```
import org.apache.spark.SparkContext
import org.apache.spark.SparkConf
import org.apache.spark.storage.StorageLevel
val conf = new SparkConf()

val ordersRDD = sc.textFile("file:///root/retail_db/orders")

val order_itemsRDD = sc.textFile("file:///root/retail_db/order_items")

val orders_completedRDD = ordersRDD.filter( rec => (rec.split(",")
(3).equals("COMPLETE")))

or

val orders_completedRDD = ordersRDD.filter( rec => (rec.split(",")(3)
== “COMPLETE"))


val ordersmap = orders_completedRDD.map( rec => (rec.split(",")
(0).toInt,rec.split(",")(1).toString))


val order_itemsmap = order_itemsRDD.map( rec => (rec.split(",")
(1).toInt,rec.split(",")(4).toFloat))

val order_items_rev_per_dayRDD = order_itemsmap.reduceByKey((acc,
value) => acc + value)

val ordersJoin = ordersmap.join(order_items_rev_per_dayRDD)

val ordersJoinmap = ordersJoin.map( rec => (rec._2._1,rec._2._2))


val revperdayRDD = ordersJoinmap.aggregateByKey((0.0,0))( (acc, value)
=> (acc._1 + value, acc._2 + 1), (total1,total2) => (total1._1 +
total2._1, total1._2 + total2._2))

val AvgrevperdayRDD = revperdayRDD.map( rec => ( rec._1,
BigDecimal(rec._2._1/rec._2._2).setScale(2,
BigDecimal.RoundingMode.HALF_UP).toFloat))

val avgrevperdaysorted = AvgrevperdayRDD.sortByKey()

 val avgrevperdayCSV = avgrevperdaysorted.map( rec => "%s,
%s".format(rec._1,rec._2))

or

val avgrevperdayCSV = avgrevperdaysorted.map( rec => rec._1 + "," +
rec._2 )
```