Data sets

# change log level to error

scala> sc.setLogLevel("ERROR")

# laod into DF

scala> val df = spark.read.format("json").load("/sparkLabData/store\_locations.json")

[Stage 0:> (0 + df: org.apache.spark.sql.DataFrame = [city: string, state: string ... 1 more field]

# see the data

scala> df.show(5)

+-----------+-----+---------+

| city|state| zip\_code|

+-----------+-----+---------+

| Antioch| CA|945097911|

| Woodland| CA|957765409|

| San Jose| CA|951311866|

|Victorville| CA|923954216|

| Chico| CA|959284422|

+-----------+-----+---------+

only showing top 5 rows

# define the case class

scala> case class Store(city: String, state: String, zip\_code: Long)

defined class Store

# convert into ds

scala> val ds = df.as[Store]

ds: org.apache.spark.sql.Dataset[Store] = [city: string, state: string ... 1 more field]

# see the content of ds. You can run all transformations and actions learnt in dataframe videos…

scala> ds.show(5)

+-----------+-----+---------+

| city|state| zip\_code|

+-----------+-----+---------+

| Antioch| CA|945097911|

| Woodland| CA|957765409|

| San Jose| CA|951311866|

|Victorville| CA|923954216|

| Chico| CA|959284422|

+-----------+-----+---------+

only showing top 5 rows

# another way to create dataset

scala> case class State(state: String, stateName: String)

defined class State

scala> val ds1 = Seq(State("CA","California")).toDS()

ds1: org.apache.spark.sql.Dataset[State] = [state: string, stateName: string]

scala> ds1.show()

+-----+----------+

|state| stateName|

+-----+----------+

| CA|California|

+-----+----------+

# transformation on data set returns data frame

scala> ds.groupBy(col("city")).count()

res15: org.apache.spark.sql.**DataFrame** = [city: string, count: bigint]

# Alternate method that returns dataset

scala> ds.**groupByKey**(x => x.city).count()

res16: org.apache.spark.sql.**Dataset**[(String, Long)] = [value: string, count(1): bigint]