Spark DataFrames and SparkSQL

Game Plan

- Spark DataFrames motivation
- Spark DataFrames basics
- Working with Spark DataFrames and SparkSQL

Objectives

- Understand the benefits of Spark DataFrames over traditional RDDs
- Know how to instantiate and interact with a Spark DataFrame
- Know how to register a Spark DataFrame in order to be able to use SQL queries on the data
- Know how to spin up a spark cluster on AWS

Game Plan

- Spark DataFrames motivation

- Spark DataFrames basics

- Working with Spark DataFrames and SparkSQL

Objectives

- Understand the benefits of Spark DataFrames over traditional RDDs

- Know how to instantiate and interact with a Spark DataFrame

- Know how to register a Spark DataFrame in order to be able to use SQL queries on the data

- Know how to spin up a spark cluster on AWS

Why DataFrames?

- They provide an abstraction that simplifies working with structured datasets
- They can read and write data in a variety of structured formats
- They let you query the data using SQL.
- They are much faster than traditional RDD's

Why DataFrames?

- 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 DataFrames?

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 DataFrames?

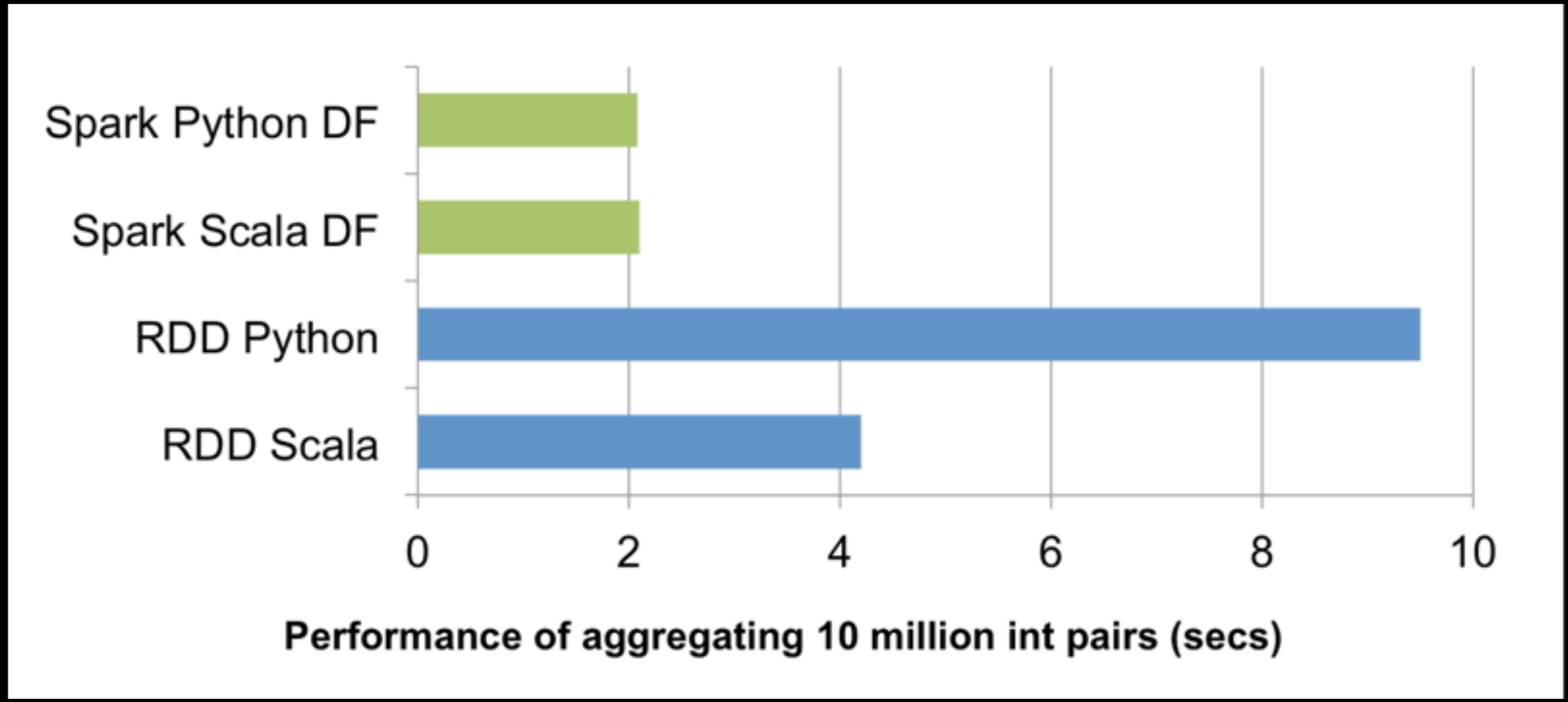
- With DataFrames, we can write this:

```
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 DataFrames?

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



Game Plan

- Spark DataFrames motivation
- Spark DataFrames basics
- Working with Spark DataFrames and SparkSQL

Objectives

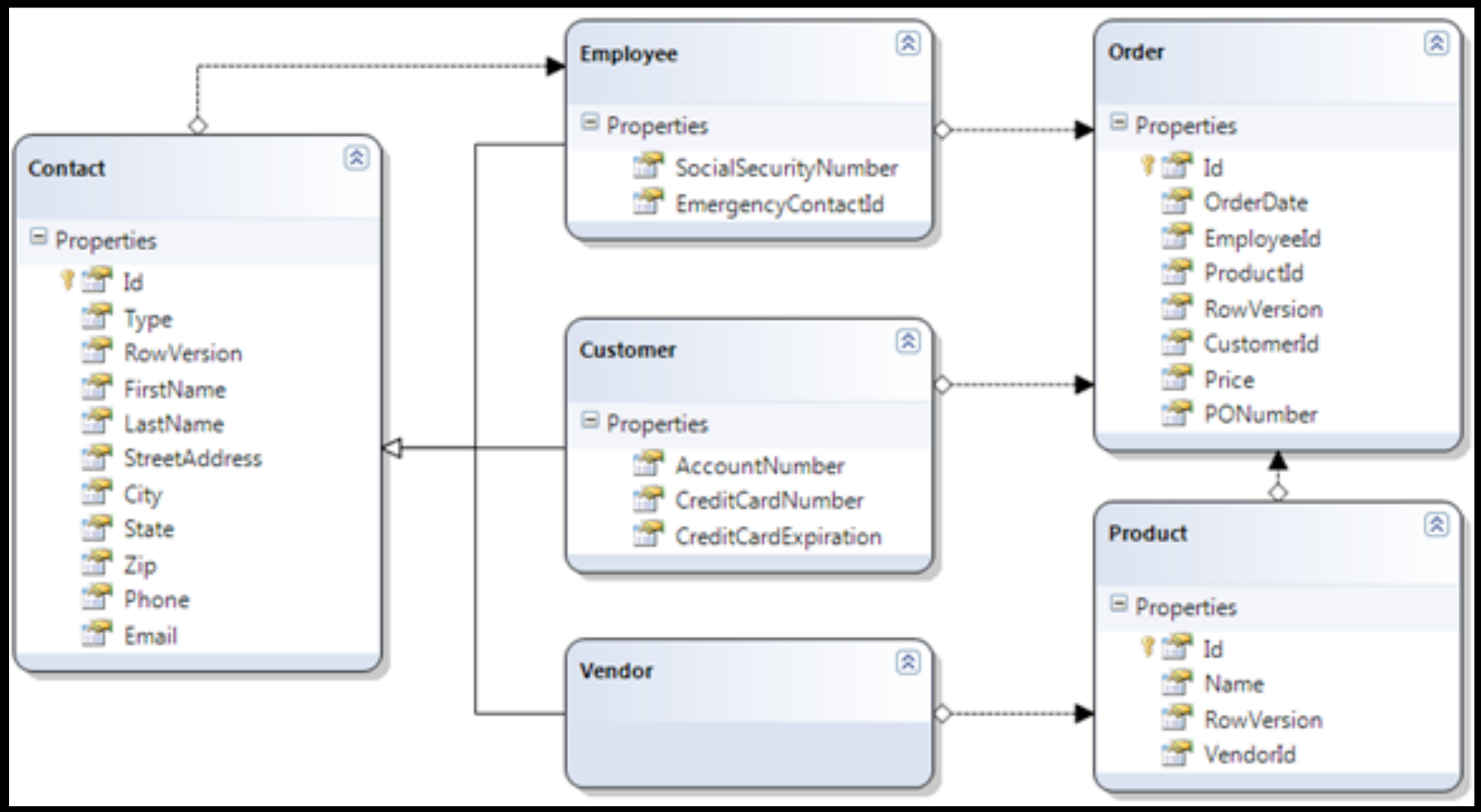
- Understand the benefits of Spark DataFrames over traditional RDDs
- Know how to instantiate and interact with a Spark DataFrame
- Know how to register a Spark DataFrame in order to be able to use SQL queries on the data
- Know how to spin up a spark cluster on AWS

RDD vs. DataFrame

Spark DataFrames are basically just RDD's, with some structure…

RDD vs. DataFrame

Or more specifically, a schema…



DataFrame Basics

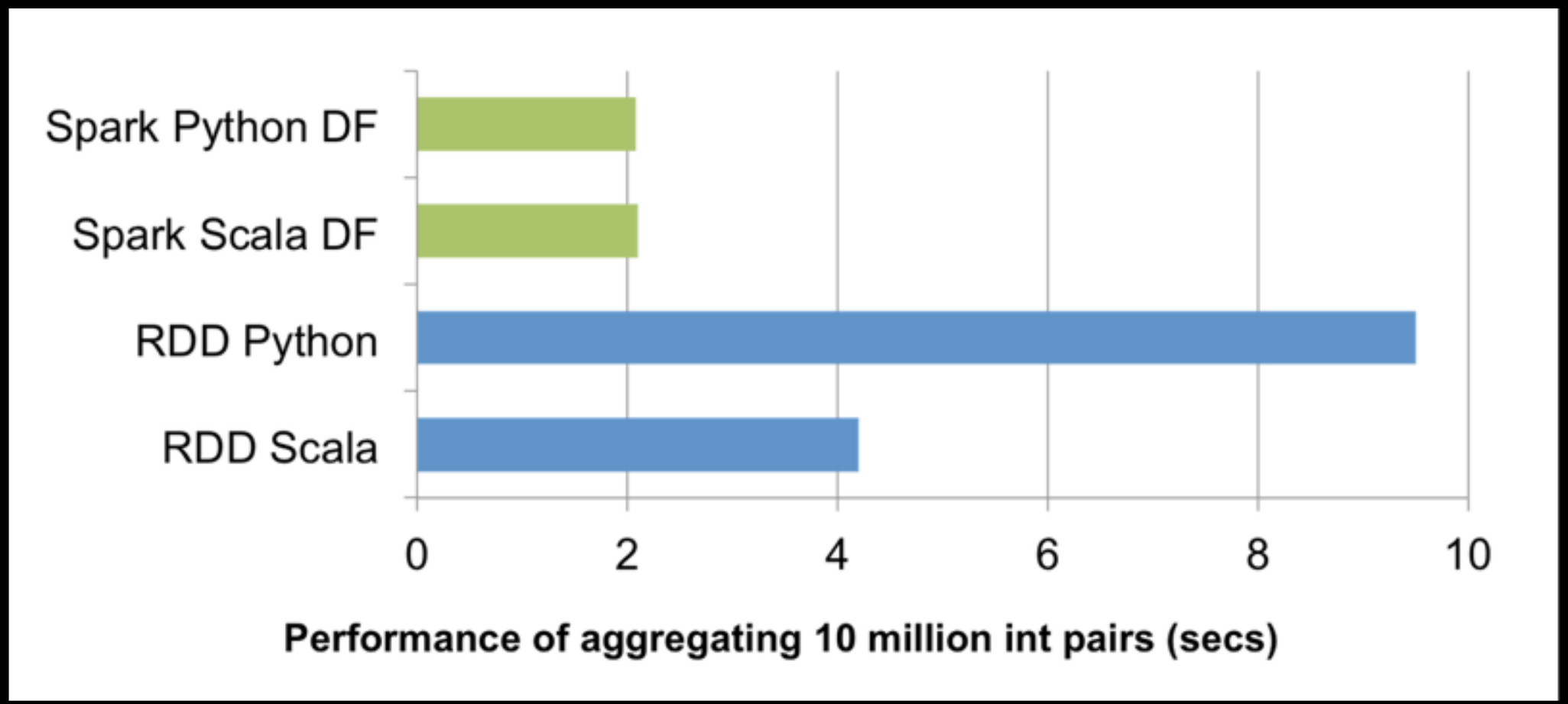
- A DataFrame contains an RDD of **Row** objects, each representing a record. A DataFrame is not technically an RDD, but we can effectively treat it as such.

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

Schema Importance

- Allows logical optimizations (e.g. predicate pushdown)
- Allows for compilation to Bytecode (Python specific)

Schema Importance



Game Plan

- Spark DataFrames motivation
- Spark DataFrames basics
- Working with Spark DataFrames and SparkSQL

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

Objectives

- Understand the benefits of Spark DataFrames over traditional RDDs
- Know how to instantiate and interact with a Spark DataFrame
- Know how to register a Spark DataFrame in order to be able to use SQL queries on the data
- Know how to spin up a spark cluster on AWS

SparkSQL

How do I get to SparkSQL?

1. data = hive_context.jsonFile(input_file)

2. data.registerTempTable(“users”)

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

Objectives

- Understand the benefits of Spark DataFrames over traditional RDDs
- Know how to instantiate and interact with a Spark DataFrame
- Know how to register a Spark DataFrame in order to be able to use SQL queries on the data
- Know how to spin up a spark cluster on AWS