MAHARISHI INTERNATIONAL UNIVERSITY

Formerly Maharishi University of Management

Big Data (CS522)- Project Presentation

PREPARED BY

ABRHA GEBRESLASSIE BERHE
AZIZA ALHOUNI
FRIEW GEBREMEDHIN ABRAHA

SUBMITTED TO

PROF. PREM NAIR

December 2020

PART 1- HADOOP

Setup

1. Create the following directories(You will find it in the submitted packages) on cloudera Desktop
   - ProjectInputFiles
   - ProjectJarFiles
2. If you need input file, Put it inside the “input” directory under its respective package name in the directory step1-a. The package names for each problem are listed below.
   - Part_1A, Part_1B, Part_1C, Part_1D, Part_2, Part_3, Part_4

Setup Cont'd

3. Open terminal

4. Sudo su hdfs
   - Run the following batch file to process the MapReduce for the problem we want just by giving the package name as an argument as follows
     - Bash-4.1$ Desktop/Execute Part_1A
5. Exit from hadoop environment to local environment
   - Similarly Run the following batch file to copy the MapReduce Outputs to their respective local directory
     - [cloudera@quickstart ~] Desktop/CopyOutputToLocal Part_1A
6. The output files could be found in the ProjectInputFiles/{Package-Name}/Output

- Note: each java files can be found in the Ecliplse project name under the directory "BigDataCourseProject" with in the submitted package (BigData-Project-Group1).

Pseudo codes

Pair approach

```
class Mapper
    method Map(docid a, doc d)
        for all term u in record r do
            for all term v in Window(u) do
                Emit((u, v), 1)
                Emit((u, *), 1)
class Reducer
    methode initialize()
        sum = 0
    method Reduce(Pair(u, v), Integer [c1, c2, …])
        s = 0
        for all Integer c in [c1, c2, …] do
            s = s + c
        if(v== “*”)
            sum = s
        else
        Emit((u, v), s/sum)
```

Stripe Approach

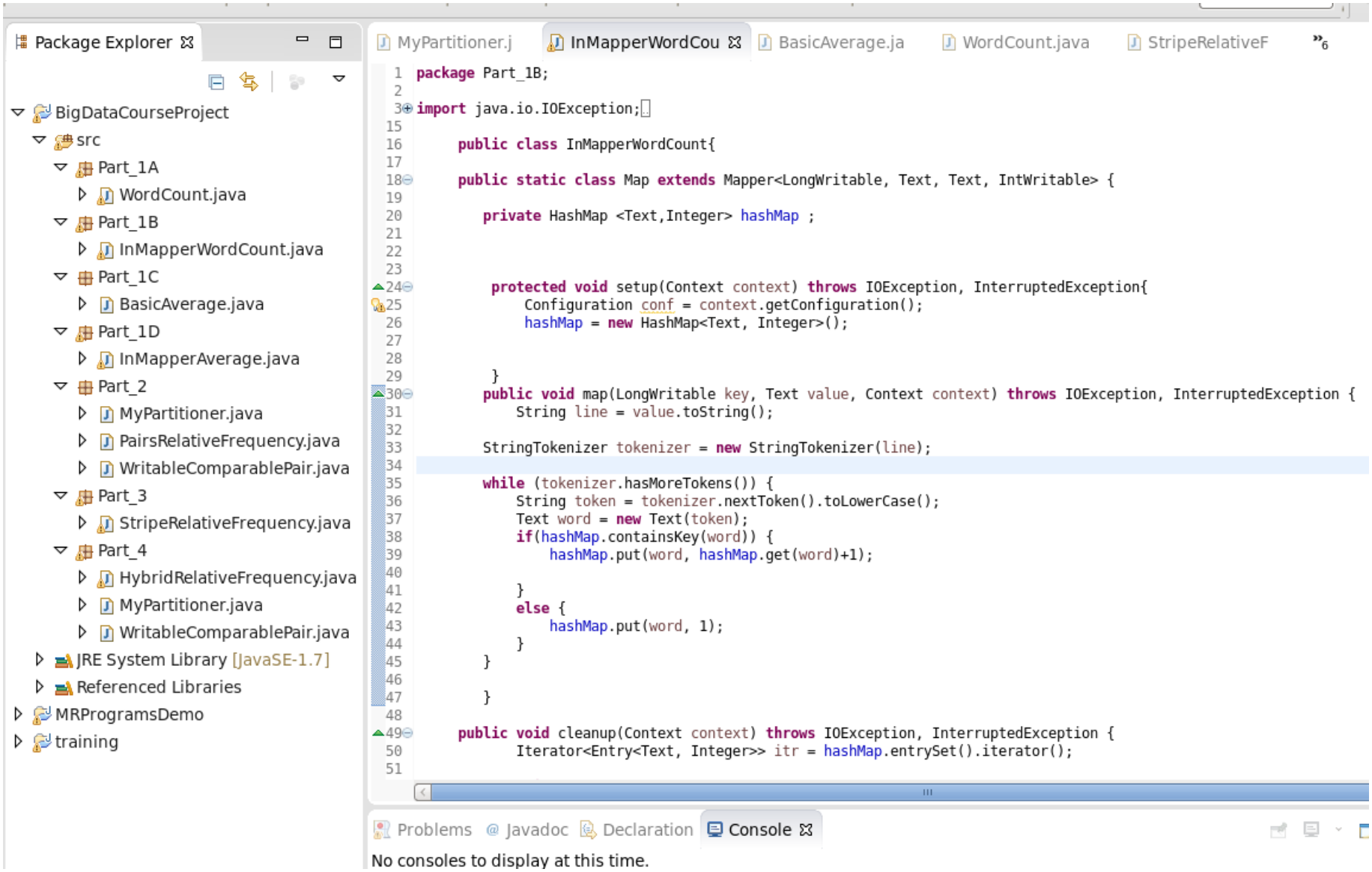
```
class Mapper
        method Map(docid a, doc d)
              for all term u in record r do
                H = new AssociativeArray
               for all term v in Window(u) do
                   H{v} = H{v} + 1 .                    //Tally words co-occurring with u
                Emit(u, H)
class Reducer
         method Reduce(term u, AssociativeArray [H1, H2, …])
                   HFINAL= new AssociativeArray
                   for all stripe H in [H1, H2, …] do
                        Hf= Hf+ H                  //elementwise addition
                    s= Sum(Hf)
                    Emit(u, Hf/s)
```

Hybrid approach

```
Class Mapper:
        Method Initialize()
                H = new AssociativeArray();
        Method map(docid a, doc d)
                for all w in d do:
                        for all u in
neighbors(w) do:
                                H{(w,u)} ++
        Method close():
                for all pair(w,u) in H do:
                        Emit(pair(w,u),
H{(w,u)})
```

```
Class Reducer:
        Method initialize:
                wPrev = null
                H = new AssociativeArray();
        Method reduce (Pair(w, u), [C1, C2, C3, ...])
                if (w != wPrev && wPrev != null) then:
                        total = total(H)
                        Emit(wPrev, H / total)
                        H.clear()
                H{u} = sum(C1,C2….)
                wPrev = w
        Method close:
                total = total(H)
                Emit(wPrev, H / total)
```

Sample – Java Code



Script shell Code – to execute some hdfs commands

```
if $(hadoop fs -test -d /input/${PACKAGE})
then
        echo "Replacing Directory /input/$PACKAGE"
        hdfs dfs -copyFromLocal -f Desktop/ProjectInputFiles/${PACKAGE}/Input/*.txt /input/${PACKAGE}
else
        echo "Creating Directory /input/$PACKAGE"
        hadoop fs -mkdir /input/$PACKAGE
        hdfs dfs -copyFromLocal -f Desktop/ProjectInputFiles/${PACKAGE}/Input/*.txt /input/${PACKAGE}
fi

echo "============== Printing Input File... ===================="
hadoop fs -cat /input/$PACKAGE/*

echo "============== Map Reduce Running... ===================="
if $(hadoop fs -test -d /output/$PACKAGE)
then
        echo "Replacing Directory /output/$PACKAGE"
        hadoop fs -rm -r /output/$PACKAGE
        hadoop jar Desktop/ProjectJarFiles/BigDataJarFiles.jar ${PACKAGE}.${CLASS} /input/${PACKAGE}/*.txt /output/
$PACKAGE
else
        echo "Creating Directory /output/$PACKAGE"
        hadoop jar Desktop/ProjectJarFiles/BigDataJarFiles.jar ${PACKAGE}.${CLASS} /input/${PACKAGE}/*.txt /output/
$PACKAGE

fi

echo "============== Printing Output File... ==========
hadoop fs -cat /output/$PACKAGE/*
```

Execute | CopyOutputToLocal

```
#!/bin/bash
if [ $# -ne 1 ]
then
 echo "Please input your parameter(Java Package Name)";
 exit 1;
fi

PACKAGE=$1

echo "============== Copying Output to Local Directory...===================="
if [ -d Desktop/ProjectInputFiles/$PACKAGE/Output ]
then
        rm -f Desktop/ProjectInputFiles/$PACKAGE/Output/part*
        hadoop fs -copyToLocal /output/$PACKAGE/part-r-* Desktop/ProjectInputFiles/${PACKAGE}/Output/
        echo "Success, output file copied";
        echo "You can find it in Desktop/ProjectInputFiles/${PACKAGE}/Output/";
        echo " ";
else
        echo "Creating Directory /output/$PACKAGE"
        hadoop fs -copyToLocal /output/$PACKAGE/part-r-* Desktop/ProjectInputFiles/${PACKAGE}/Output/

fi
```

Sample – Output

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ pwd
/home/cloudera
[cloudera@quickstart ~]$ sudo su hdfs
bash-4.1$ pwd
/home/cloudera
bash-4.1$ Desktop/Execute Part 1A
Replacing Directory /input/Part_1A
copyFromLocal: Cannot create file/input/Part_1A/WordCount.txt._COPYING_. Name node is in safe mode.
================ Printing Input File... ====================
cat: Zero blocklocations for /input/Part_1A/WordCount.txt. Name node is in safe mode.
================ Map Reduce Running... ====================
Replacing Directory /output/Part_1A
```

Running commands using shell script

```
File  Edit  View  Search  Termina
a        4
abfs     1
an       2
and      5
apache   2
api      2
applications     1
are      5
as       1
attributes       4
available        5
aws      1
azure    1
based    2
begin    1
better   1
between  1
blocks   1
changes  1
cli      1
cluster  2
connector        3
consider         1
containers       2
data     2
datalake         1
deep     1
deploy   1
details  4
develop  1
```

Sample output for WordCount using basic air approach

Sample output for relative frequency using hybrid method

```
A10     {{A12 => 0.125}{B12 => 0.25}{D76 => 0.375}{C31 => 0.25}}
A12     {{B76 => 0.08695652173913043}{A10 => 0.043478260869565216}{B12 => 0.13043478260869565}{D76 => 0.34782608695652173}{C31 => 0.391304347826087}}
B12     {{B76 => 0.07407407407407407}{A12 => 0.18518518518518517}{A10 => 0.037037037037037035}{D76 => 0.3333333333333333}{C31 => 0.3703037037037035}}
B76     {{A12 => 0.07142857142857142}{A10 => 0.07142857142857142}{B12 => 0.14285714285714285}{D76 => 0.35714285714285715}{C31 => 0.35714285714285715}}
C31     {{B76 => 0.10526315789473684}{A12 => 0.21052631578947367}{A10 => 0.052631578947368421}{B12 => 0.15789473684210525}{D76 => 0.47368421052631576}}
D76     {{B76 => 0.09523809523809523}{A12 => 0.23809523809523808}{A10 => 0.047619047619047616}{B12 => 0.19047619047619047}{C31 => 0.42857142857142855}}
[cloudera@quickstart Output]$
```

PART – 2 : SPARK/Scala

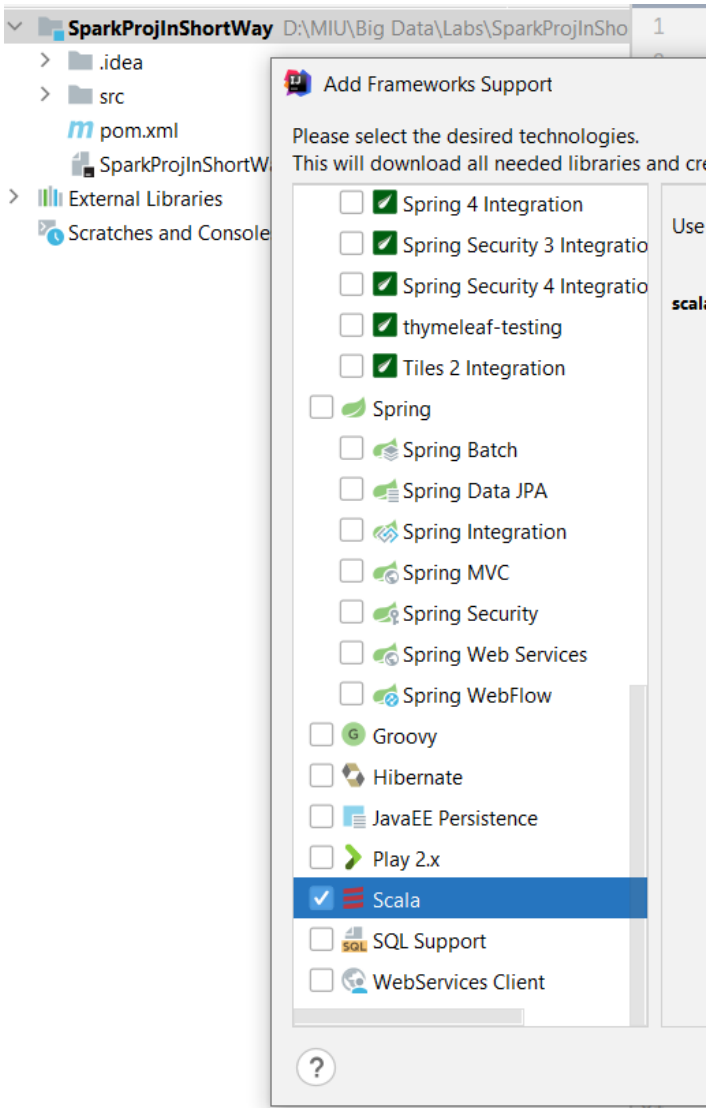
Setup steps

1. Install Scala : Go to IntelliJ IDEA -> Preferences → Plugins →Type “Scala” to search for Scala plug-in → Install the plugin
2. New Project -> Maven
3. Add the following dependencies in poem.xml

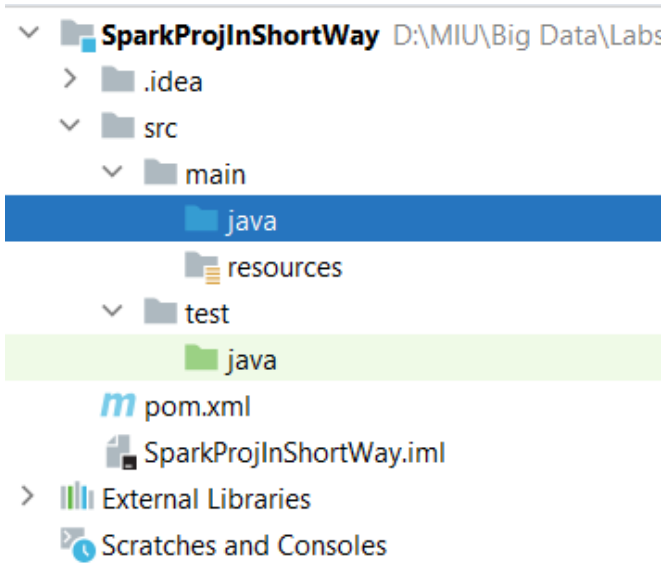
```
<dependencies>

    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-core_2.12</artifactId>
        <version>2.4.3</version>
    </dependency>

    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-sql_2.12</artifactId>
        <version>2.4.3</version>
    </dependency>

    <dependency>
        <groupId>au.com.bytecode</groupId>
        <artifactId>opencsv</artifactId>
        <version>2.4</version>
    </dependency>
</dependencies>
```

```
<dependencies>

  <dependency>
    <groupId>org.apache.spark</groupId>
    <artifactId>spark-core_2.12</artifactId>
    <version>2.4.3</version>
  </dependency>

  <dependency>
    <groupId>org.apache.spark</groupId>
    <artifactId>spark-sql_2.12</artifactId>
    <version>2.4.3</version>
  </dependency>

  <dependency>
    <groupId>au.com.bytecode</groupId>
    <artifactId>opencsv</artifactId>
    <version>2.4</version>
  </dependency>
</dependencies>
```

4. Right click on Project → Add Framework Support and select scala



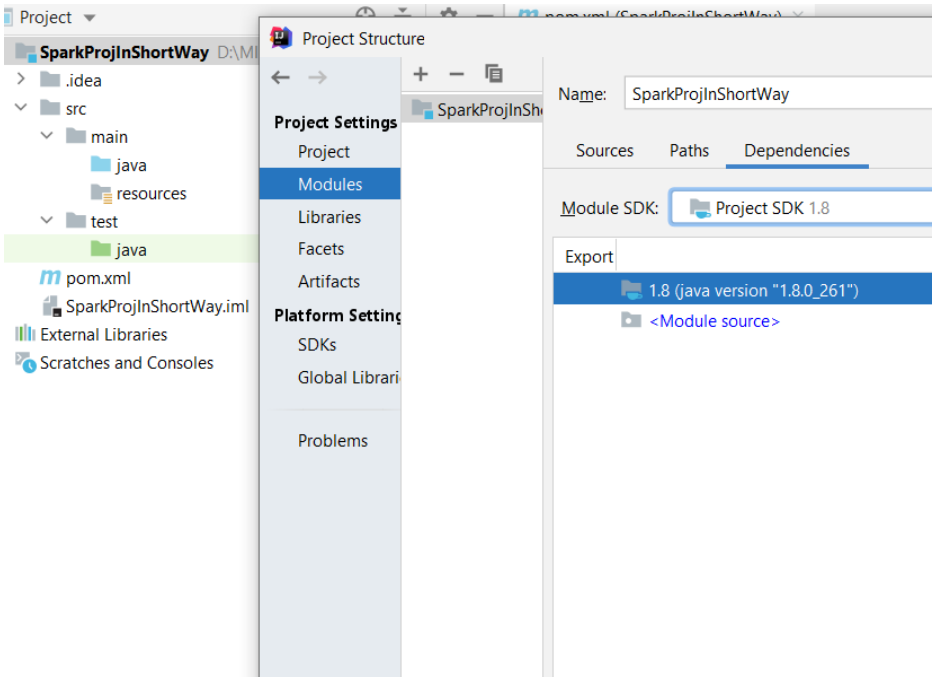
5. Rename the "java" package inside the project-->src-->main and test to "scala"

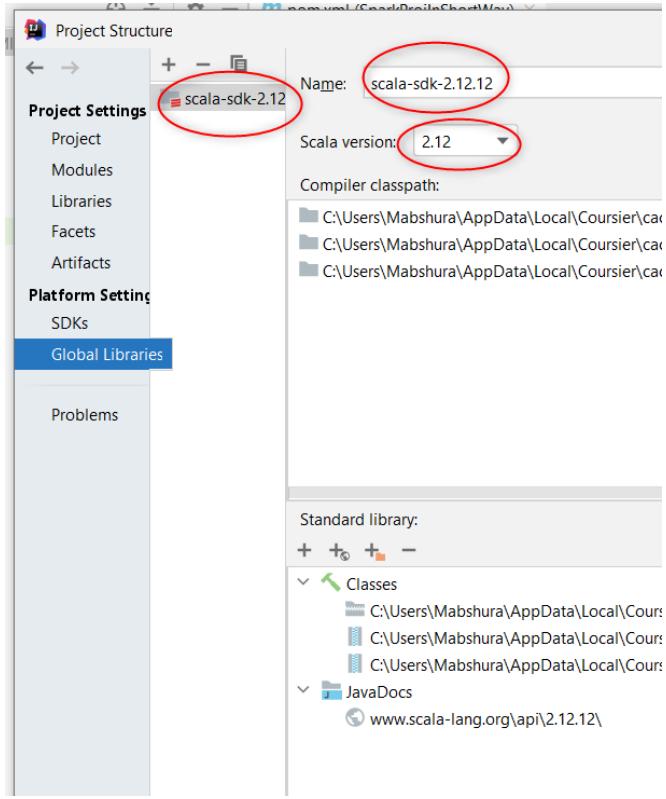


6. Create New Package with in the scala directory→create new **scala object** with in the this package

5. Check the correct versions

- Right click on the project → Open Modules Settings
- In Modules → make sure java is java 8 and check the JAVA_HOMe is also java8 In system path
- In Global Libraries → check the scala version mathes with what is installed
- Use such versions in the dependency too





If you face a problem on downloading maven Dependency

- Set the JAVA_HOME path used by MAVEN
  - Settings >> Build >> Build Tools >> Importing >> JDK for importer. Pointed it to JAVA_HOME.
  - Download the dependencies manually using the following command in the maven terminal
    - mvn clean install –U
  - Re import Maven Projects

If Hadoop displays problem related to loading winutils.exe

- Download the winutils.exe for the hadoop version you are using from https://github.com/steveloughran/winutils/tree/master/hadoop-x.x.x/bin
- Create a folder 'winutils' in C:\ drive→ create a folder 'bin' inside folder 'winutils' and copy the winutils.exe in that folder. So the location of winutils.exe will be C:\winutils\bin\winutils.exe
- Open environment variable and set HADOOP_HOME=C:\winutils [NOTE: Please do not add \bin in HADOOP_HOME]
- Then add %HADOOP_HOME%\bin to PATH environment variable
- Restart IDE

Sample Code

importing the file and doing the calculation on the population data

```
object ChicksWeightByDiet extends App {
  override def main(args: Array[String]) {
    val conf2 = new SparkConf().setAppName("Spark and SparkSql").setMaster("local")
    val sc = new SparkContext(conf2)
    sc.setLogLevel("WARN")

    val sqlContext = new org.apache.spark.sql.SQLContext(sc)


    def getDiet(in: String): String = {
      in.split( regex = "\"")(1)
    }


    var myHashMap = collection.mutable.Map[String, collection.Map[Int, (Double, Double)]]()
    var populationMap = collection.Map[Int, (Double, Double)]()

    val myData = sc.textFile( path = "src/main/java/ChicksWightAnalysis/ChickWeight.csv")
    val myLines = myData.map(line => line.split( regex = ",").map(_.trim))
    val header = myLines.first()
    val chickDataWithNoHeader = myLines.filter(_ (0) != header(0))
    val populationData = chickDataWithNoHeader.map(x => (getDiet(x(4)), x(1))).cache()
    val cleanPopulationData = populationData.map(x => (x._1.toInt, x._2.toDouble))
    val computation = cleanPopulationData.groupByKey().map(x => (x._1, (x._2.count(_ => true),
    val results = computation.map(x => (x._1, (x._2._2 / x._2._1, (x._2._3 / x._2._1 - (x._2._2
    // println("==========Printing Aggregate of the population ============")
    populationMap = results.sortBy(_._1, ascending = true).collectAsMap()
    myHashMap.put("Population", populationMap)
```

The Data Set (ChickWeight.csv)

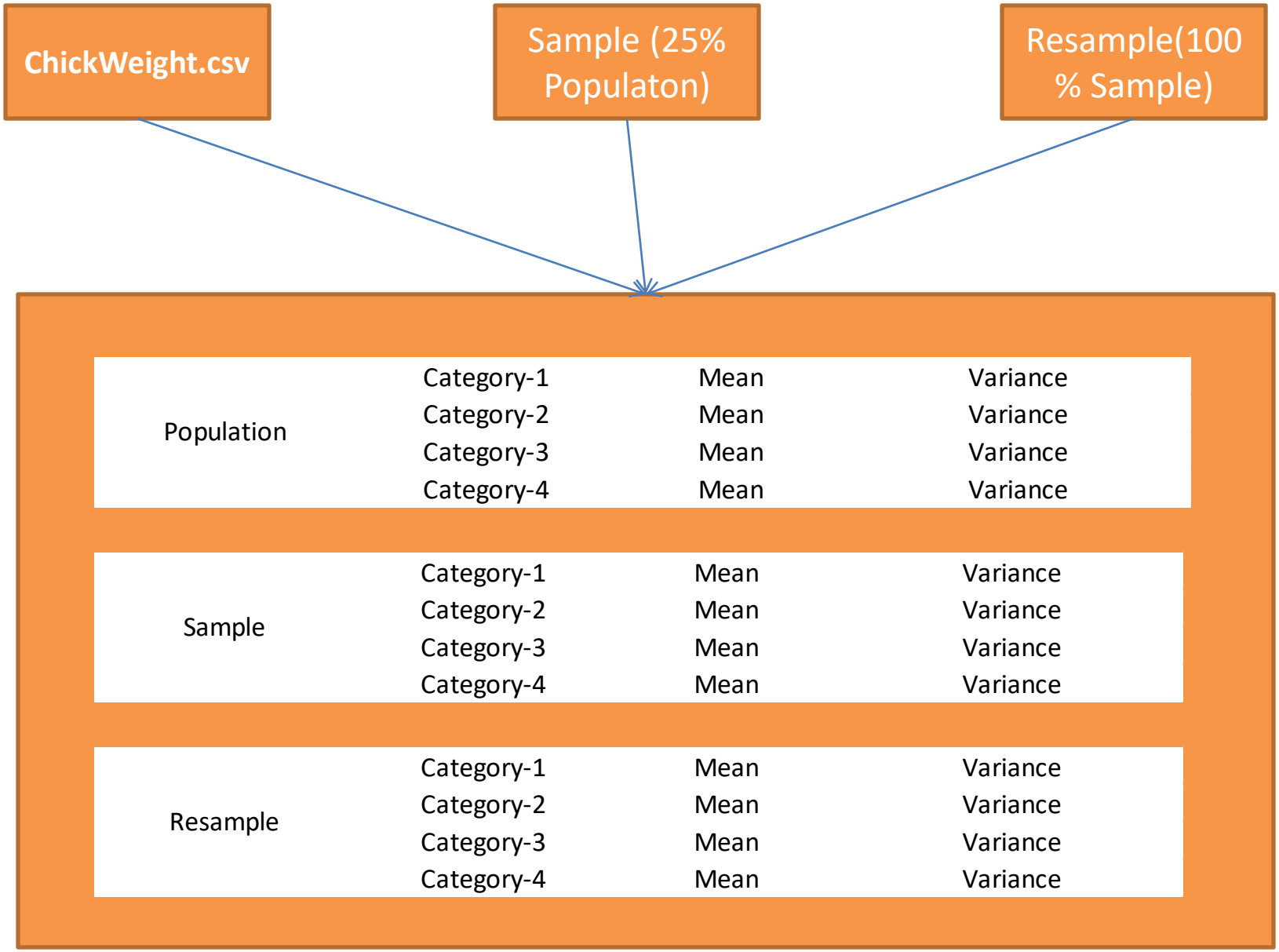
The ChickWeight data frame has 578 rows and 4 columns from an experiment on the effect of diet on early growth of chicks.

Columns : 5 **Rows :** 578

	weight	Time	Chick	Diet
1	42	0	1	1
2	51	2	1	1
3	59	4	1	1
4	64	6	1	1
5	76	8	1	1
6	93	10	1	1
7	106	12	1	1
8	125	14	1	1
9	149	16	1	1
10	171	18	1	1
11	199	20	1	1
12	205	21	1	1
13	40	0	2	1
14	49	2	2	1
15	58	4	2	1
16	72	6	2	1
17	84	8	2	1
18	103	10	2	1

For this study, the **Wight** column is taken as a numerical value and the **Diet column as** categorical value. The diet column shows a factor with levels 1to 4 indicating which experimental diet(Type of protein) the chick received.

Procedures followed



Project Result

Shows the category of the diet type, mean and variance result when the program is run on the entire population data, on the 25% of the population sample with out replacement and on the 100% resample of the 25% sample with replacement

==========**Printing Aggregate of the Pupulation** ==========

(1,(102.64545454545454,3195.3742975206624))
(2,(122.61666666666666,5084.903055555556))
(3,(142.95,7427.064166666667))
(4,(135.26271186440678,4697.244541798333))

==========**Printing Aggregate of the Sample** ==========

(1,(103.14035087719299,3172.7873191751314))
(2,(119.2,5448.21714285714))
(3,(140.14285714285714,9335.265306122452))
(4,(151.0,4309.18181818182))

==========**Printing Aggregate of the ReSample** ==========

(1,(104.01292161959496,3131.950994795693))
(2,(120.14451286125768,5460.8335986294705))
(3,(140.77024980606345,9055.914695128457))
(4,(154.58283672824112,3846.6066915170877))

Thank you !

Q & A