

# Operational Tips for Deploying Apache® Spark™

Miklos Christine  
Solutions Architect

Databricks



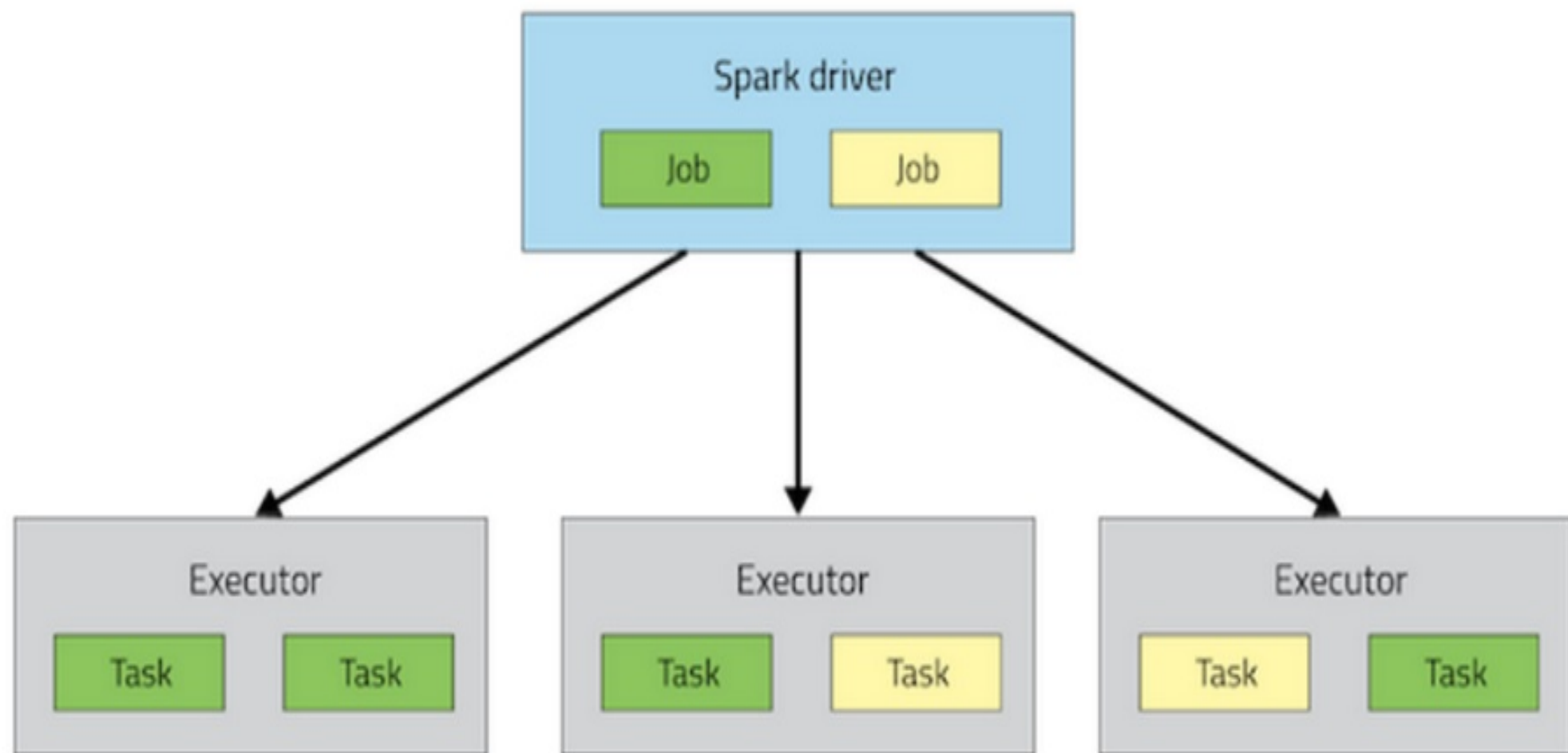
# \$ whoami

- Previously Systems Engineer @ Cloudera
- Deep Knowledge of Big Data Stack
- Apache Spark Expert
- Solutions Architect @ Databricks!

# What Will I Learn?

- Quick Apache Spark Overview
- Configuration Systems
- Pipeline Design Best Practices
- Debugging Techniques

# Apache Spark



# Apache Spark Configuration

Spark SQL +  
DataFrames

Streaming

MLlib  
*Machine Learning*

GraphX  
*Graph Computation*

Spark Core API

R

SQL

Python

Scala

Java



# Apache Spark Core Configurations

- **Command Line:**  
spark-defaults.conf  
spark-env.sh  
  
// Print SparkConfig  
sc.getConf().toString  
  
// Print Hadoop Config  
val hdConf =  
sc.hadoopConfiguration.iterator()  
while (hdConf.hasNext) {  
    println(hdConf.next().toString())  
}
- **Programmatically:**  
SparkConf()
- **Hadoop Configs:**  
core-site.xml  
hdfs-site.xml

# SparkSQL Configurations

- Set SQL Configs Through SQL Interface

```
SET key=value;
```

```
sqlContext.sql("SET spark.sql.shuffle.partitions=10;")
```

- Tools to see current configurations

```
// View SparkSQL Config Properties
```

```
val sqlConf = sqlContext.getAllConfs
```

```
sqlConf.foreach(x => println(x._1 + " : " + x._2))
```

# Apache Spark Pipeline Design

- File Formats
- Compression Codecs
- Apache Spark APIs
- Job Profiles



# File Formats

- Text File Formats
  - CSV
  - JSON

| A  | B  | C  |
|----|----|----|
| A1 | B1 | C1 |
| A2 | B2 | C2 |
| A3 | B3 | C3 |

- Avro Row Format

|    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|
| A1 | B1 | C1 | A2 | B2 | C2 | A3 | B3 | C3 |
|----|----|----|----|----|----|----|----|----|

- Parquet Columnar Format

|    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|
| A1 | A2 | A3 | B1 | B2 | B3 | C1 | C2 | C3 |
|----|----|----|----|----|----|----|----|----|

User Story:

260GB CSV Data Converted to 23GB Parquet

# Compression

- Choose and Analyze Compression Codecs
  - Snappy, Gzip, LZ0
- Configuration Parameters
  - `io.compression.codecs`
  - `spark.sql.parquet.compression.codec`

# Small Files Problem

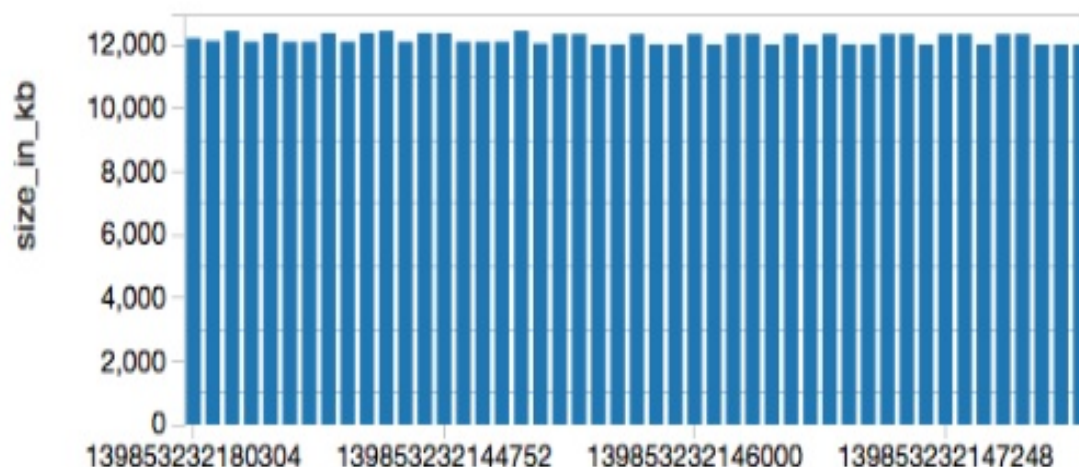
- Small files problem still exists

- Metadata loading

- Use `coalesce()`

```
display(df)
```

▶ (2) Spark Jobs

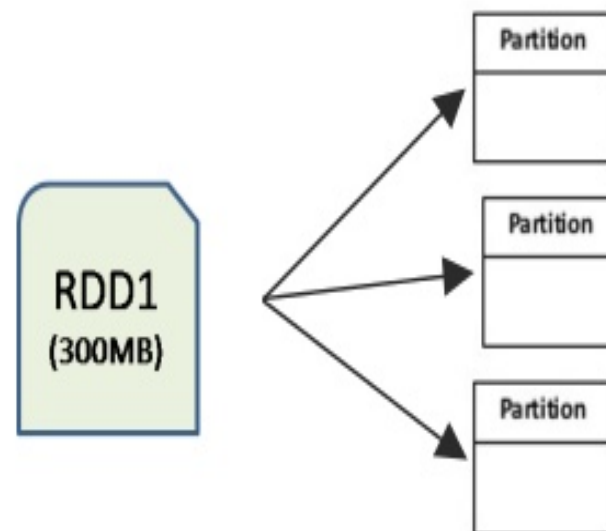


Ref:

<http://spark.apache.org/docs/latest/api/python/pyspark.sql.html#pyspark.sql.DataFrame>

# What are Partitions?

- 2 Types of Partitioning
  - Spark
  - Table Level



# Get Number of Spark

```
> df.rdd.getNumPartitions()
```

40

# Partition Controls

- **Apache Spark APIs**

- `repartition()`
- `coalesce()`

```
# Re-partition a DataFrame
```

```
> df_10 = df.repartition(10)
```

```
df = sqlContext.read.\
    jdbc(url=jdbcUrl, \
          table='employees', \
          column='emp_no', \
          lowerBound=1, \
          upperBound=100000, \
          numPartitions=100)
```

```
df.repartition(20).write.\
    parquet('/mnt/mwc/jdbc_part/')
```

# Table Partitions

- Partition by a column value within the table

```
> df.write.\  
    partitionBy("colName").\  
    saveAsTable("tableName")
```

```
ls -l /dbfs/user/hive/warehouse/tableName/
```

```
total 0  
-rw-r--r-- 1 root root 0 Jan  1  1970 _SUCCESS  
drwxr-xr-x 1 root root 0 Jan  1  1970 year=2014  
drwxr-xr-x 1 root root 0 Jan  1  1970 year=2015
```

```
Command took 0.38s
```



# Those Other Partitions

- **SparkSQL Shuffle Partitions**

`spark.sql.shuffle.partitions`

```
sqlCtx.sql("set spark.sql.shuffle.partitions=600")
```

```
sqlCtx.sql("select a1.name, a2.name from adult a1 \  
    join adult a2 \  
    where a1.age = a2.age")
```

```
sqlCtx.sql("select count(distinct(name)) from adult")
```

# How Does This Help?

- Q: Will increasing my cluster size help with my job?
- A: It depends.

```
> %sql
-- 1.6
set spark.sql.shuffle.partitions=200;
select count(distinct(author)) from reddit_all where year = 2015;
```

▼ (1) Spark Jobs

- ▼ Job 13 [View](#) (Stages: 3/3)
  - Stage 13: 240/240 ⓘ
  - Stage 14: 200/200 ⓘ
  - Stage 15: 1/1 ⓘ

\_c0

5006186

Command took 28.17s

```
> %sql
-- 1.6
set spark.sql.shuffle.partitions=40;
select count(distinct(author)) from reddit_all where year = 2015;
```

▼ (1) Spark Jobs

- ▼ Job 14 [View](#) (Stages: 3/3)
  - Stage 16: 240/240 ⓘ
  - Stage 17: 40/40 ⓘ
  - Stage 18: 1/1 ⓘ

\_c0

5006186

Command took 19.53s

# Apache Spark Job Profiles

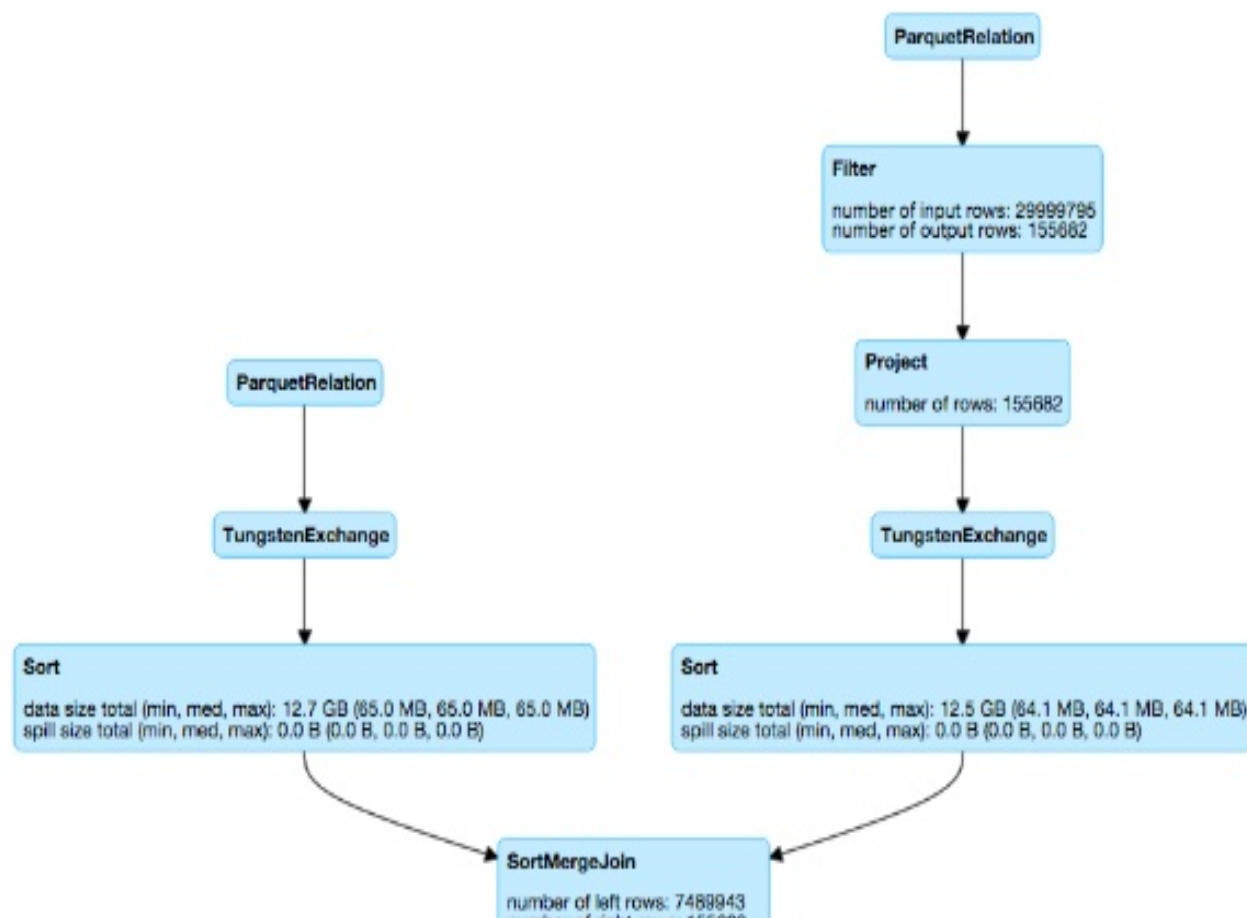
- Leverage Spark UI
  - SQL
  - Streaming

## Details for Query 49

Submitted Time: 2016/02/10 02:06:25

Duration: 12 s

Succeeded Jobs: 85



# Apache Spark Job Profiles

## Aggregated Metrics by Executor

| Executor ID ▲ | Address  | Task Time | Total Tasks | Failed Tasks | Succeeded Tasks | Input Size / Records | Shuffle Write Size / Records |
|---------------|--|-----------|-------------|--------------|-----------------|----------------------|------------------------------|
| 0             | ip-10-0-197-52.us-west-2.compute.internal:45222  | 39 min    | 291         | 0            | 291             | 10.0 GB / 82423      | 11.9 KB / 291                |
| 1             | ip-10-0-238-218.us-west-2.compute.internal:41578 | 39 min    | 292         | 0            | 292             | 10.0 GB / 82795      | 12.0 KB / 292                |
| 2             | ip-10-0-229-111.us-west-2.compute.internal:43837 | 39 min    | 294         | 0            | 294             | 10.1 GB / 82721      | 12.1 KB / 294                |
| 3             | ip-10-0-176-36.us-west-2.compute.internal:47072  | 39 min    | 294         | 0            | 294             | 10.2 GB / 84025      | 12.1 KB / 294                |
| 4             | ip-10-0-207-37.us-west-2.compute.internal:41034  | 39 min    | 293         | 0            | 293             | 10.1 GB / 82656      | 12.0 KB / 293                |
| 5             | ip-10-0-156-34.us-west-2.compute.internal:44449  | 39 min    | 298         | 0            | 298             | 10.2 GB / 83917      | 12.2 KB / 298                |
| 6             | ip-10-0-156-35.us-west-2.compute.internal:58803  | 39 min    | 298         | 0            | 298             | 10.2 GB / 84277      | 12.2 KB / 298                |
| 7             | ip-10-0-232-3.us-west-2.compute.internal:55489   | 39 min    | 297         | 0            | 297             | 10.3 GB / 84628      | 12.2 KB / 297                |
| 8             | ip-10-0-232-2.us-west-2.compute.internal:38085   | 39 min    | 296         | 0            | 296             | 10.2 GB / 83582      | 12.1 KB / 296                |
| 9             | ip-10-0-207-36.us-west-2.compute.internal:55661  | 39 min    | 288         | 0            | 288             | 9.9 GB / 81804       | 11.8 KB / 288                |
| 10            | ip-10-0-178-163.us-west-2.compute.internal:59616 | 39 min    | 294         | 0            | 294             | 10.1 GB / 83350      | 12.1 KB / 294                |
| 11            | ip-10-0-178-162.us-west-2.compute.internal:41800 | 39 min    | 295         | 0            | 295             | 10.2 GB / 83822      | 12.1 KB / 295                |

# Apache Spark Job Profiles

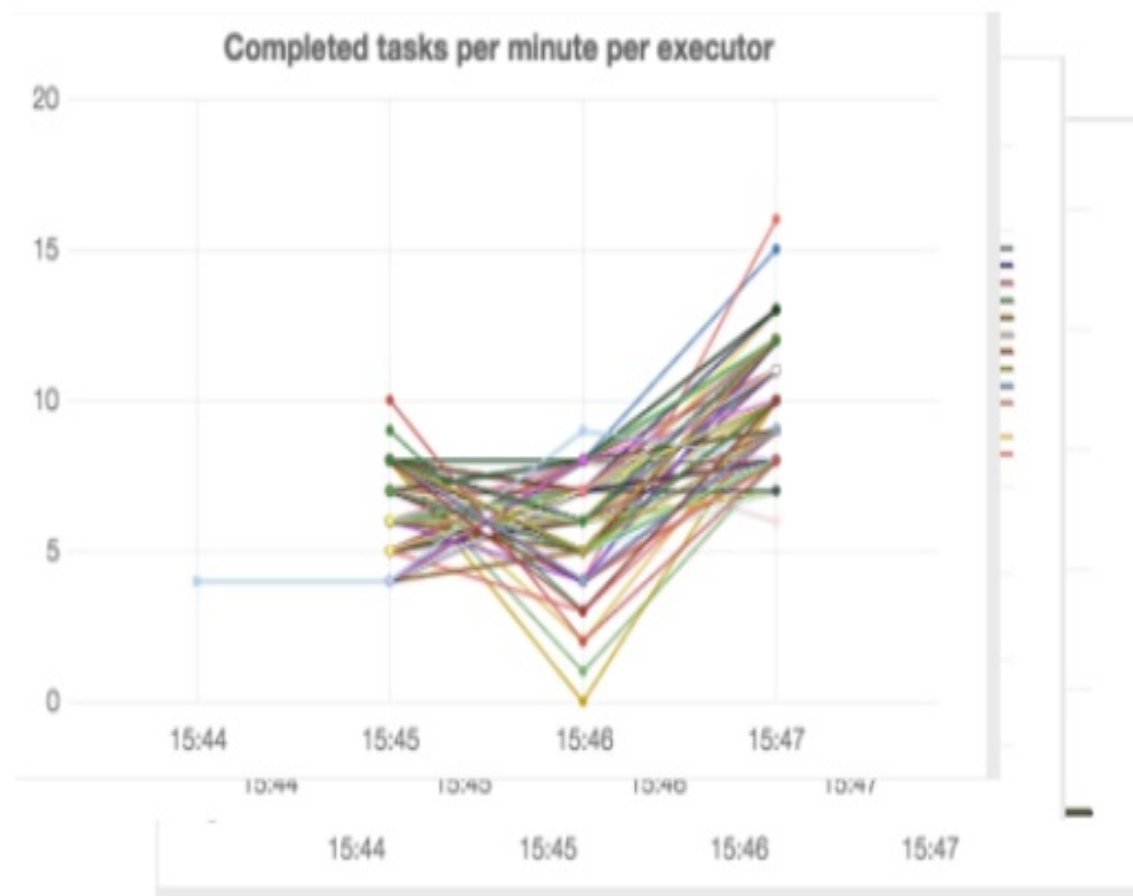
## Tasks

| Index ▲ | ID    | Attempt | Status  | Locality Level | Executor ID / Host                              | Launch Time         | Duration | GC Time | Input Size / Records      | Errors |
|---------|-------|---------|---------|----------------|---|---------------------|----------|---------|---------------------------|--------|
| 0       | 16607 | 0       | SUCCESS | PROCESS_LOCAL  | 18 / ip-10-0-171-33.us-west-2.compute.internal  | 2016/02/09 19:12:40 | 24 s     | 0.2 s   | 46.8 MB (hadoop) / 614400 |        |
| 1       | 16608 | 0       | SUCCESS | PROCESS_LOCAL  | 19 / ip-10-0-214-114.us-west-2.compute.internal | 2016/02/09 19:12:40 | 23 s     |         | 46.8 MB (hadoop) / 614400 |        |
| 2       | 16609 | 0       | SUCCESS | PROCESS_LOCAL  | 18 / ip-10-0-171-33.us-west-2.compute.internal  | 2016/02/09 19:12:40 | 4 s      |         | 6.5 MB (hadoop) / 85355   |        |
| 3       | 16610 | 0       | SUCCESS | PROCESS_LOCAL  | 19 / ip-10-0-214-114.us-west-2.compute.internal | 2016/02/09 19:12:40 | 23 s     |         | 46.8 MB (hadoop) / 614400 |        |
| 4       | 16611 | 0       | SUCCESS | PROCESS_LOCAL  | 18 / ip-10-0-171-33.us-west-2.compute.internal  | 2016/02/09 19:12:40 | 24 s     | 0.2 s   | 46.8 MB (hadoop) / 614400 |        |



# Apache Spark Job Profile: Metrics

- Monitoring & Metrics
  - Spark
  - Servers
- Toolset
  - Ganglia
  - Graphite



Ref:

<http://www.hammerlab.org/2015/02/27/monitoring-spark-with-graphite-and-grafana/>



# Debugging Apache Spark

- Analyze the Driver's stacktrace.
- Analyze the executors stacktraces
  - Find the initial executor's failure.
- Review metrics
  - Memory
  - Disk
  - Networking

# Debugging Apache Spark

- Know your tools: JDBC vs ODBC. How to test? What can I test?
  - RedShift / Mysql / Tableau to Apache Spark ,etc.

- Json SparkSQL for corrupt records

```
sqlCtx.read.json("/jsonFiles/").registerTempTable("jsonTable")  
sqlCtx.sql("SELECT _corrupt_record \  
            FROM jsonTable \  
            WHERE _corrupt_record IS NOT NULL")
```

- Steps to debug SQL issues
  - Where's the data, what's the DDL?

# Top Support Issues

- OutOfMemoryErrors
  - Driver
  - Executors
- Out of Disk Space Issues
- Long GC Pauses
- API Usage

# Top Support Issues

- Use builtin functions instead of custom UDFs
  - import [`pyspark.sql.functions`](#)
  - import `org.apache.spark.sql.functions`
- Examples:
  - `to_date()`
  - `get_json_object()`
  - `monotonically_increasing_id()`
  - `hour()` / `minute()`

# Top Support Issues

- SQL query not returning new data
  - `REFRESH TABLE <table_name>`
- Exported Parquet from External Systems
  - `spark.sql.parquet.binaryAsString`
- Tune number of Shuffle Partitions
  - `spark.sql.shuffle.partitions`

# Try Apache Spark with Databricks

- Download notebook for this talk at:  
[dbricks.co/xyz](http://dbricks.co/xyz)
- Try latest version of Apache Spark and preview of Spark 2.0

<http://databricks.com/try>

Create Cluster

New Cluster

Cancel

Create Cluster

Cluster Name

New Cluster

Spark Version

✓ Spark 2.0 (apache/branch-2.0 preview)

Spark 1.3.0 (Hadoop 1)

Spark 1.4.1 (Hadoop 1)

Spark 1.5.2 (Hadoop 1)

Spark 1.6.0 (Hadoop 1)

Spark 1.6.1 (Hadoop 1)

Spark 1.6.1 (Hadoop 2)

I automatically t  
de your Databri



# Thank you.

[mwc@databricks.com](mailto:mwc@databricks.com)

[https://www.linkedin.com/in/mrchristine@miklos\\_c](https://www.linkedin.com/in/mrchristine@miklos_c)