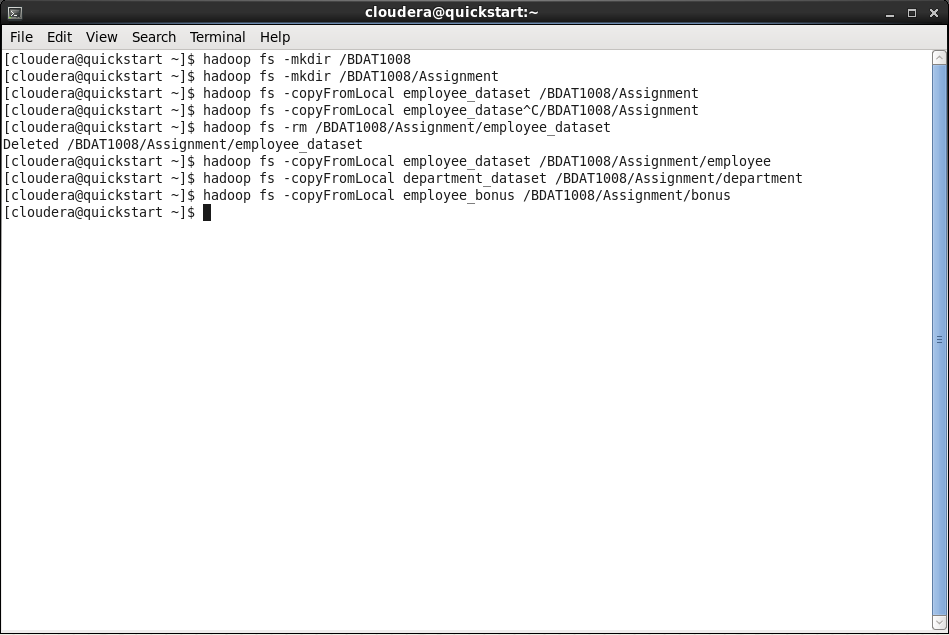
# Part-1



# Part-2

Start spark-shell:

spark-shell --master yarn --jars commons-csv-1.5.jar,spark-csv\_2.10-1.5.0.jar

Then import following packages and make sqlContext:

import org.apache.spark.sql.functions.\_

import org.apache.spark.sql.types.\_

import org.apache.spark.sql.SQLContext

val sqlContext = new SQLContext(sc)

1. **Create a relation with employee dataset with schema (column names and datatype) and name it employee. Once created, describe the relation.**

val employeeSchema = StructType(Array(

StructField("emp\_id", IntegerType, true),

StructField("emp\_name", StringType, true),

StructField("emp\_job\_desc", StringType, true),

StructField("dep\_id", StringType, true),

StructField("emp\_sal", FloatType, true)))

val employee = sqlContext.read

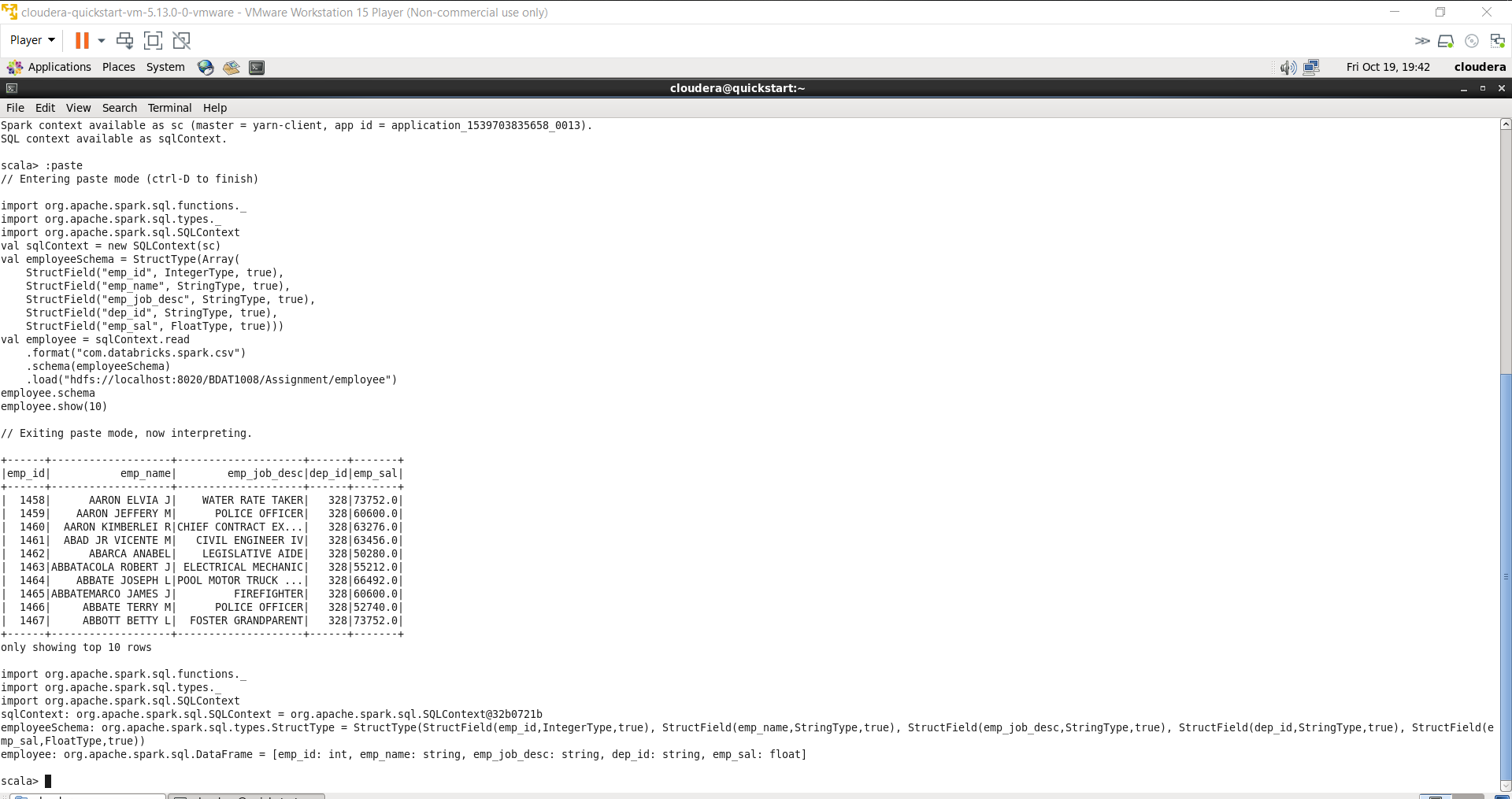
.format("com.databricks.spark.csv")

.schema(employeeSchema)

.load("hdfs://localhost:8020/BDAT1008/Assignment/employee")

employee.schema

employee.show(10)

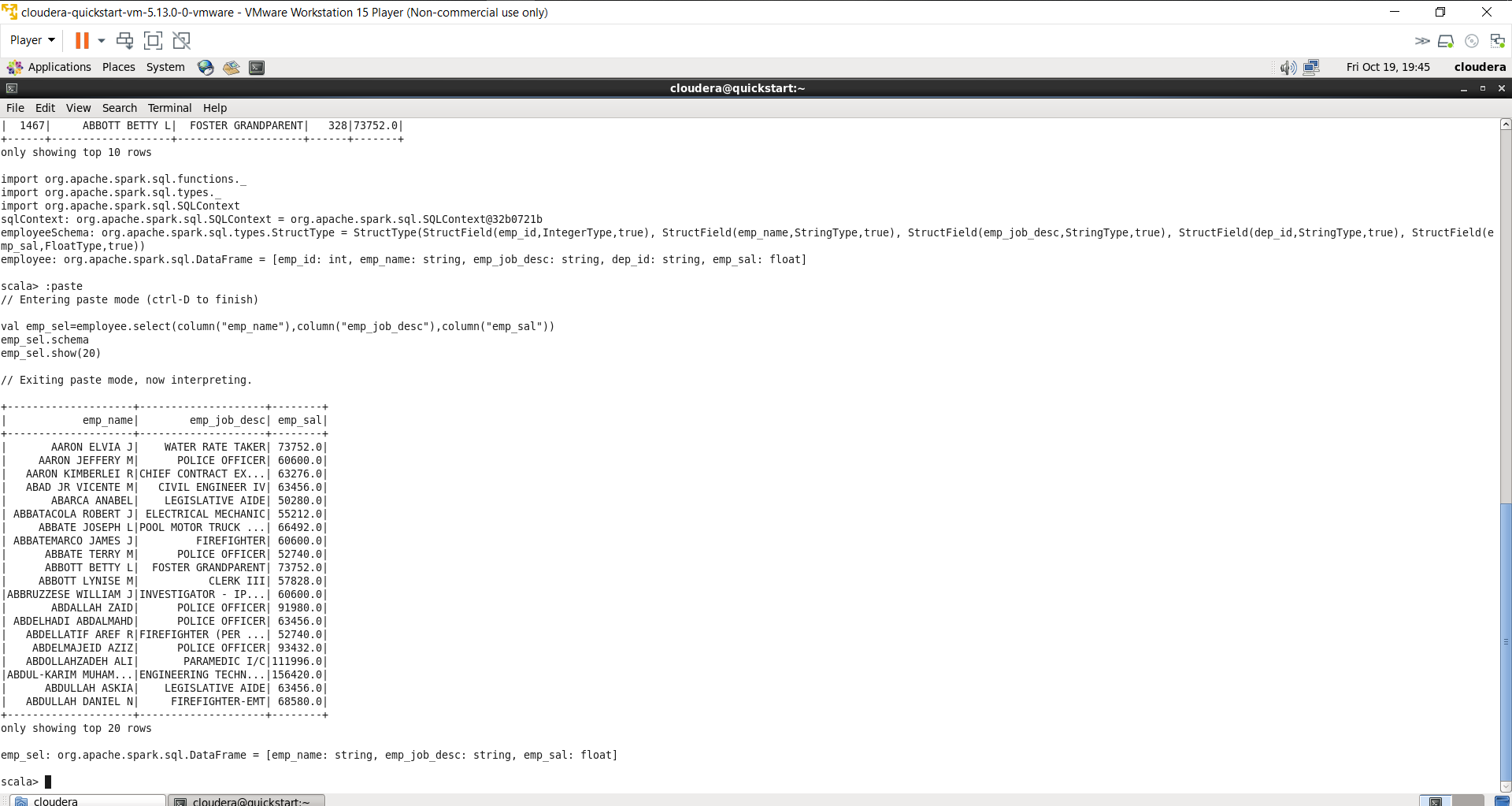


1. **Select the columns from the employee relation. Display 10 of these records on your screen.**

val emp\_sel=employee.select(column("emp\_name"),column("emp\_job\_desc"),column("emp\_sal"))

emp\_sel.schema

emp\_sel.show(20)



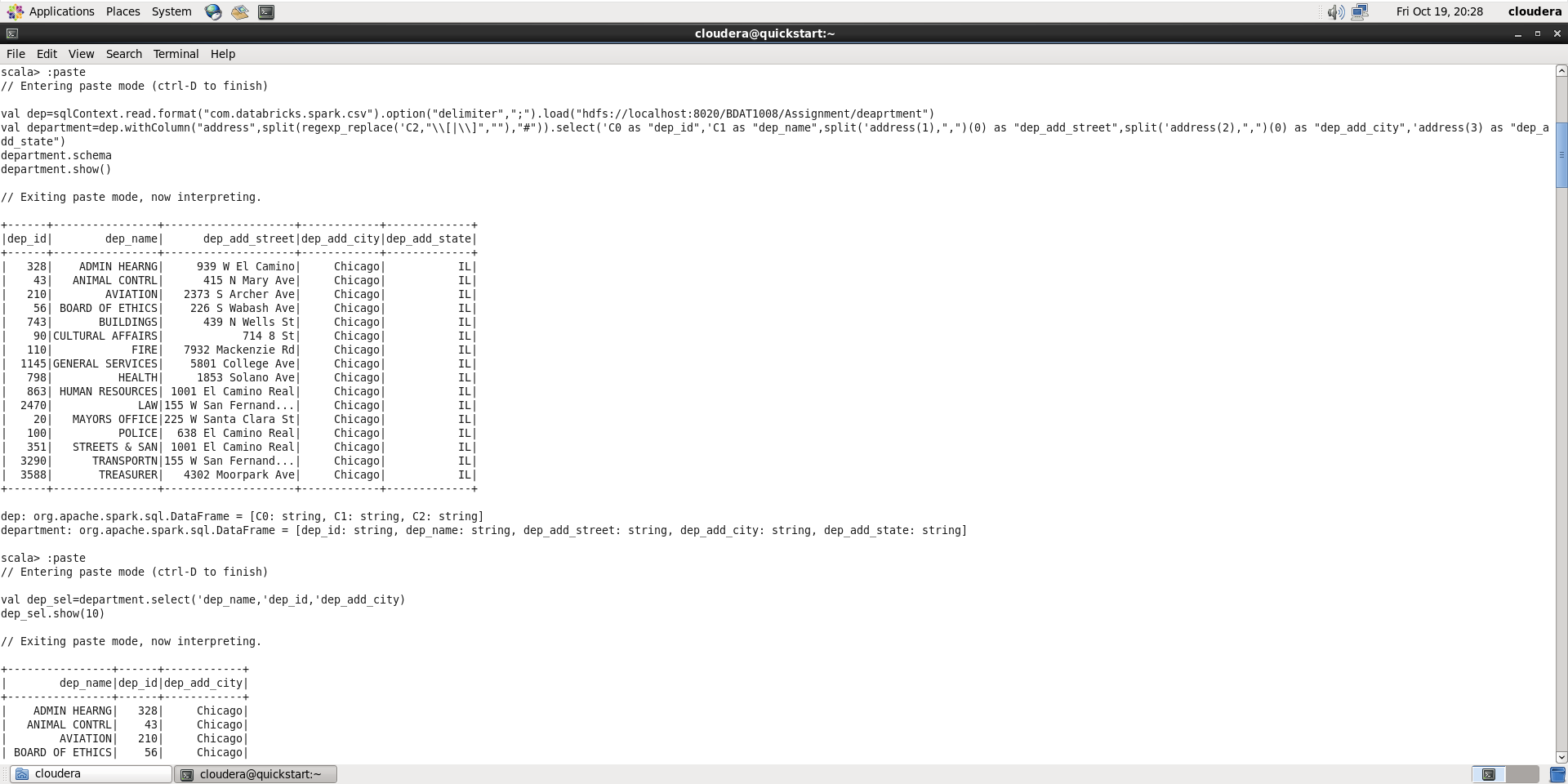
## Create a relation with department dataset with schema (column names and datatype) and name it departments. Once created, describe the relation.

val dep=sqlContext.read.format("com.databricks.spark.csv").option("delimiter",";").load("hdfs://localhost:8020/BDAT1008/Assignment/deaprtment")

val department=dep.withColumn("address",split(regexp\_replace('C2,"\\[|\\]",""),"#")).select('C0 as "dep\_id",'C1 as "dep\_name",split('address(1),",")(0) as "dep\_add\_street",split('address(2),",")(0) as "dep\_add\_city",'address(3) as "dep\_add\_state")

department.schema

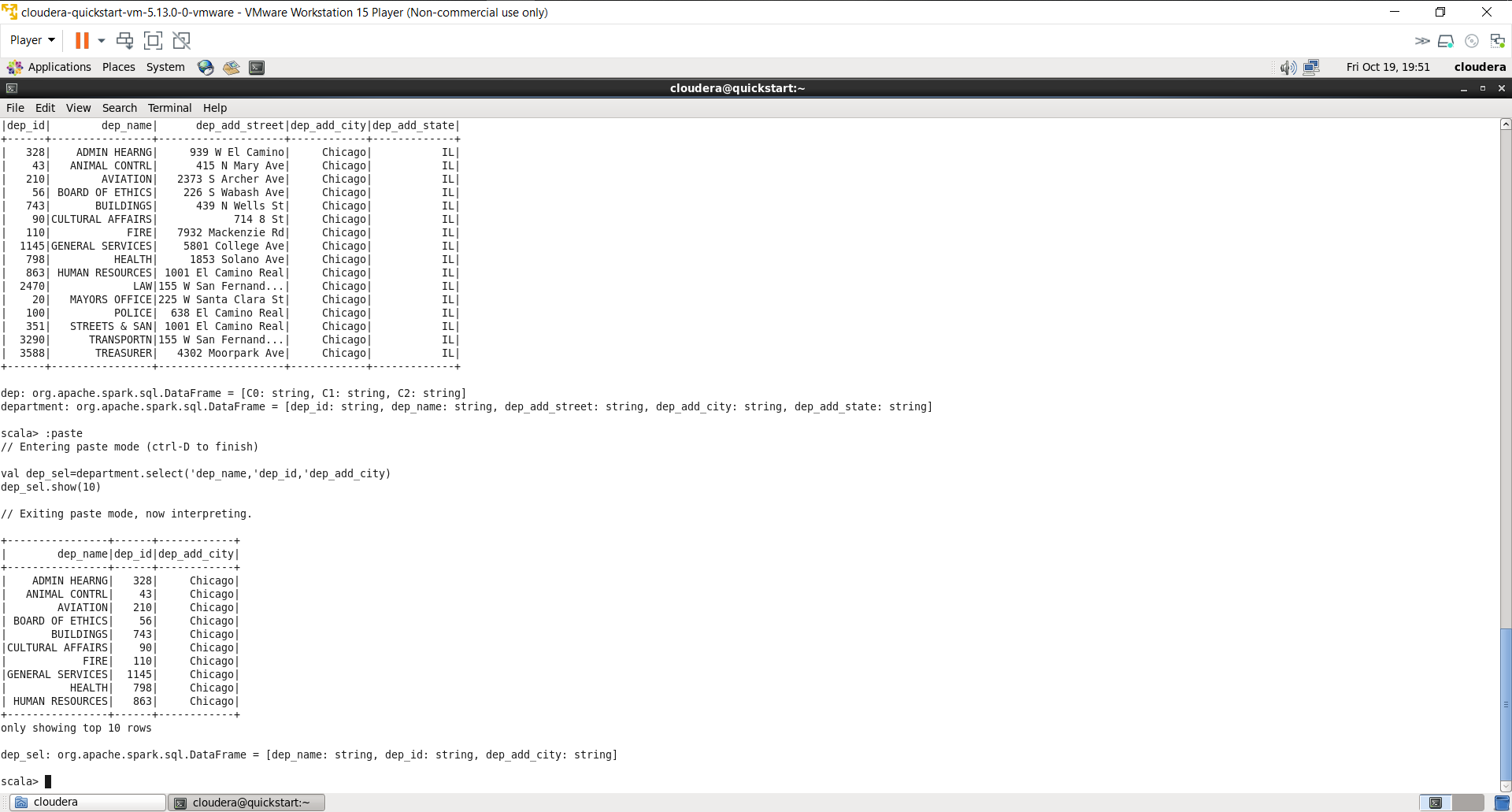
department.show()



## Select the columns from the department relation. Display 10 of these records on your screen.

val dep\_sel=department.select('dep\_name,'dep\_id,'dep\_add\_city)

dep\_sel.show(10)



## Create a relation with bonus dataset with schema (column names and datatype) and name it bonus. Once created, describe the relation.

val bonusSchema = StructType(Array(

StructField("emp\_id", IntegerType, true),

StructField("emp\_bonus", FloatType, true)))

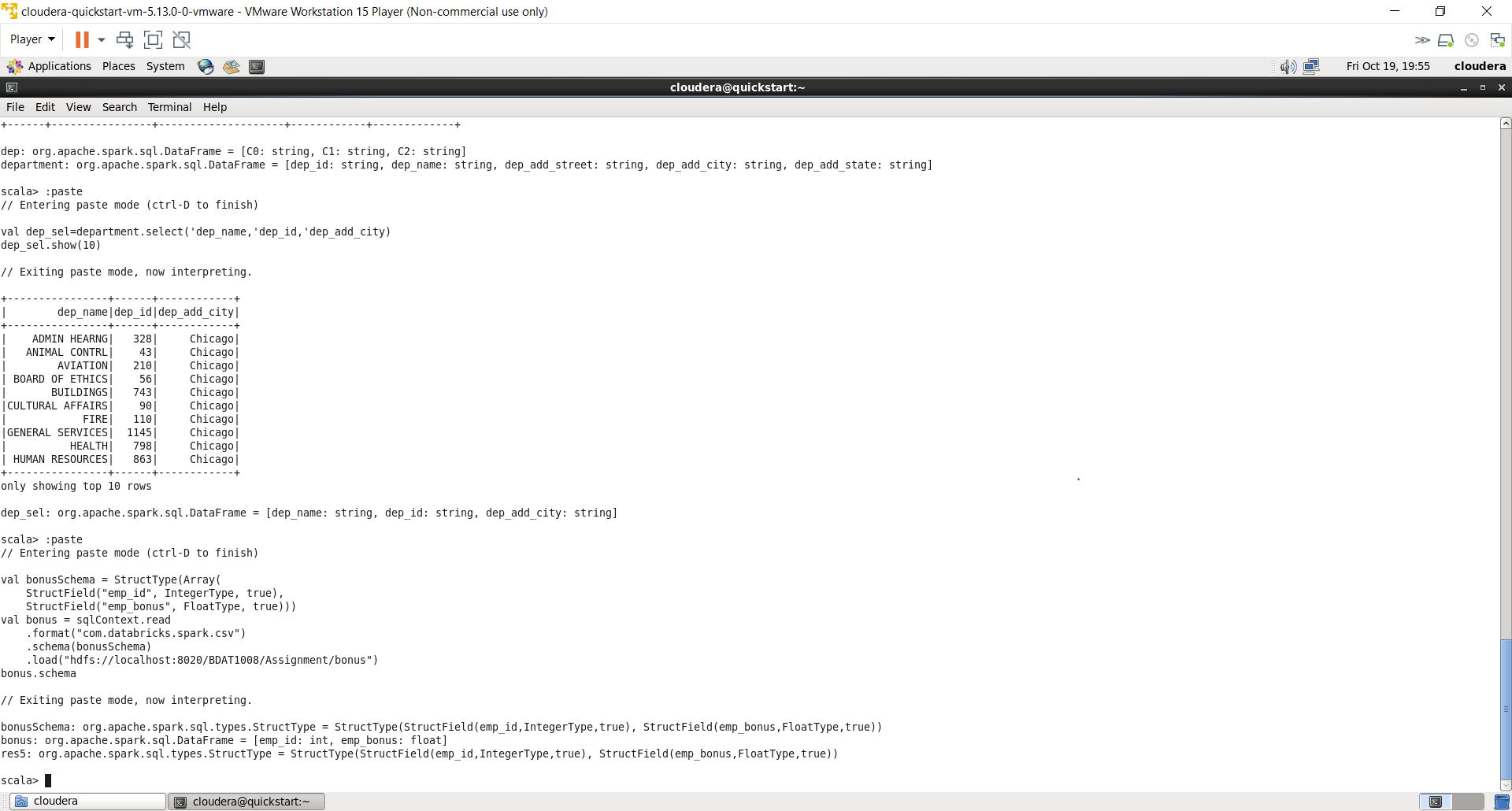
val bonus = sqlContext.read

.format("com.databricks.spark.csv")

.schema(bonusSchema)

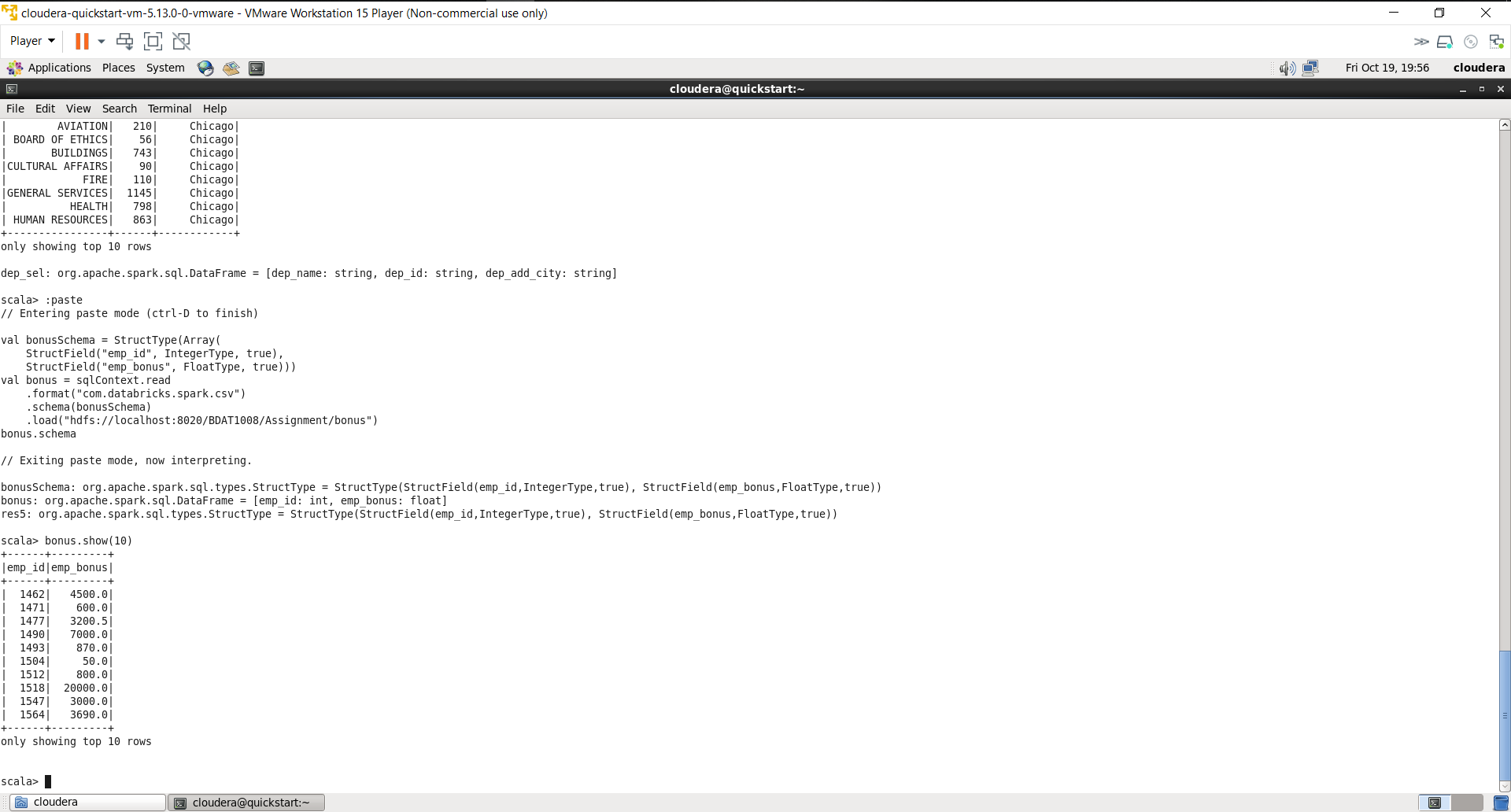
.load("hdfs://localhost:8020/BDAT1008/Assignment/bonus")

bonus.schema



## Select the columns from the bonus relation. Display 10 of these records on your screen.

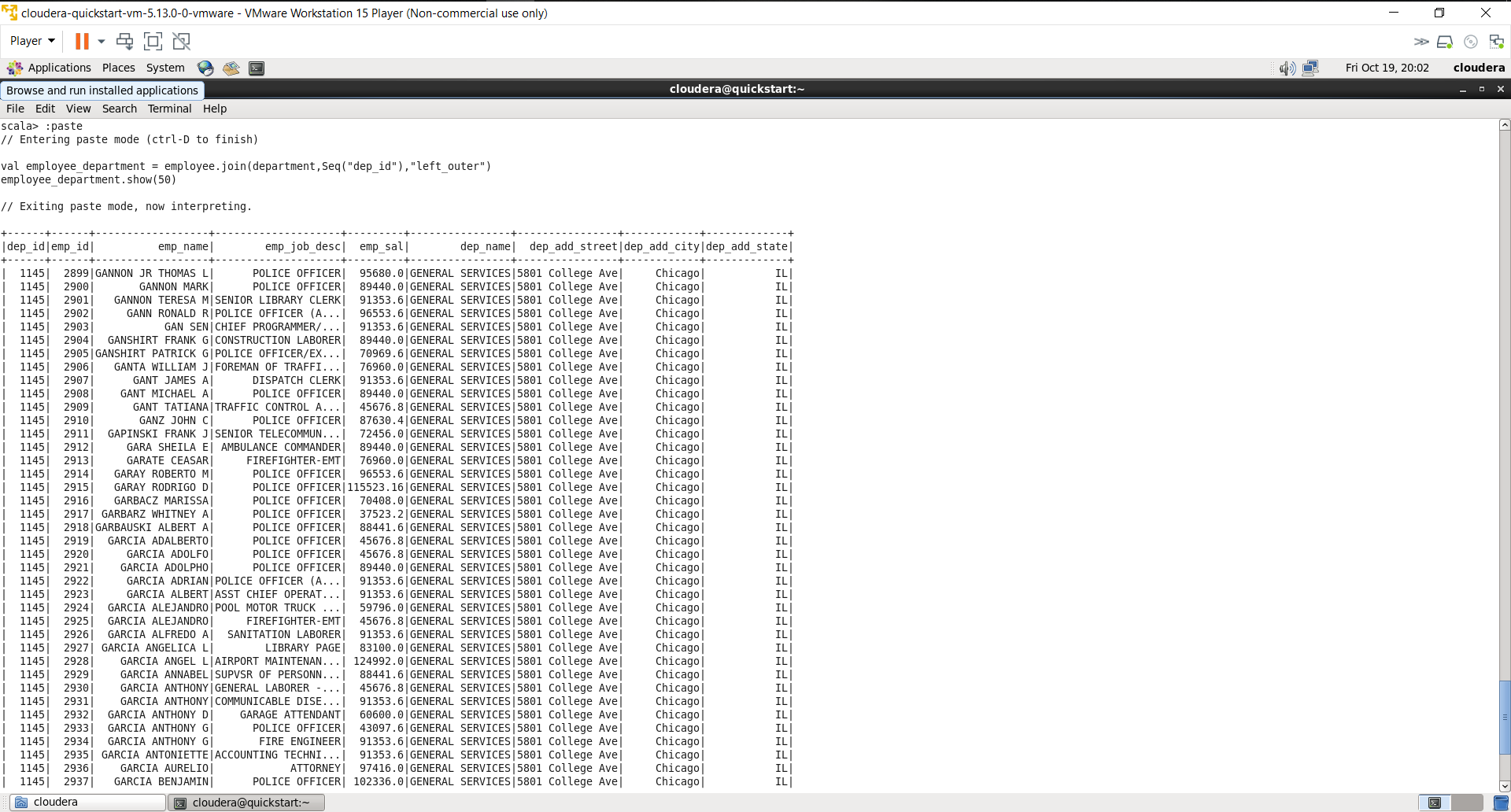
bonus.show(10)



## Join employee and department dataset to display Department Name and Department Address (as 3 separate columns - street, city and state) for each row in employee. Display 10 of these records on your screen.

val employee\_department = employee.join(department,Seq("dep\_id"),"left\_outer")

employee\_department.show(50)

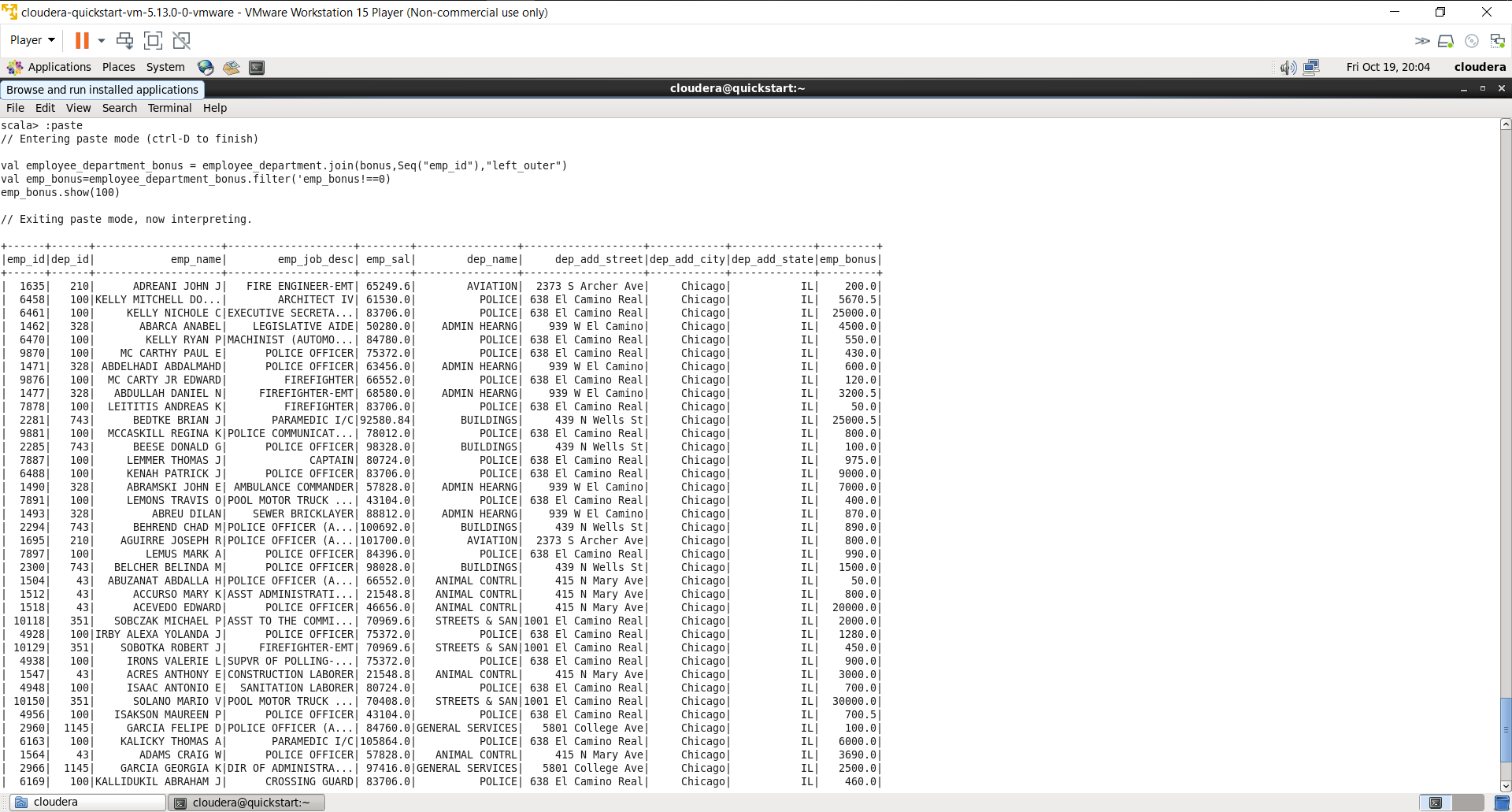


## Display all the employees who received bonus. The result should not display any employees who did not receive a bonus. Display 10 of these records on your screen.

val employee\_department\_bonus = employee\_department.join(bonus,Seq("emp\_id"),"left\_outer")

val emp\_bonus=employee\_department\_bonus.filter('emp\_bonus!==0)

emp\_bonus.show(100)

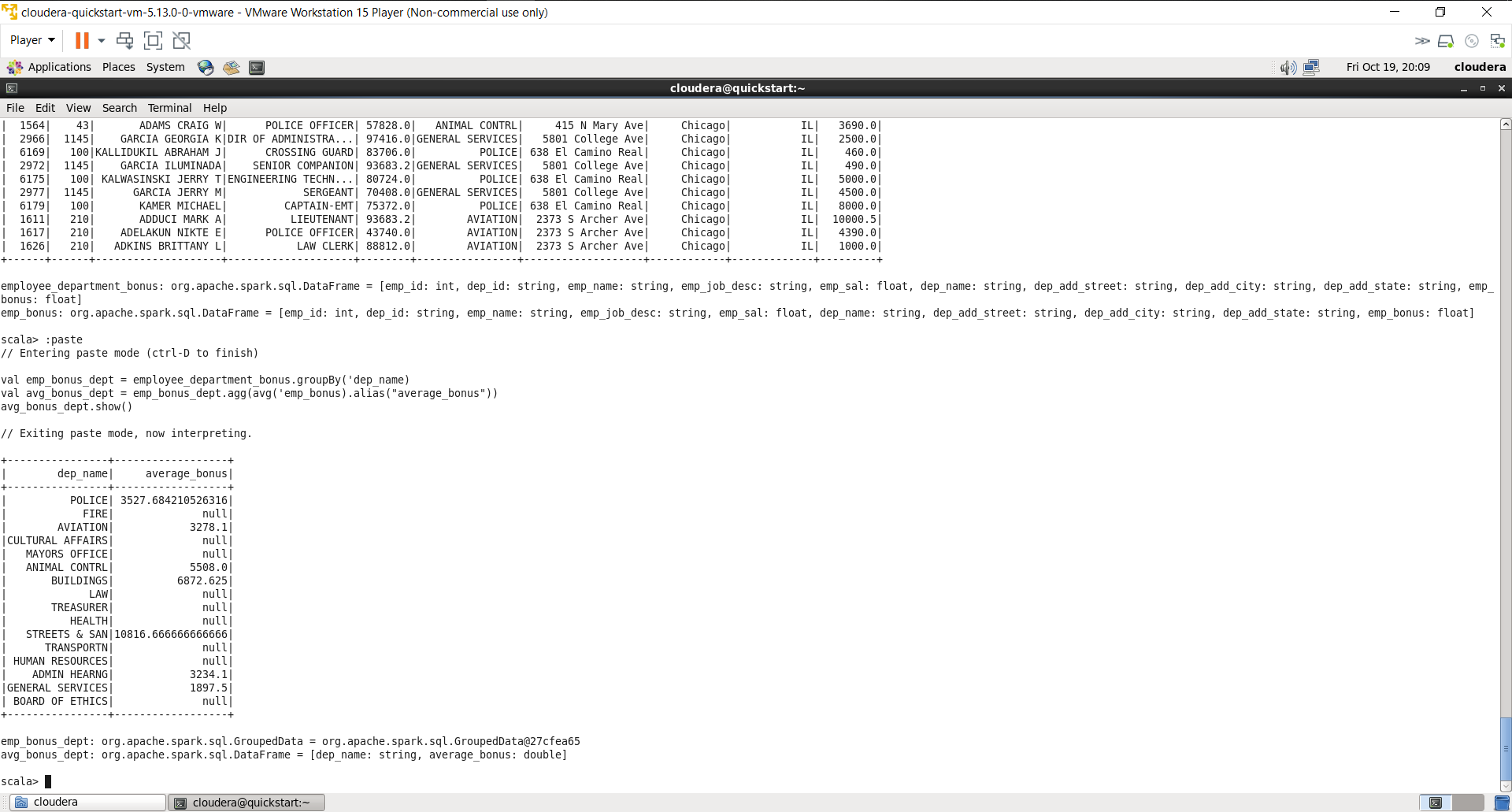


## Display the average bonus by department.

val emp\_bonus\_dept = employee\_department\_bonus.groupBy('dep\_name)

val avg\_bonus\_dept = emp\_bonus\_dept.agg(avg('emp\_bonus).alias("average\_bonus"))

avg\_bonus\_dept.show()



## Display the number of employees in each department.

val emp\_dept = emp\_bonus\_dept.count()

emp\_dept.show()

