MANGO SOLUTIONS

The dplyr Package for Spark

Aimee Gott
agott@mango-solutions.com

Overview

- The Data
- Connect & Read
- The Core dplyr Functions
- SQL Functions, Joins and Sampling Data
- Save and Disconnect

Airlines Data

- Arrival and departure details for all commercial flights in US between October 1987 and April 2008.
- 120,000,000 records. **12 GB**
- stat-computing.org/dataexpo/2009/

MANGO SOLUTIONS

Connect & Read

Connecting to Spark

```
sc <- spark_connect(
            master = "…",
            app_name = "my_name"
        )
```

Importing the Data

- Multiple options for connecting to data:
  - Copy data from R data frame
  - Import from file (CSV, JSON, PARQUET)
  - Using Spark SQL
  - From a Hive table

Importing the Data - Today

- We are going to read from a parquet file
  - Column store data format
- Can be read with

```
spark_read_parquet()
```

Read a Parquet File

```
airlines <-
  spark_read_parquet(
          sc,
          name = "name_in_spark",
          path = "path/to/file"
    )
```

Importing the Data - Today

- We are also going to read from csv
- Can be read with

```
spark_read_csv()
```

The Data Object in R

- We don't work with data in R
- We work with a connection to the data
- dplyr will create Spark SQL to run
- The data will come to R (if we ask for it) after running the dplyr code

Exercise

- Ensure you have a connection to Spark
- Read the data and create a connection object to the table
- Read in the airports, carriers and plane-data CSV files
- What columns are in the airports data?
- Print the object in the console

  (DO NOT `View` THE DATA YET)

MANGO SOLUTIONS

Core dplyr Functions

The Core dplyr Functions

Function	Usage
`filter`	Filter the rows of a data set
`select`	Select columns from a data set
`mutate`	Add or manipulate columns in a data set
`arrange`	Sort the data
`summarise`	Generate summaries for columns in the data

A Quick Example

```
friday <- filter(flights, DayOfWeek == 5)

friday <- select(friday, -DayOfWeek)

friday <- mutate(friday,
   Date = paste(Year, Month, DayofMonth,
          sep = "-"))

select(friday, Year,
       Month, DayofMonth, Date)
```

The Pipe Operator

- We can use dplyr as usual when working with spark
- This includes using the pipe operator (%>%) to simplify operations

Example with Pipes

```
flights %>%
  filter(DayOfWeek == 5) %>%
  select(-DayOfWeek) %>%
  mutate(Date = paste(Year,
  Month, DayofMonth, sep = "-")) 
      %>%
  select(Year, Month,
         DayofMonth, Date)
```

Exercise

- Create a query for the flights data where:
  - depdelay > 15, depdelay < 240, dayofmonth == 15
  - Columns: year, month, arrdelay, depdelay, distance, uniquecarrier
- Ensure your query is using the pipe operator

Lazy Execution

- The SQL query will be run at the last possible moment
- If you simply print results only 10 rows will be retrieved

How Do You Get All The Data?

```
flights %>%
  filter(year > 2007) %>%
  filter(depdelay == 240) %>%
  collect()
```

What is the Code Doing?

- The dplyr functions create a Spark SQL statement
- We can see the SQL statement by using

```
show_query()
```

Exercise

- Using the code you wrote in the last exercise
  - What does the Spark SQL query look like?
  - What are the dimensions of the data?

MANGO SOLUTIONS

SQL Functions, Joins & Sampling

Translating to SQL

Operators	`+, -, *, /, %%, ^`
Mathematical functions	`abs, acos, cosh, sin, asinh, atan, atan2, atanh, ceiling, cos, cosh, cot, coth, exp, floor, log, log10, round, sign, sin, sinh, sqrt, tan, tanh`
Comparisons	`<, <=, !=, >=, >, ==, %in%`
Booleans	`&, &&, |, ||, !`
Aggregations	`mean, sum, min, max, sd, var, cor, cov, n`
Characters	`paste, tolower, toupper, nchar`
Casting	`as.double, as.integer, as.logical, as.character, as.date`

Joining and Sampling

Function	Usage
`group_by`	Apply a grouping to the data
`left_join`, `right_join`, `...`	Perform data joins (left, right, outer, ...)
`sample_n`	Sample n rows from the data

Joining

```
friday <- flights %>%
  filter(DayOfWeek == 5) %>%
  left_join(airports,
    by = c(Dest = "iata")) %>%
  filter(Origin %in%
           c("SFO", "BOS"))
```

Sampling

```
friday %>%
   sample_n(5)
```

Creating New Spark Data Frames

- When you join or sample you create a new data frame
- You may want to save this to use later
- We can do this with

```
sdf_register(r_obj, "spark_name")
```

Exercise

- Join the airport data to the flights data
- Filter the data to retain only flights originating in San Francisco and Boston
- Only retain data for Fridays
- Create a new Spark data frame containing this data

MANGO SOLUTIONS

Save & Disconnect

Saving Your Results

- When we quit Spark our data won't be saved
- If we want to keep results we need to save them
- All of our import functions have equivalent write functions

```
spark_write_parquet()
spark_write_csv()
```

Disconnecting (Don't Do This Now)

- Once we are done we need to disconnect from the spark instance

```
spark_disconnect(sc)
```