CASE STUDY 4

Objective1: Load file into Spark

Sol:

```
Step1 -> Create RDD and load file into it
```

//open spark-shell and create rdd

→ val baseRDD = sc.textFile("/DatasetCS4/inpatientCharges.csv")

//remove header

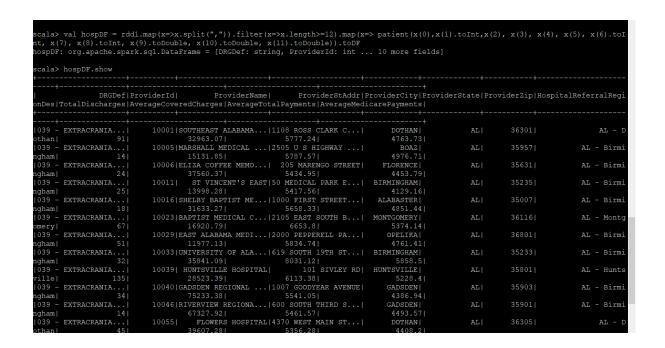
- → val header = baseRDD.first()
- → val rdd1 = baseRDD.filter(row => row != header)

//define case class for the schema

case class patient(DRGDef:String, ProviderId:String, ProviderName:String, ProviderStAddr:String, ProviderCity:String, ProviderState:String, ProviderZip:String, HospitalReferralRegionDes:String, TotalDischarges:String, AverageCoveredCharges:String, AverageTotalPayments:String, AverageMedicarePayments:String)

//convert to dataframe

- → val hospDF = rdd1.map(x=>x.split(",")).filter(x=>x.length>=12).map(x=> patient(x(0),x(1).toInt,x(2), x(3), x(4), x(5), x(6).toInt, x(7), x(8).toInt, x(9).toDouble, x(10).toDouble, x(11).toDouble)).toDF
- → hospDF.show



Objective2:

2.1 What is the average amount of AverageCoveredCharges per state

Sol: Step1: Register the dataframe as temp table

→ hospDF.registerTempTable("hospDF")

Step2: write sql query to calculate average

→ spark.sql("select ProviderState, avg(AverageCoveredCharges) as AvgAmt from hospDF group by ProviderState").show

```
acadgild@localhost:~
 cala> hospDF.registerTempTable("hospDF")
warning: there was one deprecation warning; re-run with -deprecation for details
scala> spark.sql("select ProviderState, avg(AverageCoveredCharges) as AvgAmt from hospDF group by ProviderState").show
18/05/27 14:30:30 WARN executor.Executor: Managed memory leak detected; size = 17039360 bytes, TID = 34
        ProviderState
                                   AvgAmt|
              TOWANDA
                                     27.2|
null|
            SAN PABLO
           CUMBERLAND| 54.57142857142857
              HANCOCK
            WATERTOWN | 30.571428571428573 |
             EDMONDS | 23.571428571428573 |
          MCMINNVILLE|
                                     38.0
                                      null
    750 MORPHY AVENUE
                                      null
2500 ROCKY MOUNTA...|
20 YORK ST
     1000 MAR-WALT DR
```

2.2 find out the AverageTotalPayments charges per state

Sol: Step1: write query to calculate the sum

→ spark.sql("select ProviderState, sum(AverageTotalPayments) as TotalCharges from hospDF group by ProviderState").show

```
acadgild@localhost:~
 scala> spark.sql("select ProviderState, sum(AverageTotalPayments) as TotalCharges from hospDF group by ProviderState").show
 L8/05/27 14:57:34 WARN executor.Executor: Managed memory leak detected; size = 17039360 bytes, TID = 40
                             TotalCharges|
        ProviderState|
              TOWANDA |
            SAN PABLO|310797.86999999994|
         PO BOX 1727"|
                                   254.0|
           CUMBERLAND |
                                 80783.02|
                                 22929.57|
              HANCOCK |
            PRINCETON | 331020.16000000003|
            WATERTOWN |
              EDMONDS |
                                260504.04|
          MCMINNVILLE |
                                  34349.8|
               BAXLEY
                                 10866.35|
    750 MORPHY AVENUE
 2500 ROCKY MOUNTA...
                                     44.0|
           20 YORK ST|
     1000 MAR-WALT DR
     14000 FIVAY ROAD|
                                     74.01
 only showing top 20 rows
```

2.3 Find out the AverageMedicarePayments charges per state

Sol: Step1: write query to calculate medicareCharges

→ spark.sql("select ProviderState, sum(AverageMedicarePayments) as MedicareCharges from hospDF group by ProviderState").show

Objective3:

3.1 Find out the total number of Discharges per state and for each disease

Sol: command used:

→ spark.sql("select DRGDef, ProviderState, sum(TotalDischarges) as SumDischarges from hospDF group by ProviderState, DRGDef").show

```
acadgild@localhost:~
 scala> spark.sql("select DRGDef, ProviderState, sum(TotalDischarges) as SumDischarges from hospDF group by ProviderState, DRGDef").shc
 18/05/27 15:24:12 WARN executor.Executor: Managed memory leak detected; size = 17039360 bytes, TID = 255
                DRGDef | ProviderState | SumDischarges |
 057 - DEGENERATIV...
                              SAGINAW
                                                 null|
                              CONYERS
                            GREENBRAE
                                                4305.0
                                   NY
 101 - SEIZURES W/...
                         JOHNSON CITY
 149 - DYSEQUILIBRIUM
176 - PULMONARY E...
                                                 700.0
                           BINGHAMTON
 190 - CHRONIC OBS...|
192 - CHRONIC OBS...|
                             ARDMORE
                                                 null
 193 - SIMPLE PNEU...|SAN FRANCISCO|
 193 - SIMPLE PNEU...
                           FALL RIVER
 208 - RESPIRATORY...
```

3.2 Sort the output in descending order of totalDischarges

Sol: Command used:

→ spark.sql("select DRGDef, ProviderState, sum(TotalDischarges) as SumDischarges from hospDF group by ProviderState, DRGDef order by SumDischarges desc").show

