```
#Codificación utf-8

from pyspark import SparkContext, SparkConf
conf = SparkConf().setAppName("Pruebas").setMaster("local")
sc = SparkContext(conf=conf)

from pyspark.sql import SparkSession
from pyspark.sql.functions import col, desc, rtrim, expr
from pyspark.sql.types import StructField, StructType, StringType, IntegerType, DoubleType,
FloatType

sparkSQL = SparkSession \
    .builder \
    .appName("SparkSQL") \
    .master("local") \
    .getOrCreate()


mcp = sc.textFile("hdfs://localhost:9000/ficheros/PECMunicipios.csv")
elec = sc.textFile("hdfs://localhost:9000/ficheros/PECElecciones.csv")

#Creamos una funcion para poder quitar la cabecera del fichero
#Lo que hacemos es tener el indice de particion y si es 0 la sacamos
def f(idx, iter):
    output=[]
    for sublist in iter:
        output.append(sublist)
    if idx>0:
        return(output)
    else:
        return(output[1:])


def p(x): print(x)


#Utilizamos la funcion para quitar la cabecera
data = mcp.map(lambda x : x.replace("\"","").split(";")).mapPartitionsWithIndex(f)
data2 = elec.map(lambda x : x.replace("\"","").split(";")).mapPartitionsWithIndex(f)

print(data.take(10))
print(data2.take(10))
#Opción 1, hacer una join entre los dos rdds
#Requiere tener dos clases diferentes

datajoin = data.join(data2)
print(datajoin.take(10))

#Opción 2, dar de alta dos tablas y hacer join

schema = StructType([
    StructField('code', StringType(), False),
    StructField('Autonomia', StringType(), False),
    StructField('Provincia', StringType(), False),
    StructField('Municipio', StringType(), False),
    StructField('Poblacion', StringType(), False)
])

df = sparkSQL.createDataFrame(data, schema)
df = df.withColumn("code", df["code"].cast(IntegerType()))
df = df.withColumn("Poblacion", df["Poblacion"].cast(IntegerType()))
df = df.withColumn("Autonomia", rtrim(df["Autonomia"]))
df.show()

schema2 = StructType([
    StructField('code', StringType(), False),
    StructField('Mesas', StringType(), False),
    StructField('Censo', StringType(), False),
```

```
    StructField('Votantes', StringType(), False),
    StructField('Validos', StringType(), False),
    StructField('Blanco', StringType(), False),
    StructField('Nulos', StringType(), False),
    StructField('PP', StringType(), False),
    StructField('PSOE', StringType(), False),
    StructField('PODEMOS-IU-EQUO', StringType(), False),
    StructField('CIUDADANOS', StringType(), False),
    StructField('ECP', StringType(), False),
    StructField('PODEMOS-COMPROMIS-EUPV', StringType(), False),
    StructField('ERC-CATSI', StringType(), False),
    StructField('CDC', StringType(), False),
    StructField('PODEMOS-EN MAREA-ANOVA-EU', StringType(), False),
    StructField('EAJ-PNV', StringType(), False),
    StructField('EH-BILDU', StringType(), False),
    StructField('CCA-PNC', StringType(), False),
    StructField('PACMA', StringType(), False),
    StructField('RECORTES CERO-GRUPO VERDE', StringType(), False),
    StructField('UPYD', StringType(), False),
    StructField('VOX', StringType(), False),
    StructField('BNG-NOS', StringType(), False),
    StructField('PCPE', StringType(), False),
    StructField('GBAI', StringType(), False),
    StructField('EB', StringType(), False),
    StructField('FE DE LAS JONS', StringType(), False),
    StructField('SI', StringType(), False),
    StructField('SOMVAL', StringType(), False),
    StructField('CCD', StringType(), False),
    StructField('SAIN', StringType(), False),
    StructField('PH', StringType(), False),
    StructField('CENTRO MODERADO', StringType(), False),
    StructField('P-LIB', StringType(), False),
    StructField('CCD-CI', StringType(), False),
    StructField('UPL', StringType(), False),
    StructField('PCOE', StringType(), False),
    StructField('AND', StringType(), False),
    StructField('JXC', StringType(), False),
    StructField('PFYV', StringType(), False),
    StructField('CILUS', StringType(), False),
    StructField('PXC', StringType(), False),
    StructField('MAS', StringType(), False),
    StructField('IZAR', StringType(), False),
    StructField('UNIDAD DEL PUEBLO', StringType(), False),
    StructField('PREPAL', StringType(), False),
    StructField('LN', StringType(), False),
    StructField('REPO', StringType(), False),
    StructField('INDEPENDIENTES-FIA', StringType(), False),
    StructField('ENTABAN', StringType(), False),
    StructField('IMC', StringType(), False),
    StructField('PUEDE', StringType(), False),
    StructField('FE', StringType(), False),
    StructField('ALCD', StringType(), False),
    StructField('FME', StringType(), False),
    StructField('HRTS-LN', StringType(), False),
    StructField('UDT', StringType(), False)
])

df2 = sparkSQL.createDataFrame(data2, schema2)
df2.show()

df.registerTempTable('municipios')
df2.registerTempTable('votaciones')

joined = sparkSQL.sql('select * from municipios t1 join votaciones t2 on t1.code=t2.code order by
1')

#Este stack es incompleto porque no tiene todos los partidos
unpivot = joined.select('t1.code', 'Municipio', expr("stack(3, 'PP', PP, 'PSOE', PSOE,
'CIUDADANOS', CIUDADANOS) as (Partido, Votos)"))
```

```
unpivot = unpivot.withColumn('Votos', unpivot["Votos"].cast(IntegerType()))
#unpivot.repartition(1).write.format('csv') \
#    .option("delimiter", ";").save("hdfs://localhost:9000/ficheros/elecciones_out.csv")


#unpivot.filter(col('t1.code')
<2000).filter(col('Partido')=='PP').repartition(1).write.format('csv') \
#    .option("delimiter", ";").save("hdfs://localhost:9000/ficheros/elecciones_out.csv")

unpivot.groupBy('Partido').sum('Votos').show()
```