Task4 Approach:

The goal of the distributed join was to execute the join operations locally.

First approach:
I partitioned the customer pairRDD.
I created a HashMap from the orders file, where the key was the O_CUSTKEY and the Value was all the O_COMMENT for this key, separated by |.
Then I used sc.Broadcast so that every worker has the whole HashMap locally and execute the join locally.
This worked perfectly with the small dataset, but when I used the larger the execution became slower and I had to set memory variables higher to make it work.
In this case, the broadcast variable was too large to fit the memory so I followed the next(second) approach.

Second approach:
I partitioned the customerpairrdd where the key is the C_CUSTKEY and as value I put just an empty string, because I do not need anything more for the join.
I created a pairrdd for the orders, where the key for each Tuple2 of this rdd is the O_CUSTKEY and the Values are all the O_COMMENT, separated by |.
Then I also partitioned this rdd.
I used persist in Memory so that these are not computed again.
Then, with the zipPartitions I had two Iterators one on the customer's partitioned rdd and 1 on the order's.
So the join is executed locally on each worker.
In this case I persist on each node a partition of the order's pairrdd, so in terms of memory this approach better.