time: 3.206 s
[INFO] Finished at: 2021-04-20T21:14:54-04:00
[INFO] --
Writable
Smart Insert
34 : 1 : 980

Mapper:


Edit Source Refactor Navigate Search Project Run Window Help
Desktop - LeftOuterJoin2/src/main/java/id/LeftOuterJoin2/LeftOuterJoin2Mapper.java - Eclipse IDE
Package Explorer
LeftOuterJoin
LeftOuterJoin2
src/main/java
id.LeftOuterJoin2
LeftOuterJoin2.java
LeftOuterJoin2Mapper.java
LeftOuterJoin2Reducer.java
src/test/java
JRE System Library [JavaSE-1.7]
Maven Dependencies
src
target
pom.xml
LeftOuterJoinHadoop
JRE System Library [JavaSE-14]
src
(default package)
LeftOuterJoinHadoop.java
LeftOuterJoinHadoopMapper.java
LeftOuterJoinHadoopReducer.java
package id.LeftOuterJoin2;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class LeftOuterJoin2Mapper
extends Mapper<LongWritable, Text, Text, IntWritable> {
private static final int MISSING = 9999;
@Override
public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedEx
String[] tokens = StringUtil.split(value, "\t");
String userID = tokens[0];
String locationID = tokens[1];
outputkey = Pair(userID, 1); //make sure lacations shows before products
outputValue = Pair("L", locationID);
emit(outputkey, outputValue);
}
}
Task List
Outline
id.LeftOuterJoin2
LeftOuterJoin2Mapper
MISSING : int
map(LongWritable, Text, Conte
Problems Javadoc Declaration Console
<terminated> LeftOuterJoin2 (1) [Maven Build] /Library/Java/JavaVirtualMachines/jdk-14.0.1.jdk/Contents/Home/bin/java (Apr 20, 2021, 9:14:49 PM – 9:14:54 PM)
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ LeftOuterJoin2 ---
[INFO] Building jar: /Users/sandra/Desktop/LeftOuterJoin2/target/LeftOuterJoin2-0.0.1-SNAPSHOT.jar
[INFO] --
[INFO] BUILD SUCCESS
[INFO] --
[INFO] Total time: 3.206 s
[INFO] Finished at: 2021-04-20T21:14:54-04:00
[INFO] --
Writable
Smart Insert
24 : 2 : 737

Reducer:



Unable to proceed!