Spark SQL

Spark SQL
Game Plan

- Spark SQL background/introduction

- Discussion of Spark DataFrames

- Working with Spark DataFrames

Spark SQL
Goals

- Understand the benefits of Spark DataFrames over traditional RDDs

- Know how to instantiate and interact with a Spark DataFrame

- Know how to spin up a spark cluster on AWS

Why Spark SQL?

- It provides a DataFrame abstraction that simplifies working with structured datasets

- It can read and write data in a variety of structured formats

- It lets you query the data using SQL.

Why Spark SQL?

- Spark default RDDs —> `(Key, Value)`
- What if our data is not `(Key, Value)`, and looks like this?

```
{‘name’: ‘Amy’, age: 18,  hobby: ‘drinking’}

{‘name’: ‘Greg’, age: 60,  hobby: ‘fishing’}

{‘name’: ‘Susan’, age: 30}
```

Why Spark SQL?

To get this: **Older than 18, With hobbies**

With traditional RDDs, we have to write this:

```
rdd.filter(lambda d: d['age'] > 18) \
.filter(lambda d: 'hobby' in d.keys()) \
.map(lambda d: d['name'])
```

Why Spark SQL?
Spark DataFrames

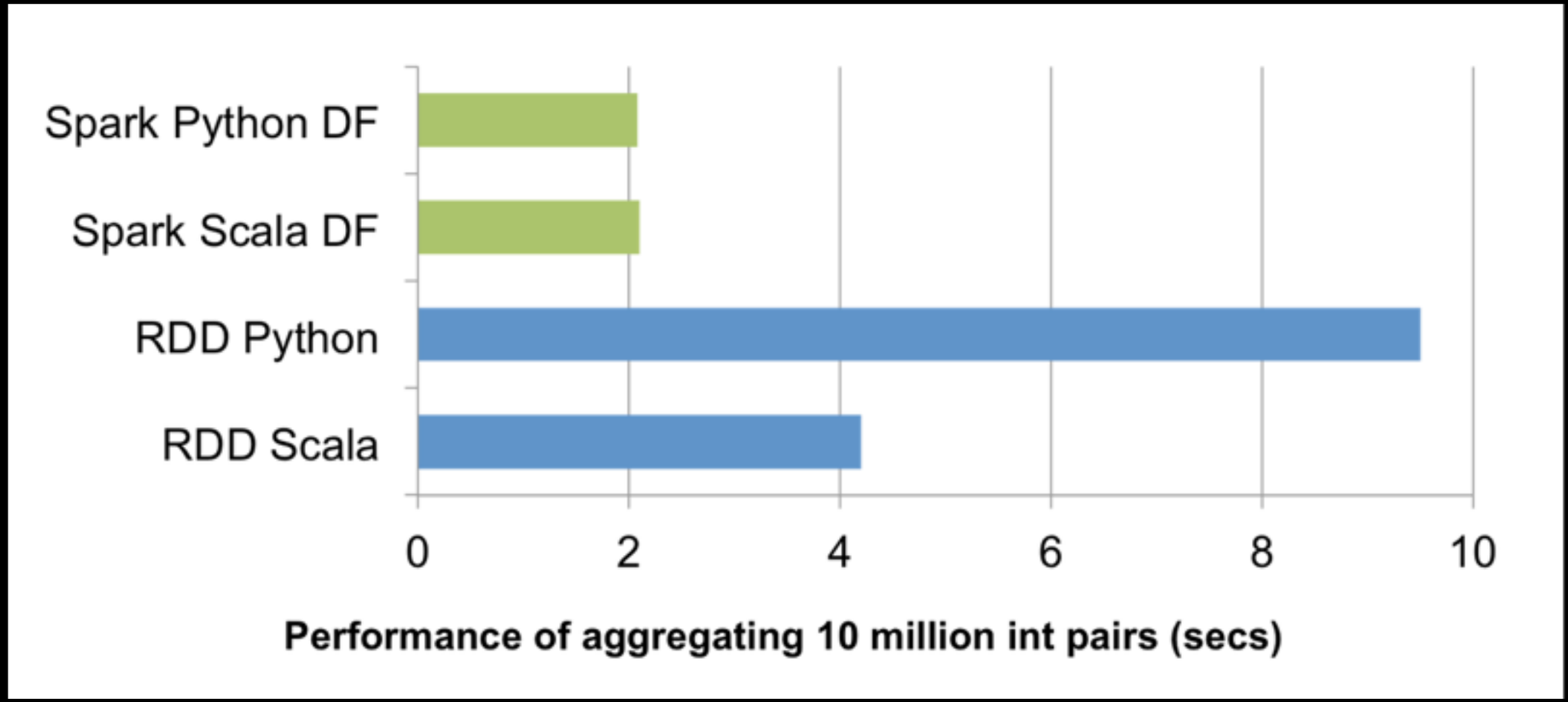
- With DataFrames, we can write this:

```
hive_context.sql(“SELECT name
                  FROM table
                  WHERE age > 18
                  AND hobby IS NOT NULL”)
```

- This is much simpler, even for just a simple query!

Why Spark SQL?
Spark DataFrames

- On top of the ease with which we can perform operations, DataFrames are also much faster!



Why Spark SQL?
Spark DataFrames

- A DataFrame contains an RDD of **Row** objects, each representing a record. A DataFrame is not technically an RDD, but we can effectively treat it as such.

- A DataFrame knows the schema of its rows, which means that it can store and process data in a more efficient manner

Spark DataFrames

- How do I get one of these things?

1. sc = SparkContext()
2. hive_context = HiveContext(sc)

OR

sql_context = SQLContext(sc)

- HiveContext() offers more functionality, and this should be your go to

Spark DataFrames

- How do I get data in one of these things and interact with it?

1. data = hive_context.jsonFile(input_file)

2. data.registerTempTable(“users”)

3. transaction_counts = hive_context.sql(“SELECT COUNT(transactions) FROM users”)