Sample datasets : https://github.com/jdorfman/awesome-json-datasets

Bu bölümde Dataset hakkında genel bilgiler vereceğiz

Dataset ile farklı kaynaklardan verileri toplayarak SQL benzeri script ve metodlarla analizler yapabiliriz .

Genel olarak RDD metodlarına göre kullanımı daha basittir bu yüzden Spark ile

SparkSession

DataSet ile analizler yapmak için öncelikle SparkSession oluşturmamız gerekir.

```
val spark = SparkSession.builder.master("local").
 appName("SparkByExample")
 .getOrCreate()
```

Dataset

Öncelikle alttaki veri kümesini bilgisayarımız indirelim

https://gist.github.com/raine/da15845f332a2fb8937b344504abfbe0

show

Bu metod ile veri kaynağının şema yapısını ve örnek verileri gösterebiliriz

```
import org.apache.spark.sql.SparkSession


object DataSetShow {
 def main(args: Array[String]): Unit = {

   val spark = SparkSession.builder.master("local").
     appName("SparkByExample")
     .getOrCreate()


   val json =
spark.read.json("/Users/serkan/Desktop/Training/ApacheSpark/peoplejson/people.json"
);

   json.show()
 }
}
```

printSchema

Bu metod ile veri kaynağının şema yapısını gösterebiliriz

```
import org.apache.spark.sql.SparkSession


object DataSetPrintSchema {
 def main(args: Array[String]): Unit = {

   val spark = SparkSession.builder.master("local").
     appName("SparkByExample")
     .getOrCreate()


   val json =
spark.read.json("/Users/serkan/Desktop/Training/ApacheSpark/peoplejson/people.json"
);

   json.printSchema()
 }
}
```

Select metodu

Bu metod ile belirli kolonların gösterilmesini sağlayabiliriz

```
import org.apache.spark.sql.SparkSession


object DataSetPrintSchema {
 def main(args: Array[String]): Unit = {

   val spark = SparkSession.builder.master("local").
     appName("SparkByExample")
     .getOrCreate()


   val ds =
spark.read.json("/Users/serkan/Desktop/Training/ApacheSpark/peoplejson/people.json"
);

   ds.select("name","city").show()
 }
}
```

Kolonlar uzerinde islem yapilabilir

```
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.col


object DataSetColumnProcess {
 def main(args: Array[String]): Unit = {

   val spark = SparkSession.builder.master("local").
     appName("SparkByExample")
     .getOrCreate()


   val df =
spark.read.json("/Users/serkan/Desktop/Training/ApacheSpark/peop
lejson/people.json");

   val df1 = df.select(col("name").as("name"),
     col("city").substr(0,3).as("city"))
   df1.show()
 }
}
```

Filter metodu

Bu metod ile filtreleme işlemleri yapabiliriz

```
import org.apache.spark.sql.functions.col


object DataSetFilter {
 def main(args: Array[String]): Unit = {

   val spark = SparkSession.builder.master("local").
     appName("SparkByExample")
     .getOrCreate()


   val df =
spark.read.json("/Users/serkan/Desktop/Training/ApacheSpark/peoplejson/people.json"
);

   val dfFiltered = df.filter(col("city").like("South%"))
   dfFiltered.show()
 }
}
```

groupBy metodu

Bu metod ile belirli alanlara göre gruplama yapabiliriz.Bu gruplama üzerinden işlemler(aggregation)yapabiliriz.Mesela aynı şehirlerde yaşayan kişilerin ortalama yaşları bulunabilir

```
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.col


object DataSetGroupBy {
 def main(args: Array[String]): Unit = {

   val spark = SparkSession.builder.master("local").
     appName("SparkByExample")
     .getOrCreate()


   val df =
spark.read.json("/Users/serkan/Desktop/Training/ApacheSpark/peop
lejson/people.json");
   df.printSchema()
   df.show()
   val dfFiltered =
df.groupBy("city").count().orderBy("city").show()
   //val dfFiltered = df.groupBy(col("city")).avg("age").show()
 }
}
```