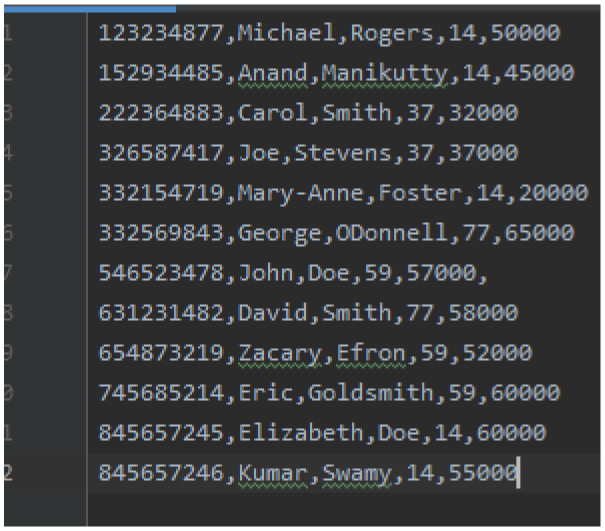
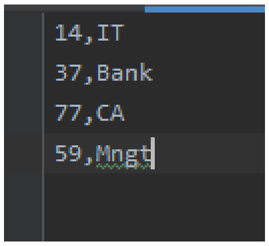
Emp1.txt and dept1.xt

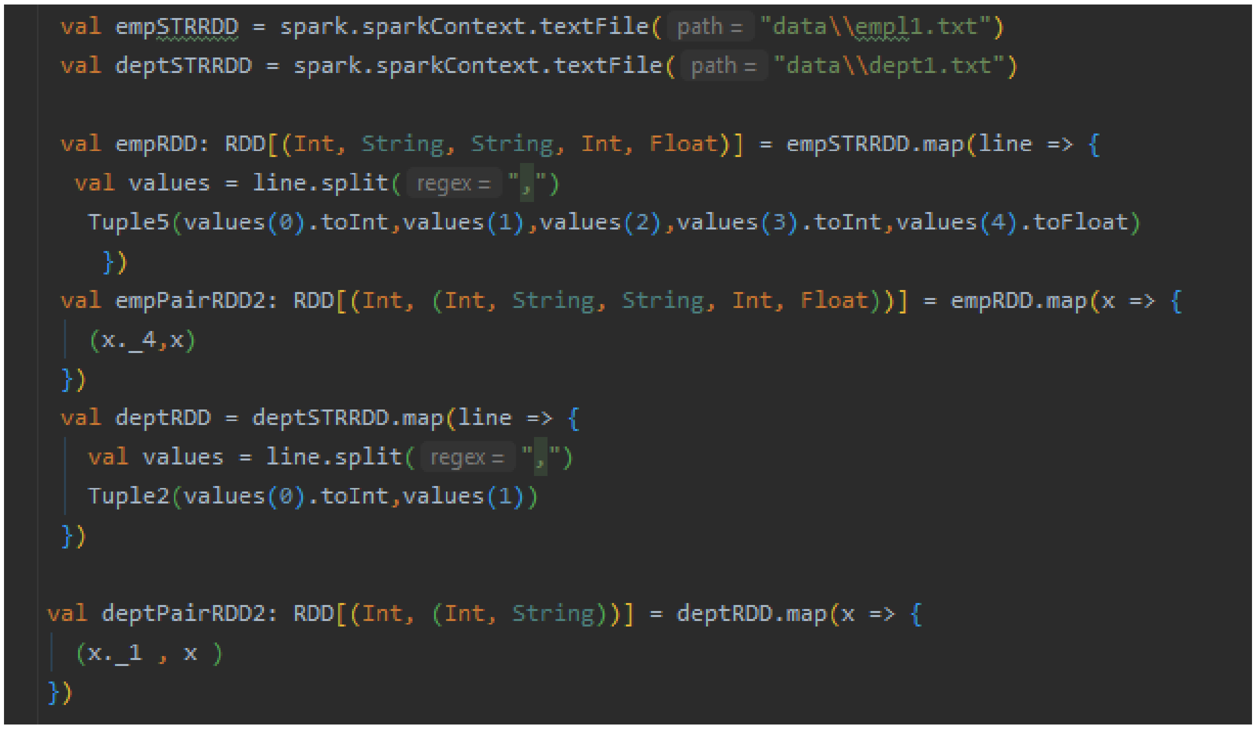


```
123234877,Michael,Rogers,14,50000
152934485,Anand,Manikutty,14,45000
222364883,Carol,Smith,37,32000
326587417,Joe,Stevens,37,37000
332154719,Mary-Anne,Foster,14,20000
332569843,George,ODonnell,77,65000
546523478,John,Doe,59,57000,
631231482,David,Smith,77,58000
654873219,Zacary,Efron,59,52000
745685214,Eric,Goldsmith,59,60000
845657245,Elizabeth,Doe,14,60000
845657246,Kumar,Swamy,14,55000
```



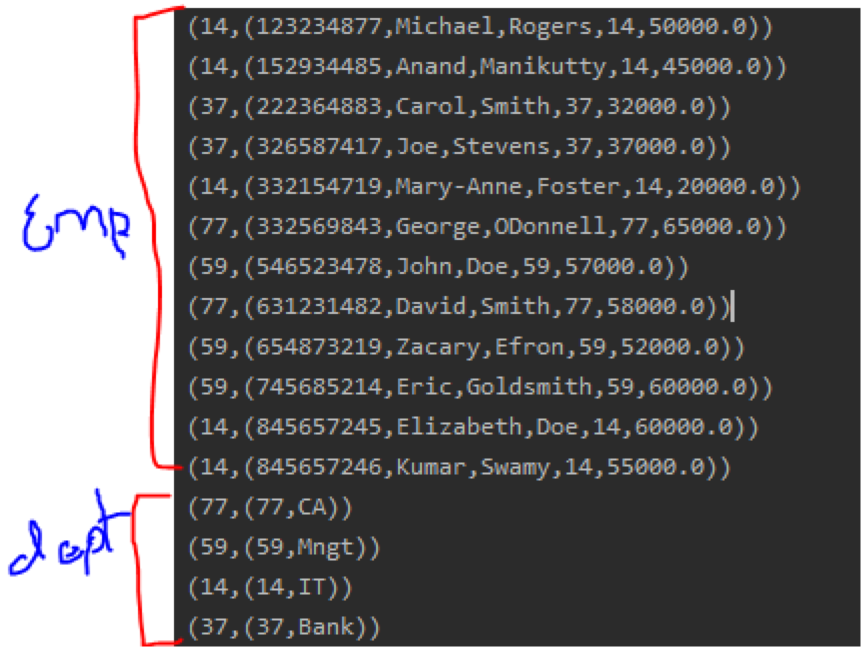
```
14,IT
37,Bank
77,CA
59,Mngt
```

RDD approach

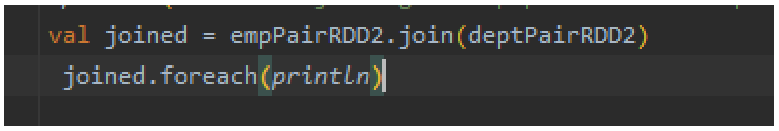


```
 val empSTRRDD = spark.sparkContext.textFile( path = "data\\empl1.txt")
 val deptSTRRDD = spark.sparkContext.textFile( path = "data\\dept1.txt")

 val empRDD: RDD[(Int, String, String, Int, Float)] = empSTRRDD.map(line => {
  val values = line.split( regex = ",")
   Tuple5(values(0).toInt,values(1),values(2),values(3).toInt,values(4).toFloat)
    })
 val empPairRDD2: RDD[(Int, (Int, String, String, Int, Float))] = empRDD.map(x => {
   (x._4,x)
 })
 val deptRDD = deptSTRRDD.map(line => {
   val values = line.split( regex = ",")
   Tuple2(values(0).toInt,values(1))
 })

val deptPairRDD2: RDD[(Int, (Int, String))] = deptRDD.map(x => {
  (x._1 , x )
})
```

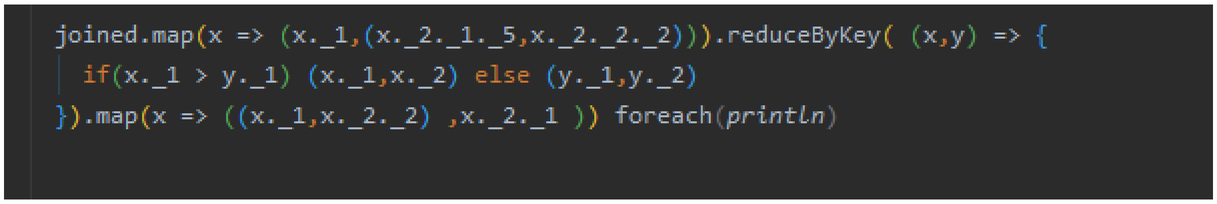
Emp pair RDD and Dept pair RDD is generated with dept no as key.



```
(14,(123234877,Michael,Rogers,14,50000.0))
(14,(152934485,Anand,Manikutty,14,45000.0))
(37,(222364883,Carol,Smith,37,32000.0))
(37,(326587417,Joe,Stevens,37,37000.0))
(14,(332154719,Mary-Anne,Foster,14,20000.0))
(77,(332569843,George,ODonnell,77,65000.0))
(59,(546523478,John,Doe,59,57000.0))
(77,(631231482,David,Smith,77,58000.0))
(59,(654873219,Zacary,Efron,59,52000.0))
(59,(745685214,Eric,Goldsmith,59,60000.0))
(14,(845657245,Elizabeth,Doe,14,60000.0))
(14,(845657246,Kumar,Swamy,14,55000.0))
(77,(77,CA))
(59,(59,Mngt))
(14,(14,IT))
(37,(37,Bank))
```



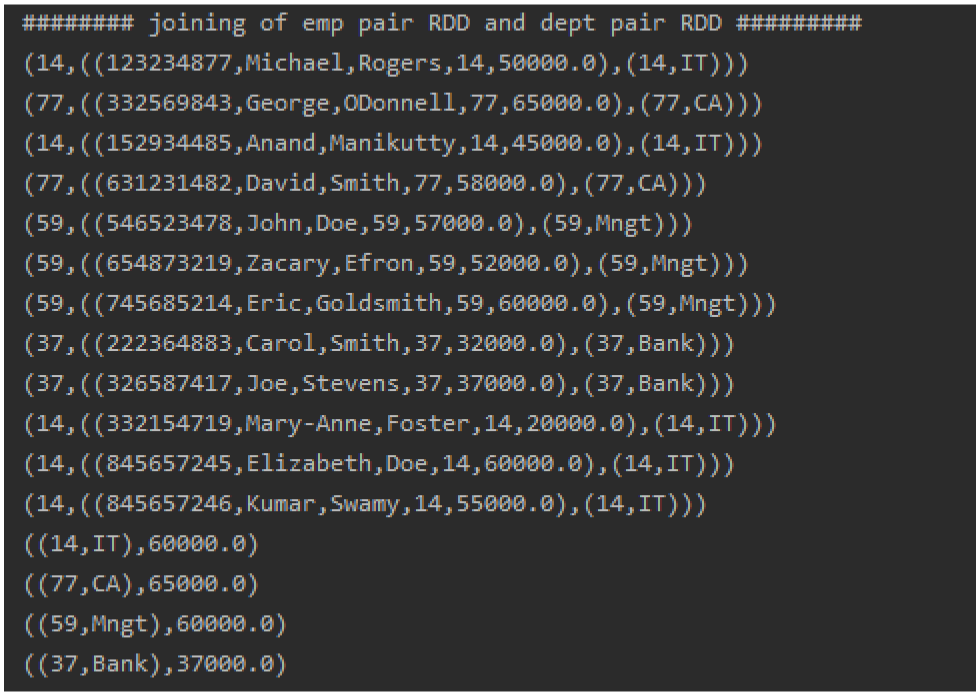
```
val joined = empPairRDD2.join(deptPairRDD2)
 joined.foreach(println)
```



```
joined.map(x => (x._1,(x._2._1._5,x._2._2._2))).reduceByKey( (x,y) => {
  if(x._1 > y._1) (x._1,x._2) else (y._1,y._2)
}).map(x => ((x._1,x._2._2) ,x._2._1 )) foreach(println)
```

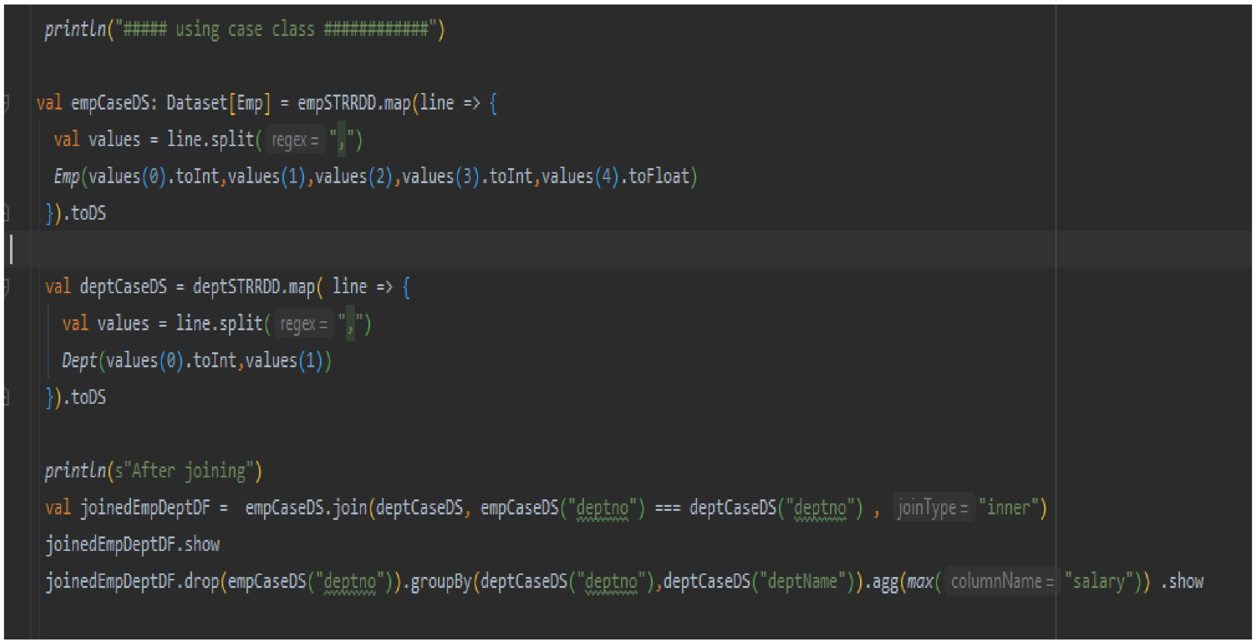
Both emp pair RDD and dept pair RDD are joined with dept key . Then map used select only dept no , salary and deptName , after that reducebyKey is applied to find maximum salary.

Finally map is again applied to arrange output



```
######## joining of emp pair RDD and dept pair RDD #########
(14,((123234877,Michael,Rogers,14,50000.0),(14,IT)))
(77,((332569843,George,ODonnell,77,65000.0),(77,CA)))
(14,((152934485,Anand,Manikutty,14,45000.0),(14,IT)))
(77,((631231482,David,Smith,77,58000.0),(77,CA)))
(59,((546523478,John,Doe,59,57000.0),(59,Mngt)))
(59,((654873219,Zacary,Efron,59,52000.0),(59,Mngt)))
(59,((745685214,Eric,Goldsmith,59,60000.0),(59,Mngt)))
(37,((222364883,Carol,Smith,37,32000.0),(37,Bank)))
(37,((326587417,Joe,Stevens,37,37000.0),(37,Bank)))
(14,((332154719,Mary-Anne,Foster,14,20000.0),(14,IT)))
(14,((845657245,Elizabeth,Doe,14,60000.0),(14,IT)))
(14,((845657246,Kumar,Swamy,14,55000.0),(14,IT)))
((14,IT),60000.0)
((77,CA),65000.0)
((59,Mngt),60000.0)
((37,Bank),37000.0)
```

Dataframe approach :

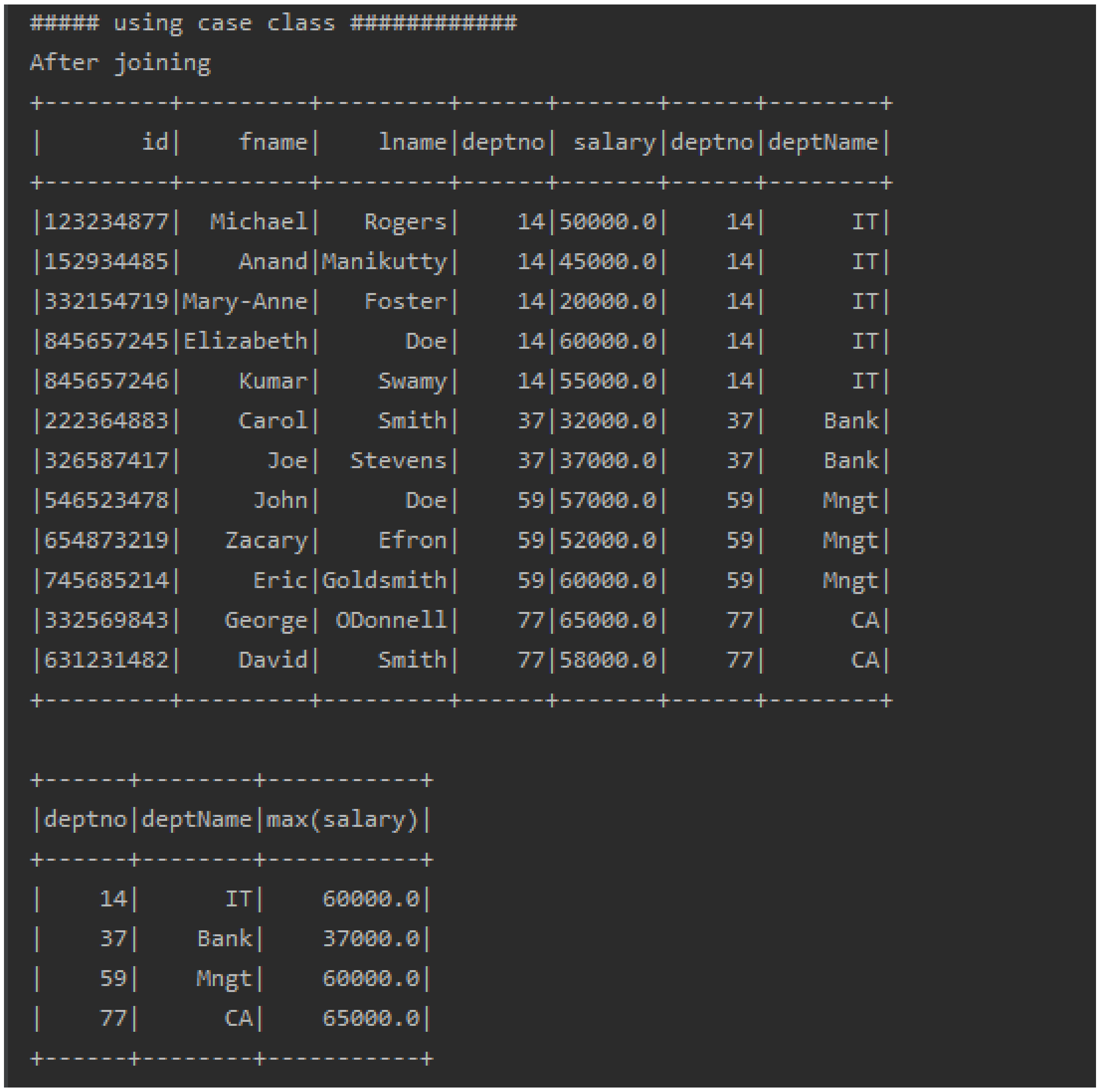


```
println("##### using case class ###########")

val empCaseDS: Dataset[Emp] = empSTRRDD.map(line => {
  val values = line.split( regex = ",")
  Emp(values(0).toInt,values(1),values(2),values(3).toInt,values(4).toFloat)
 }).toDS

 val deptCaseDS = deptSTRRDD.map( line => {
   val values = line.split( regex = ",")
   Dept(values(0).toInt,values(1))
 }).toDS

 println(s"After joining")
 val joinedEmpDeptDF =  empCaseDS.join(deptCaseDS, empCaseDS("deptno") === deptCaseDS("deptno") , joinType = "inner")
 joinedEmpDeptDF.show
 joinedEmpDeptDF.drop(empCaseDS("deptno")).groupBy(deptCaseDS("deptno"),deptCaseDS("deptName")).agg(max( columnName = "salary")) .show
```

DataSets of Emp and Dept case class are created.

Dataset are joined on deptno using inner option. Resultant dataframe has two times deptno so dropping one column coming from emp. Finally groupBy function followed by agg and max function applied

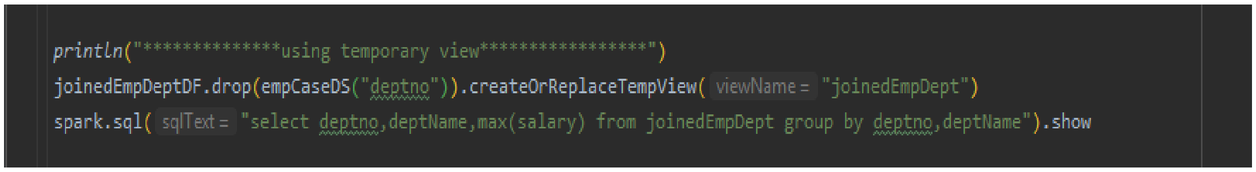
Below is output after joining and groping then aggregation



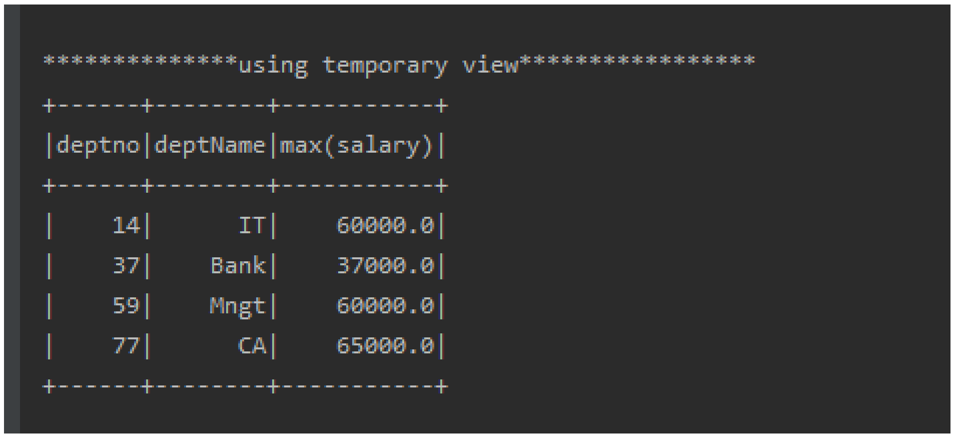
```
##### using case class ############
After joining
+---------+---------+---------+------+-------+------+--------+
|       id|    fname|    lname|deptno| salary|deptno|deptName|
+---------+---------+---------+------+-------+------+--------+
|123234877|  Michael|   Rogers|    14|50000.0|    14|      IT|
|152934485|    Anand|Manikutty|    14|45000.0|    14|      IT|
|332154719|Mary-Anne|   Foster|    14|20000.0|    14|      IT|
|845657245|Elizabeth|      Doe|    14|60000.0|    14|      IT|
|845657246|    Kumar|    Swamy|    14|55000.0|    14|      IT|
|222364883|    Carol|    Smith|    37|32000.0|    37|    Bank|
|326587417|      Joe|  Stevens|    37|37000.0|    37|    Bank|
|546523478|     John|      Doe|    59|57000.0|    59|    Mngt|
|654873219|   Zacary|    Efron|    59|52000.0|    59|    Mngt|
|745685214|     Eric|Goldsmith|    59|60000.0|    59|    Mngt|
|332569843|   George| ODonnell|    77|65000.0|    77|      CA|
|631231482|    David|    Smith|    77|58000.0|    77|      CA|
+---------+---------+---------+------+-------+------+--------+


+------+--------+-----------+
|deptno|deptName|max(salary)|
+------+--------+-----------+
|    14|      IT|    60000.0|
|    37|    Bank|    37000.0|
|    59|    Mngt|    60000.0|
|    77|      CA|    65000.0|
+------+--------+-----------+
```
.

Query approach

Joined dataframe of previous approach can be registered temporarily . spark.sql is used to write basic sql query for group by and max(salary).



```
println("**************using temporary view*****************")
joinedEmpDeptDF.drop(empCaseDS("deptno")).createOrReplaceTempView( viewName = "joinedEmpDept")
spark.sql( sqlText = "select deptno,deptName,max(salary) from joinedEmpDept group by deptno,deptName").show
```



```
**************using temporary view*****************
+------+--------+-----------+
|deptno|deptName|max(salary)|
+------+--------+-----------+
|    14|      IT|    60000.0|
|    37|    Bank|    37000.0|
|    59|    Mngt|    60000.0|
|    77|      CA|    65000.0|
+------+--------+-----------+
```

.