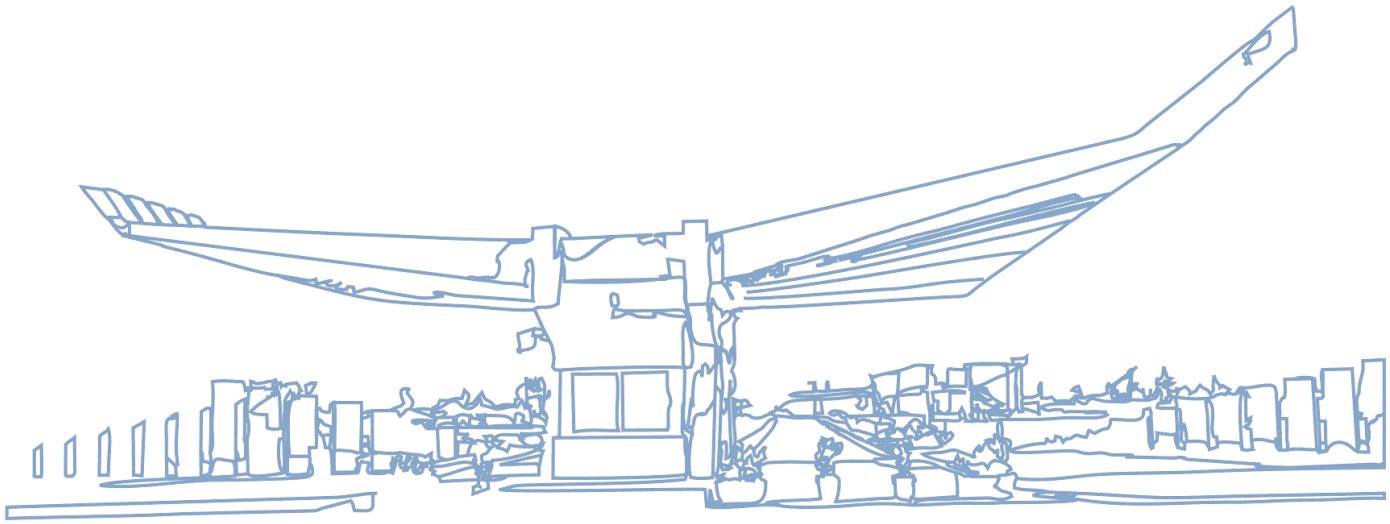
EPOKA UNIVERSITY | Computer Engineering Department

CEN 571 – Data Mining

Assignment 7: RDD



PREPARED:
Baftjar TABAKU

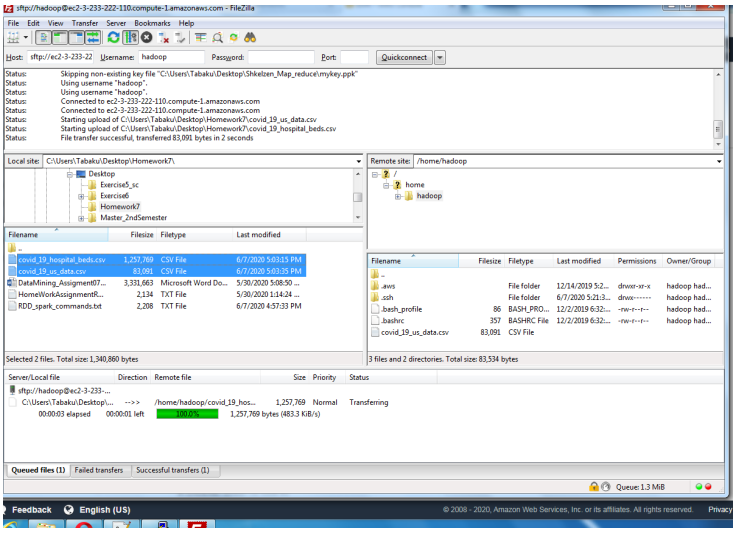
07.06.2020
Epoka University
Tirana, ALBANIA

ACCEPTED:
Prof.Dr. Arben Asllani

Tasks to complete

In this assignment, we work with 'csv' files named ‘covid_19_us_data’ and ‘covid_19_us_hospital_beds’

1. Download the files in your computer, study the structure of the file, and upload both of them in the HDFS subdirectory RDD/Ex4. Start the Spark shell.



```
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R::::R      R::::R
  E::::::EEEEEEEEEE  M:::::M M:::M M:::M M:::::M   R::::RRRRRR:::::R
  E:::::::::::::::E  M:::::M  M:::M:::M  M:::::M   R:::::::::::RR
  E::::::EEEEEEEEEE  M:::::M   M:::::M   M:::::M   R::::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R::::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R::::R      R::::R
EE::::::EEEEEEEE::::E M:::::M             M:::::M   R::::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRR      RRRRRR

[hadoop@ip-172-31-15-73 ~]$ ls -a
.  ..  .aws  .bash_profile  .bashrc  .ssh
[hadoop@ip-172-31-15-73 ~]$ hadoop fs -mkdir RDD
[hadoop@ip-172-31-15-73 ~]$ hadoop fs -mkdir RDD/Ex4
[hadoop@ip-172-31-15-73 ~]$ hadoop fs -copyFromLocal covid_19_us_data.csv RDD/Ex4
[hadoop@ip-172-31-15-73 ~]$ hadoop fs -copyFromLocal covid_19_hospital_beds.csv RDD/Ex4
[hadoop@ip-172-31-15-73 ~]$ hadoop fs -ls RDD/Ex4
Found 2 items
-rw-r--r--   1 hadoop hadoop    1257769 2020-06-07 15:24 RDD/Ex4/covid_19_hospital_beds.csv
-rw-r--r--   1 hadoop hadoop      83091 2020-06-07 15:24 RDD/Ex4/covid_19_us_data.csv
[hadoop@ip-172-31-15-73 ~]$ spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
20/06/07 15:24:41 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading l
Spark context Web UI available at http://ip-172-31-15-73.ec2.internal:4040
Spark context available as 'sc' (master = yarn, app id = application_1591543408514_0001).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.4.4
      /_/

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_252)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

2. Create an RDD named ‘casesRDD’ that captures the data from the ‘covid19_us_data.csv’ file. Display the first few records and the total number of records in this RDD.

```
version 2.4.4

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_252)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val casesRDD =sc.textFile("RDD/Ex4/covid_19_us_data.csv")
casesRDD: org.apache.spark.rdd.RDD[String] = RDD/Ex4/covid_19_us_data.csv MapPartitionsRDD[1] at textFile at <console>:24

scala> val topcaseslist = casesRDD.first
topcaseslist: String = date,state,fips,cases,deaths

scala> val firstCasesRdd = casesRDD.take(6)
firstCasesRdd: Array[String] = Array(date,state,fips,cases,deaths, 21-01-20,Washington,53,1,0, 22-01-20,Washington,53,1,0, 23-01-20,Washington,53,1,0, 24-01-20,Illinois,17,1,
24-01-20,Washington,53,1,0)

scala> firstCasesRdd.foreach(println)
date,state,fips,cases,deaths
21-01-20,Washington,53,1,0
22-01-20,Washington,53,1,0
23-01-20,Washington,53,1,0
24-01-20,Illinois,17,1,0
24-01-20,Washington,53,1,0

scala>
```

3. Create an RDD named ‘hospitalsRDD’ that captures the data from the covid19_us_hospital_beds.csv file. Display the first few records and the total number of records in this RDD.

```
scala> val hospitalsRDD= sc.textFile("RDD/Ex4/covid_19_hospital_beds.csv")
hospitalsRDD: org.apache.spark.rdd.RDD[String] = RDD/Ex4/covid_19_hospital_beds.csv MapPartitionsRDD[3] at textFile at <console>:24

scala> val hospitalsFirstCases = hospitalsRDD.take(6)
hospitalsFirstCases: Array[String] = Array(OBJECTID,HOSPITAL_NAME,HOSPITAL_TYPE,HQ_ADDRESS,HQ_ADDRESS1,HQ_CITY,HQ_STATE,HQ_ZIP_CODE,COUNTY_NAME,STATE_NAME,STATE_FIPS,CNTY_FIPS,F
IPS,NUM_LICENSED_BEDS,NUM_STAFFED_BEDS,NUM_ICU_BEDS,ADULT_ICU_BEDS,PEDI_ICU_BEDS,BED_UTILIZATION,Potential_Increase_In_Bed_Capac,AVG_VENTILATOR_USAGE,latitude,longtitude, 6001,W
estern New York Childrens Psychiatric Center,Psychiatric Hospital,1010 East And West Rd,,West Seneca,NY,14224,Erie,New York,36,29,36029,46,46,0,0,,,0,0,42.8203288,-78.7337105, 6
002,Western Regional Medical Facility - John Montford Unit,Short Term Acute Care Hospital,8602 Peach Ave,,Lubbock,TX,79404,Lubbock,Texas,48,303,48303,550,550,96,96,,,0,0,33.5184
611,-101.7793662, 6003,Wildwood Lifestyle Center,Short Term Acute Care Hospital,435 Life ...
scala> val topHospitalsList = hospitalsRDD.first
topHospitalsList: String = OBJECTID,HOSPITAL_NAME,HOSPITAL_TYPE,HQ_ADDRESS,HQ_ADDRESS1,HQ_CITY,HQ_STATE,HQ_ZIP_CODE,COUNTY_NAME,STATE_NAME,STATE_FIPS,CNTY_FIPS,FIPS,NUM_LICENSED
_BEDS,NUM_STAFFED_BEDS,NUM_ICU_BEDS,ADULT_ICU_BEDS,PEDI_ICU_BEDS,BED_UTILIZATION,Potential_Increase_In_Bed_Capac,AVG_VENTILATOR_USAGE,latitude,longtitude

scala>

scala>

scala> hospitalsFirstCases.foreach(println)
OBJECTID,HOSPITAL_NAME,HOSPITAL_TYPE,HQ_ADDRESS,HQ_ADDRESS1,HQ_CITY,HQ_STATE,HQ_ZIP_CODE,COUNTY_NAME,STATE_NAME,STATE_FIPS,CNTY_FIPS,FIPS,NUM_LICENSED_BEDS,NUM_STAFFED_BEDS,NUM_
ICU_BEDS,ADULT_ICU_BEDS,PEDI_ICU_BEDS,BED_UTILIZATION,Potential_Increase_In_Bed_Capac,AVG_VENTILATOR_USAGE,latitude,longtitude
6001,Western New York Childrens Psychiatric Center,Psychiatric Hospital,1010 East And West Rd,,West Seneca,NY,14224,Erie,New York,36,29,36029,46,46,0,0,,,0,0,42.8203288,-78.7337
105
6002,Western Regional Medical Facility - John Montford Unit,Short Term Acute Care Hospital,8602 Peach Ave,,Lubbock,TX,79404,Lubbock,Texas,48,303,48303,550,550,96,96,,,0,0,33.518
4611,-101.7793662
6003,Wildwood Lifestyle Center,Short Term Acute Care Hospital,435 Life Style Ln,,Wildwood,GA,30757,Dade,Georgia,13,83,13083,36,36,6,6,,,0,0,34.980559,-85.401676
6004,EastPointe Hospital,Psychiatric Hospital,7400 Roper Ln,,Daphne,AL,36526,Baldwin,Alabama,1,3,1003,66,66,0,0,0,0.5968451,0,0,30.6355144,-87.8953143
6005,Al Rashid Health and Wellness Center,Short Term Acute Care Hospital,2500 Main St,,Lawrenceville,NJ,8648,Mercer,New Jersey,34,21,34021,18,18,3,3,,,0,0,40.2927431,-74.7336705

scala>
```

4. Execute a query to display two select ‘state’, ‘cases’ and ‘deaths’ from the 'casesRDD' and store it into finalCasesRDD. Show the first few records and provide a screenshot of the result.

```
scala> import org.apache.spark.sql._
import org.apache.spark.sql._

scala> import org.apache.spark.sql.types._
import org.apache.spark.sql.types._

scala> import org.apache.spark.sql.Row
import org.apache.spark.sql.Row

scala> val states_cases_deaths_RDD = casesRDD.filter(line=>line!=topcaseslist).map(line=>line.split(",")).map(values=>Row(values(1),values(3).toLong,values(4).toLong))
states_cases_deaths_RDD: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[6] at map at <console>:34

scala> val state_deaths = states_cases_deaths_RDD.take(10)
state_deaths: Array[org.apache.spark.sql.Row] = Array([Washington,1,0], [Washington,1,0], [Washington,1,0], [Illinois,1,0], [Washington,1,0], [California,1,0], [Illinois,1,0],
Washington,1,0], [Arizona,1,0], [California,2,0])

scala> state_deaths.foreach(println)
[Washington,1,0]
[Washington,1,0]
[Washington,1,0]
[Illinois,1,0]
[Washington,1,0]
[California,1,0]
[Illinois,1,0]
[Washington,1,0]
[Arizona,1,0]
[California,2,0]

scala>
```

5. Execute a query to display two select ‘STATE_NAME’, ‘HOSPITAL_NAME’ and ‘HOSPITAL_TYPE’ from the 'hospitalsRDD' and store it into finalHospitalsRDD. Show the first few records and provide a screenshot of the result.

```
scala> val hospitalsRddSelect = hospitalsRDD.filter(line=>line!=topHospitalsList).map(line=>line.split(",")).map(values=>Row(values(9),values(1),values(2)))
hospitalsRddSelect: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[9] at map at <console>:34

scala> val state_deaths = hospitalsRddSelect.take(6)
state_deaths: Array[org.apache.spark.sql.Row] = Array([New York,Western New York Childrens Psychiatric Center,Psychiatric Hospital], [Texas,Western Regional Medical Facility - J
ohn Montford Unit,Short Term Acute Care Hospital], [Georgia,Wildwood Lifestyle Center,Short Term Acute Care Hospital], [Alabama,EastPointe Hospital,Psychiatric Hospital], [New J
ersey,Al Rashid Health and Wellness Center,Short Term Acute Care Hospital], [Massachusetts,Austen Riggs Psychiatric Center,Psychiatric Hospital])

scala> state_deaths.foreach(println)
[New York,Western New York Childrens Psychiatric Center,Psychiatric Hospital]
[Texas,Western Regional Medical Facility - John Montford Unit,Short Term Acute Care Hospital]
[Georgia,Wildwood Lifestyle Center,Short Term Acute Care Hospital]
[Alabama,EastPointe Hospital,Psychiatric Hospital]
[New Jersey,Al Rashid Health and Wellness Center,Short Term Acute Care Hospital]
[Massachusetts,Austen Riggs Psychiatric Center,Psychiatric Hospital]

scala>
```

6. Create a DataFrame named ‘casesDF’ and transform the data from ‘finalCasesRDD’ to this DataFrame. Display the schema and the first few records of the ‘casesDF’. Provide a screenshot of the results.

```
scala> val schema_of_cases = StructType(Array(StructField("state", StringType), StructField("cases", LongType), StructField("deaths", LongType)))
schema_of_cases: org.apache.spark.sql.types.StructType = StructType(StructField(state,StringType,true), StructField(cases,LongType,true), StructField(deaths,LongType,true))

scala>

scala> val casesDF = spark.createDataFrame(states_cases_deaths_RDD,schema_of_cases)
casesDF: org.apache.spark.sql.DataFrame = [state: string, cases: bigint ... 1 more field]

scala> casesDF.show(5)
+----------+-----+------+
|     state|cases|deaths|
+----------+-----+------+
|Washington|    1|     0|
|Washington|    1|     0|
|Washington|    1|     0|
|  Illinois|    1|     0|
|Washington|    1|     0|
+----------+-----+------+
only showing top 5 rows


scala>
```

7. Create a DataFrame named ‘hospitalsDF’ and transform the data from ‘finalHospitalsRDD’ to this DataFrame. Display the schema and the first few records of the ‘hospitalsDF’. Provide a screenshot of the results.

```
scala> val schema_of_hospitals = StructType(Array(StructField("state", StringType), StructField("hospital", StringType), StructField("hospital_type", StringType)))
schema_of_hospitals: org.apache.spark.sql.types.StructType = StructType(StructField(state,StringType,true), StructField(hospital,StringType,true), StructField(hospital_type,St
ngType,true))

scala> val hospitalDF = spark.createDataFrame(hospitalsRddSelect,schema_of_hospitals)
hospitalDF: org.apache.spark.sql.DataFrame = [state: string, hospital: string ... 1 more field]

scala> hospitalDF.show(5)
+----------+--------------------+--------------------+
|     state|            hospital|       hospital_type|
+----------+--------------------+--------------------+
|  New York|Western New York ...|Psychiatric Hospital|
|     Texas|Western Regional ...|Short Term Acute ...|
|   Georgia|Wildwood Lifestyl...|Short Term Acute ...|
|   Alabama| EastPointe Hospital|Psychiatric Hospital|
|New Jersey|Al Rashid Health ...|Short Term Acute ...|
+----------+--------------------+--------------------+
only showing top 5 rows


scala>
```

8. Execute another query that results in the 'casesDF' with just ‘state’ and ‘cases’ and only includes states with more than 1000 cases. Provide a screenshot of the result.

```
scala> val custom_casesDF = casesDF.select("state","cases").groupBy("state").agg(sum("cases").as("totcases")).where("totcases>1000")
custom_casesDF: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [state: string, totcases: bigint]

scala> custom_casesDF.show()
+--------------------+--------+
|               state|totcases|
+--------------------+--------+
|                Utah|   54643|
|              Hawaii|   11665|
|           Minnesota|   38793|
|                Ohio|  169432|
|              Oregon|   36537|
|            Arkansas|   33537|
|               Texas|  308551|
|        North Dakota|    8453|
|        Pennsylvania|  500116|
|         Connecticut|  278162|
|            Nebraska|   20754|
|             Vermont|   16217|
|              Nevada|   67260|
|         Puerto Rico|   18025|
|          Washington|  262490|
|            Illinois|  476045|
|            Oklahoma|   44593|
|      Virgin Islands|    1269|
|District of Columbia|   43129|
|            Delaware|   36881|
+--------------------+--------+
only showing top 20 rows


scala>
```

9. Create a new DataFrame that joins the two original DataFrames: 'casesDF' and 'hospitalsDF' by the state. Display the first 10 records. Provide a screenshot of the result.

```
scala> val joinedDataFrame = custom_casesDF.join(hospitalDF,custom_casesDF("state")===hospitalDF("state"))
joinedDataFrame: org.apache.spark.sql.DataFrame = [state: string, totcases: bigint ... 3 more fields]

scala> joinedDataFrame.show(10)
+-----+--------+-----+--------------------+--------------------+
|state|totcases|state|            hospital|       hospital_type|
+-----+--------+-----+--------------------+--------------------+
| Utah|   54643| Utah|   Center for Change|Psychiatric Hospital|
| Utah|   54643| Utah|Copper Hills Yout...|Psychiatric Hospital|
| Utah|   54643| Utah|Northern Utah Reh...|Rehabilitation Ho...|
| Utah|   54643| Utah|     Layton Hospital|Short Term Acute ...|
| Utah|   54643| Utah|    ViewPoint Center|Short Term Acute ...|
| Utah|   54643| Utah|Kane County Hospital|Critical Access H...|
| Utah|   54643| Utah|Utah Valley Speci...|Long Term Acute C...|
| Utah|   54643| Utah| Utah State Hospital|Psychiatric Hospital|
| Utah|   54643| Utah|South Davis Commu...|Long Term Acute C...|
| Utah|   54643| Utah|Marian Center at ...|Psychiatric Hospital|
+-----+--------+-----+--------------------+--------------------+
only showing top 10 rows


scala>
```

10. Transform the last DataFrame into an RDD. Display the first 10 elements of the RDD.

```
scala> val transformedRdd = joinedDataFrame.rdd
transformedRdd: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[53] at rdd at <console>:32

scala> transformedRdd.take(5).foreach(println)
[Utah,54643,Utah,Center for Change,Psychiatric Hospital]
[Utah,54643,Utah,Copper Hills Youth Center,Psychiatric Hospital]
[Utah,54643,Utah,Northern Utah Rehabilitation Hospital,Rehabilitation Hospital]
[Utah,54643,Utah,Layton Hospital,Short Term Acute Care Hospital]
[Utah,54643,Utah,ViewPoint Center,Short Term Acute Care Hospital]

scala>
```

11. Save the data from the RDD in the above step onto the cluster. Open another terminal and verify that results are stored in the cluster. Provide a screenshot of the result.

```
scala> [hadoop@ip-172-31-15-73 ~]$ hadoop fs -ls RDD
Found 2 items
drwxr-xr-x   - hadoop hadoop          0 2020-06-07 15:24 RDD/Ex4
drwxr-xr-x   - hadoop hadoop          0 2020-06-07 15:31 RDD/output.txt
[hadoop@ip-172-31-15-73 ~]$ hadoop fs -copyToLocal RDD/output.txt
[hadoop@ip-172-31-15-73 ~]$ ls output.txt
part-00000  part-00015  part-00030  part-00045  part-00060  part-00075  part-00090  part-00105  part-00120  part-00135  part-00150  part-00165  part-00180  part-00195
part-00001  part-00016  part-00031  part-00046  part-00061  part-00076  part-00091  part-00106  part-00121  part-00136  part-00151  part-00166  part-00181  part-00196
part-00002  part-00017  part-00032  part-00047  part-00062  part-00077  part-00092  part-00107  part-00122  part-00137  part-00152  part-00167  part-00182  part-00197
part-00003  part-00018  part-00033  part-00048  part-00063  part-00078  part-00093  part-00108  part-00123  part-00138  part-00153  part-00168  part-00183  part-00198
part-00004  part-00019  part-00034  part-00049  part-00064  part-00079  part-00094  part-00109  part-00124  part-00139  part-00154  part-00169  part-00184  part-00199
part-00005  part-00020  part-00035  part-00050  part-00065  part-00080  part-00095  part-00110  part-00125  part-00140  part-00155  part-00170  part-00185  _SUCCESS
part-00006  part-00021  part-00036  part-00051  part-00066  part-00081  part-00096  part-00111  part-00126  part-00141  part-00156  part-00171  part-00186
part-00007  part-00022  part-00037  part-00052  part-00067  part-00082  part-00097  part-00112  part-00127  part-00142  part-00157  part-00172  part-00187
part-00008  part-00023  part-00038  part-00053  part-00068  part-00083  part-00098  part-00113  part-00128  part-00143  part-00158  part-00173  part-00188
part-00009  part-00024  part-00039  part-00054  part-00069  part-00084  part-00099  part-00114  part-00129  part-00144  part-00159  part-00174  part-00189
part-00010  part-00025  part-00040  part-00055  part-00070  part-00085  part-00100  part-00115  part-00130  part-00145  part-00160  part-00175  part-00190
part-00011  part-00026  part-00041  part-00056  part-00071  part-00086  part-00101  part-00116  part-00131  part-00146  part-00161  part-00176  part-00191
part-00012  part-00027  part-00042  part-00057  part-00072  part-00087  part-00102  part-00117  part-00132  part-00147  part-00162  part-00177  part-00192
part-00013  part-00028  part-00043  part-00058  part-00073  part-00088  part-00103  part-00118  part-00133  part-00148  part-00163  part-00178  part-00193
part-00014  part-00029  part-00044  part-00059  part-00074  part-00089  part-00104  part-00119  part-00134  part-00149  part-00164  part-00179  part-00194
[hadoop@ip-172-31-15-73 ~]$
[hadoop@ip-172-31-15-73 ~]$
```