Appendix 2
Anonymizing numerical data by Scala programming

SSG_UDF.scala

```scala
import org.apache.spark.sql.functions._
import org.apache.spark.sql.Row
import scala.collection.JavaConversions._
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import java.io.PrintWriter;
//val sqlContext = new org.apache.spark.sql.SQLContext(sc)
//import sqlContext.implicits._
//import org.apache.spark.sql.functions.{udf, explode}

case class Rating(age: Int, job: String, marital: String, edu: String, social: String, race: String, sex: String, position: String, county: String, country: String, salary: String)
val ki=5 //the ki value
val distance=5
var rep=0
var all_intervals=""
var all_interval=""
var i=0
var left=0
var rep_total=0
var te=0
var w=0
val Result = sc.textFile("hdfs://zobbi01:8020/input/adult.csv").map(_.split(","))\n.map(p =>
```
//FILTER EACH CLASS VALUE TO CREATE G GROUPS

val f1=Result.filter("edu"==="Doctorate")

val f2=Result.filter("edu"==="Masters")

val f3=Result.filter("edu"==="Bachelors")

val f4=Result.filter("edu"==="Some-college")

val f5=Result.filter("edu"==="Assoc-voc")

val f6=Result.filter("edu"==="Prof-school")

val f7=Result.filter("edu"==="HS-grad")

val f8=Result.filter("edu"==="12th")

val f9=Result.filter("edu"==="11th")

val f10=Result.filter("edu"==="10th")

val f11=Result.filter("edu"==="9th")

val f12=Result.filter("edu"==="7th-8th")

val f13=Result.filter("edu"==="5th-6th")

val f14=Result.filter("edu"==="1st-4th")
val f15 = Result.filter("edu" === "Preschool")

//GROUP [STAGE ONE]
val SG1 = f1.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" >= ki)

val SSG1 = f1.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" < ki)

//GROUP [STAGE ONE]
val SG2 = f2.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" >= ki)

val SSG2 = f2.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" < ki)

//GROUP [STAGE ONE]
val SG3 = f3.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" >= ki)

val SSG3 = f3.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" < ki)

//GROUP [STAGE ONE]
val SG4 = f4.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" >= ki)

val SSG4 = f4.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" < ki)
"county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val SSG4 = f4.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SG5 = f5.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG5 = f5.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val SG6 = f6.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG6 = f6.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val SG7 = f7.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val SSG7 = f7.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SG8 = f8.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)
val SSG8 = f8.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < $ki)

val SG9 = f9.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= $ki)

val SSG9 = f9.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < $ki)

val SG10 = f10.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= $ki)

val SSG10 = f10.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < $ki)

val SG11 = f11.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= $ki)

val SSG11 = f11.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < $ki)

val SG12 = f12.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= $ki)

val SSG12 = f12.groupBy("age", "job", "marital", "edu").agg(collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < $ki)
val SG13 = f13.groupBy("age","job","marital","edu").agg(collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val SSG13 = f13.groupBy("age","job","marital","edu").agg(collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)

val SG14 = f14.groupBy("age","job","marital","edu").agg(collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val SSG14 = f14.groupBy("age","job","marital","edu").agg(collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)

val SG15 = f15.groupBy("age","job","marital","edu").agg(collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val SSG15 = f15.groupBy("age","job","marital","edu").agg(collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)

//ADJUST SG
```scala
def assertSameSize(arrs:Seq[_]*) = {
  assert(arrs.map(_.size).distinct.size==1,"sizes differ")
}

val zip1 = udf((xa:Seq[String],xb:Seq[String],xc:Seq[String],xd:Seq[String],xe:Seq[String],xf:Seq[String]) => {
  assertSameSize(xa,xb,xc,xd,xe,xf)
  xa.indices.map(i=> (xa(i),xb(i),xc(i),xd(i),xe(i),xf(i)))
})

val SG1_AD=SG1.withColumn("vars",
    explode(zip1($"social",$"race",$"position",$"county",$"country",$"salary"))).select($"age",
    $"job",$"marital",$"edu",$"vars._1".alias("social"),
    $"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country")
    ,$"vars._6".alias("salary"))

val SG2_AD=SG2.withColumn("vars",
    explode(zip1($"social",$"race",$"position",$"county",$"country",$"salary"))).select($"age",
    $"job",$"marital",$"edu",$"vars._1".alias("social"),
    $"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country")
    ,$"vars._6".alias("salary"))

val SG3_AD=SG3.withColumn("vars",
    explode(zip1($"social",$"race",$"position",$"county",$"country",$"salary"))).select($"age",
    $"job",$"marital",$"edu",$"vars._1".alias("social"),
    $"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country")
    ,$"vars._6".alias("salary"))

val SG4_AD=SG4.withColumn("vars",
    explode(zip1($"social",$"race",$"position",$"county",$"country",$"salary"))).select($"age",
    $"job",$"marital",$"edu",$"vars._1".alias("social"),
    $"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country")
    ,$"vars._6".alias("salary"))

val SG5_AD=SG5.withColumn("vars",
    explode(zip1($"social",$"race",$"position",$"county",$"country",$"salary"))).select($"age",
    $"job",$"marital",$"edu",$"vars._1".alias("social"),
    $"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country")
    ,$"vars._6".alias("salary"))

val SG6_AD=SG6.withColumn("vars",
    explode(zip1($"social",$"race",$"position",$"county",$"country",$"salary"))).select($"age",
    $"job",$"marital",$"edu",$"vars._1".alias("social"),
    $"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country")
    ,$"vars._6".alias("salary"))
```

val SG7_AD=SG7.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG8_AD=SG8.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG9_AD=SG9.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG10_AD=SG10.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG11_AD=SG11.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG12_AD=SG12.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG13_AD=SG13.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age","$job","$marital","$edu","$vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG14_AD=SG14.withColumn("vars", explode(zip1("$social","$race","$position","$county","$country","$salary"))).select("$age",
$"vars._1".alias("social"),$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
val SG15_AD=SG15.withColumn("vars", explode(zip1("social","race","position","county","country","salary"))).select("age", "job","marital","edu","vars._1".alias("social"),"vars._2".alias("race"),"vars._3".alias("position"),"vars._4".alias("county"),"vars._5".alias("country"),"vars._6".alias("salary"))

//STORE SG [first]
SG1_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/1")
SG1_AD.unpersist()
SG2_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/2")
SG2_AD.unpersist()
SG3_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/3")
SG3_AD.unpersist()
SG4_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/4")
SG4_AD.unpersist()
SG5_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/5")
SG5_AD.unpersist()
SG6_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/6")
SG6_AD.unpersist()
SG7_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/7")
SG7_AD.unpersist()
SG8_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/8")
SG8_AD.unpersist()
SG9_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/9")
SG9_AD.unpersist()
SG10_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/10")
SG10_AD.unpersist()
SG11_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/11")
SG11_AD.unpersist()
SG12_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/12")
SG12_AD.unpersist()
SG13_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/13")
SG13_AD.unpersist()
SG14_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/14")
SG14_AD.unpersist()
SG15_AD.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result/15")
SG15_AD.unpersist()

// ADJUST SSG
def assertSameSize(arrs:Seq[_]*) = {
  assert(arrs.map(_.size).distinct.size==1,"sizes differ")
}

val zip1_1 = udf((xa:Seq[String],xb:Seq[String],xc:Seq[String],xd:Seq[String],xe:Seq[String],xf:Seq[String]) =>
{  
  assertSameSize(xa,xb,xc,xd,xe,xf)
  xa.indices.map(i=> (xa(i),xb(i),xc(i),xd(i),xe(i),xf(i)))
})

val SSG1_AD=SSG1.withColumn("vars",
  explode(zip1_1($"social","race","position","county","country","salary"))).select("age",
  "job","marital","edu","vars._1".alias("social"),
  "vars._2".alias("race"),"vars._3".alias("position"),"vars._4".alias("county"),"vars._5".alias("country"),
  "vars._6".alias("salary"))
val SSG2_AD=SSG2.withColumn("vars",
  explode(zip1_1($"social","race","position","county","country","salary"))).select("age",}
val SSG3_AD = SSG3.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG4_AD = SSG4.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG5_AD = SSG5.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG6_AD = SSG6.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG7_AD = SSG7.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG8_AD = SSG8.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG9_AD = SSG9.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))

val SSG10_AD = SSG10.withColumn("vars", 
  explode(zip1_1($"social",$"race","position","county","country","salary"))).select($"age", $"job",$"marital",$"edu","vars._1".alias("social"),
$"vars._2".alias("race"),$"vars._3".alias("position"),$"vars._4".alias("county"),$"vars._5".alias("country"),
$"vars._6".alias("salary"))
```scala
val SSG11_AD = SSG11.withColumn("vars", explode(zip1_1("social", "race", "position", "county", "country", "salary"))).select("age", "job", "marital", "edu", "vars._1".alias("social"),
"vars._2".alias("race"), "vars._3".alias("position"), "vars._4".alias("county"), "vars._5".alias("country"),
"vars._6".alias("salary"))

val SSG12_AD = SSG12.withColumn("vars", explode(zip1_1("social", "race", "position", "county", "country", "salary"))).select("age", "job", "marital", "edu", "vars._1".alias("social"),
"vars._2".alias("race"), "vars._3".alias("position"), "vars._4".alias("county"), "vars._5".alias("country"),
"vars._6".alias("salary"))

val SSG13_AD = SSG13.withColumn("vars", explode(zip1_1("social", "race", "position", "county", "country", "salary"))).select("age", "job", "marital", "edu", "vars._1".alias("social"),
"vars._2".alias("race"), "vars._3".alias("position"), "vars._4".alias("county"), "vars._5".alias("country"),
"vars._6".alias("salary"))

val SSG14_AD = SSG14.withColumn("vars", explode(zip1_1("social", "race", "position", "county", "country", "salary"))).select("age", "job", "marital", "edu", "vars._1".alias("social"),
"vars._2".alias("race"), "vars._3".alias("position"), "vars._4".alias("county"), "vars._5".alias("country"),
"vars._6".alias("salary"))

val SSG15_AD = SSG15.withColumn("vars", explode(zip1_1("social", "race", "position", "county", "country", "salary"))).select("age", "job", "marital", "edu", "vars._1".alias("social"),
"vars._2".alias("race"), "vars._3".alias("position"), "vars._4".alias("county"), "vars._5".alias("country"),
"vars._6".alias("salary"))

//GROUP SSG [STAGE TWO]

val SSG1_2 = SSG1_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" >= ki)

val ASSG1_2 = SSG1_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where("cnt" < ki)
```
val SSG2_2 = SSG2_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG2_2 = SSG2_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG3_2 = SSG3_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG3_2 = SSG3_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG4_2 = SSG4_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG4_2 = SSG4_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG5_2 = SSG5_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG5_2 = SSG5_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)
val SSG6_2 = SSG6_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG6_2 = SSG6_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG7_2 = SSG7_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG7_2 = SSG7_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG8_2 = SSG8_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG8_2 = SSG8_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG9_2 = SSG9_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG9_2 = SSG9_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)

val SSG10_2 = SSG10_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" >= ki)

val ASSG10_2 = SSG10_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age", collect_list("social") as "social", collect_list("race") as "race", collect_list("position") as "position", collect_list("county") as "county", collect_list("country") as "country", collect_list("salary") as "salary", count("*") as "cnt").where($"cnt" < ki)
"position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val ASSG10_2= SSG10_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)


val SSG11_2= SSG11_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val ASSG11_2= SSG11_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)


val SSG12_2= SSG12_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val ASSG12_2= SSG12_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)


val SSG13_2= SSG13_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val ASSG13_2= SSG13_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)


val SSG14_2= SSG14_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)
val ASSG14_2= SSG14_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)

val SSG15_2= SSG15_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" >= ki)

val ASSG15_2= SSG15_AD.groupBy("job", "marital", "edu").agg(collect_list("age") as "age",collect_list("social") as "social",collect_list("race") as "race",collect_list("position") as "position",collect_list("county") as "county",collect_list("country") as "country",collect_list("salary") as "salary",count("*") as "cnt").where($"cnt" < ki)

//AGE ANONYMIZE

//ENABLE CARTESIAN JOIN
spark.conf.set("spark.sql.crossJoin.enabled", true)

val AnonUdf = udf((lists: Seq[Int]) => {
  val ascending = lists.sorted  //sorts in ascending order
  val length_of_Array=ascending.size
  val ki=5
  var objects_no=0
  var groups_num=length_of_Array/ki
  var minimum=ascending(0)-ascending(0)%5
  var medium=minimum + distance
  //var maximum=5-(ascending(ascending.size - 1)%5)+ascending(ascending.size - 1)
  if(groups_num >=1){

while(objects_no<length_of_Array){
    for (j <- 0 to ascending.size - 1){
        if(ascending(j)<medium && ascending(j)>=minimum){
            rep+=1
        }
    }
    if(rep==0){
        minimum=medium;
        medium=minimum+distance;
    }else{
        if(rep>=ki){
            rep_total=rep_total+rep
            left=ascending.size-rep_total
            if(left<ki){
                medium=5-(ascending(ascending.size - 1)%5)+ascending(ascending.size - 1)
                rep=rep+left
            }
        }
    }
    for (k <- 0 to rep-1){
        w+=1
        if(w==ascending.size){
            objects_no+=1
            all_intervals=all_intervals+"["+minimum+" - "+medium+"]"
        }else{
            objects_no+=1
            all_intervals=all_intervals+"["+minimum+" - "+medium+","
        }
    }
}
minimum=medium
medium=minimum+distance
te=rep_total-1
rep=0

}else{
medium=medium+5
rep=0
} }//end if
}}//end for(rep==0)
}}//end while
} }//end if(group_num>=1)
//down
s"${all_intervals}"
}
////////////////////END ANONYMIZE AGE

//6 columns
def assertSameSize(arrs:Seq[_]*) = {assert(arrs.map(_.size).distinct.size==1,"sizes differ")}

val zip4 = udf((xa:Seq[String],xb:Seq[String],xc:Seq[String],xd:Seq[String],xe:Seq[String],xf:Seq[String]) =>
{
  assertSameSize(xa,xb,xc,xd,xe,xf)
  xa.indices.map(i=> (xa(i),xb(i),xc(i),xd(i),xe(i),xf(i)))
//7 columns
val zip5 = udf((xa:Seq[String], xb:Seq[String], xc:Seq[String], xd:Seq[String], xe:Seq[String], xf:Seq[String], xg:Seq[String]) => {
    assertSameSize(xa, xb, xc, xd, xe, xf, xg)
    xa.indices.map(i => (xa(i), xb(i), xc(i), xd(i), xe(i), xf(i), xg(i)))
})

//WITHOUT JOIN
//Create Anonymization for >= ki
val intervals_1 = SSG1_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
 val SSG1_2_expand = intervals_1.withColumn("vars", explode(zip5(split(col("ages"), ","), "$social", "$race", "$position", "$county", "$country", "$salary"))).select("marital", "$edu", "$vars._1".alias("age"), "$vars._2".alias("social"), "$vars._3".alias("race"), "$vars._4".alias("position"), "$vars._5".alias("county"), "$vars._6".alias("country"), "$vars._7".alias("salary"))
SSG1_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/1")

val intervals_2 = SSG2_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "County", "country", "salary")
 val SSG2_2_expand = intervals_2.withColumn("vars", explode(zip5(split(col("ages"), ","), "$social", "$race", "$position", "$country", "$country", "$salary"))).select("marital", "$edu", "$vars._1".alias("age"), "$vars._2".alias("social"), "$vars._3".alias("race"), "$vars._4".alias("position"), "$vars._5".alias("county"), "$vars._6".alias("country"), "$vars._7".alias("salary"))
SSG2_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/2")
```scala
val intervals_3 = SSG3_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG3_2_expand = intervals_3.withColumn("vars", explode(zip5(split(col("ages"), ","), $"social", $"race", $"position", $"county", $"country", $"salary"))).select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))
SSG3_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/3")

val intervals_4 = SSG4_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG4_2_expand = intervals_4.withColumn("vars", explode(zip5(split(col("ages"), ","), $"social", $"race", $"position", $"county", $"country", $"salary"))).select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))
SSG4_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/4")

val intervals_5 = SSG5_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG5_2_expand = intervals_5.withColumn("vars", explode(zip5(split(col("ages"), ","), $"social", $"race", $"position", $"county", $"country", $"salary"))).select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))
SSG5_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/5")

val intervals_6 = SSG6_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG6_2_expand = intervals_6.withColumn("vars", explode(zip5(split(col("ages"), ","), $"social", $"race", $"position", $"county", $"country", $"salary"))).select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))
SSG6_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/6")
```
val intervals_7 = SSG7_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG7_2_expand = intervals_7.withColumn("vars", explode(zip5(split(col("ages"), ""), "$"social", "$"race", "$"position", "$"county", "$"country", "$"salary").select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))))

SSG7_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/7")

val intervals_8 = SSG8_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG8_2_expand = intervals_8.withColumn("vars", explode(zip5(split(col("ages"), ""), "$"social", "$"race", "$"position", "$"county", "$"country", "$"salary").select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))))

SSG8_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/8")

val intervals_9 = SSG9_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG9_2_expand = intervals_9.withColumn("vars", explode(zip5(split(col("ages"), ""), "$"social", "$"race", "$"position", "$"county", "$"country", "$"salary").select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))))

SSG9_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/9")

val intervals_10 = SSG10_2.withColumn("ages", AnonUdf($"age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG10_2_expand = intervals_10.withColumn("vars", explode(zip5(split(col("ages"), ""), "$"social", "$"race", "$"position", "$"county", "$"country", "$"salary").select($"marital", $"edu", $"vars._1".alias("age"), $"vars._2".alias("social"), $"vars._3".alias("race"), $"vars._4".alias("position"), $"vars._5".alias("county"), $"vars._6".alias("country"), $"vars._7".alias("salary"))))

SSG10_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/10")
val intervals_11 = SSG11_2.withColumn("ages", AnonUdf("$age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG11_2_expand = intervals_11.withColumn("vars", explode(zip5(split(col("ages"),""),"$social", "$race","$position","$county","$country","$salary"))).select("$marital","$edu","vars._1".alias("age"), $"vars._2".alias("social"),$"vars._3".alias("race"),$"vars._4".alias("position"),$"vars._5".alias("county"),$ "vars._6".alias("country"),$"vars._7".alias("salary"))
SG11_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/11")

val intervals_12 = SSG12_2.withColumn("ages", AnonUdf("$age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG12_2_expand = intervals_12.withColumn("vars", explode(zip5(split(col("ages"),""),"$social", "$race","$position","$county","$country","$salary"))).select("$marital","$edu","vars._1".alias("age"), $"vars._2".alias("social"),$"vars._3".alias("race"),$"vars._4".alias("position"),$"vars._5".alias("county"),$ "vars._6".alias("country"),$"vars._7".alias("salary"))
SG12_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/12")

val intervals_13 = SSG13_2.withColumn("ages", AnonUdf("$age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG13_2_expand = intervals_13.withColumn("vars", explode(zip5(split(col("ages"),""),"$social", "$race","$position","$county","$country","$salary"))).select("$marital","$edu","vars._1".alias("age"), $"vars._2".alias("social"),$"vars._3".alias("race"),$"vars._4".alias("position"),$"vars._5".alias("county"),$ "vars._6".alias("country"),$"vars._7".alias("salary"))
SG13_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/13")

val intervals_14 = SSG14_2.withColumn("ages", AnonUdf("$age").select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")
val SSG14_2_expand = intervals_14.withColumn("vars", explode(zip5(split(col("ages"),""),"$social", "$race","$position","$county","$country","$salary"))).select("$marital","$edu","vars._1".alias("age"), $"vars._2".alias("social"),$"vars._3".alias("race"),$"vars._4".alias("position"),$"vars._5".alias("county"),$ "vars._6".alias("country"),$"vars._7".alias("salary"))
SG14_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/14")
val intervals_15 = SSG15_2.withColumn("ages", AnonUdf($"age$)).select("ages", "marital", "edu", "social", "race", "position", "county", "country", "salary")

val SSG15_2_expand = intervals_15.withColumn("vars", explode(zip5(split(col("ages"),""), "$"social", "$"race", "$"position", "$"county", "$"country", "$"salary$))).select($"marital", $"edu", $"vars._1$.alias("age"), $"vars._2$.alias("social"), $"vars._3$.alias("race"), $"vars._4$.alias("position"), $"vars._5$.alias("county"), $"vars._6$.alias("country"), $"vars._7$.alias("salary"))

SSG15_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/15")

//val SSG1_2_anon = SSG1_2.withColumn("age", AnonUdf($"age$)).select("age")
//val SSG1_2_anon_p1 = SSG1_2_anon.withColumn("age", explode(split(col("age"), ",")))  
//val SSG1_2_expand_p2 = SSG1_2.withColumn("vars", explode(zip4($"social", $"race", $"position", $"county", $"country", $"salary$))).select($"marital", $"edu", $"vars._1$.alias("social"), $"vars._2$.alias("race"), $"vars._3$.alias("position"), $"vars._4$.alias("county"), $"vars._5$.alias("country"), $"vars._6$.alias("salary"))

//val SSG1_2_join = SSG1_2_anon_p1.join(SSG1_2_expand_p2)
//SSG1_2_join.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SG/result2/1")

//Store the < ki after expanding them

val ASSG1_2_expand = ASSG1_2.withColumn("vars", explode(zip5($"age", $"social", $"race", $"position", $"county", $"country", $"salary"))).select($"job", $"marital", $"edu", $"vars._1$.alias("age"), $"vars._2$.alias("social"), $"vars._3$.alias("race"), $"vars._4$.alias("position"), $"vars._5$.alias("county"), $"vars._6$.alias("country"), $"vars._7$.alias("salary"))

val ASSG2_2_expand = ASSG2_2.withColumn("vars", explode(zip5($"age", $"social", $"race", $"position", $"county", $"country", $"salary"))).select($"job", $"marital", $"edu", $"vars._1$.alias("age"), $"vars._2$.alias("social"), $"vars._3$.alias("race"), $"vars._4$.alias("position"), $"vars._5$.alias("county"), $"vars._6$.alias("country"), $"vars._7$.alias("salary"))
val ASSG3_2_expand = ASSG3_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG4_2_expand = ASSG4_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG5_2_expand = ASSG5_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG6_2_expand = ASSG6_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG7_2_expand = ASSG7_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG8_2_expand = ASSG8_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG9_2_expand = ASSG9_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG10_2_expand = ASSG10_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))
val ASSG11_2_expand = ASSG11_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG12_2_expand = ASSG12_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG13_2_expand = ASSG13_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG14_2_expand = ASSG14_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

val ASSG15_2_expand = ASSG15_2.withColumn("vars", explode(zip5("age", "social", "race", "position", "county", "country", "salary"))).select("job", "marital", "edu", "vars._1".alias("age"), "vars._2".alias("social"), "vars._3".alias("race"), "vars._4".alias("position"), "vars._5".alias("county"), "vars._6".alias("country"), "vars._7".alias("salary"))

ASSG1_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/1")

ASSG2_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/2")

ASSG3_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/3")

ASSG4_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/4")

ASSG5_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/5")

ASSG6_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/6")
ASSG7_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/7")

ASSG8_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/8")

ASSG9_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/9")

ASSG10_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/10")

ASSG11_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/11")

ASSG12_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/12")

ASSG13_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/13")

ASSG14_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/14")

ASSG15_2_expand.write.format("com.databricks.spark.csv").save("hdfs://zobbi01:8020/input/spark/SSG/15")

igest/END UDF