Cassandra and SparkSQL

You don't need Functional Programming for Fun!

Russell (left) and Cara (right)

- Software Engineer

DATASTAX

- Spark-Cassandra Integration since Spark 0.9
- Cassandra since Cassandra 1.2
- 2 Year Scala Convert
- Still not comfortable talking about Monads in public

A Story in 3 Parts

- Why SparkSQL?
- The Spark SQL Thrift Server
- Writing SQL for Spark

You have lots of options why Spark SQL?

- Scala?
- Java?
- Python?
- R?
- Notebooks?
- Erlang?

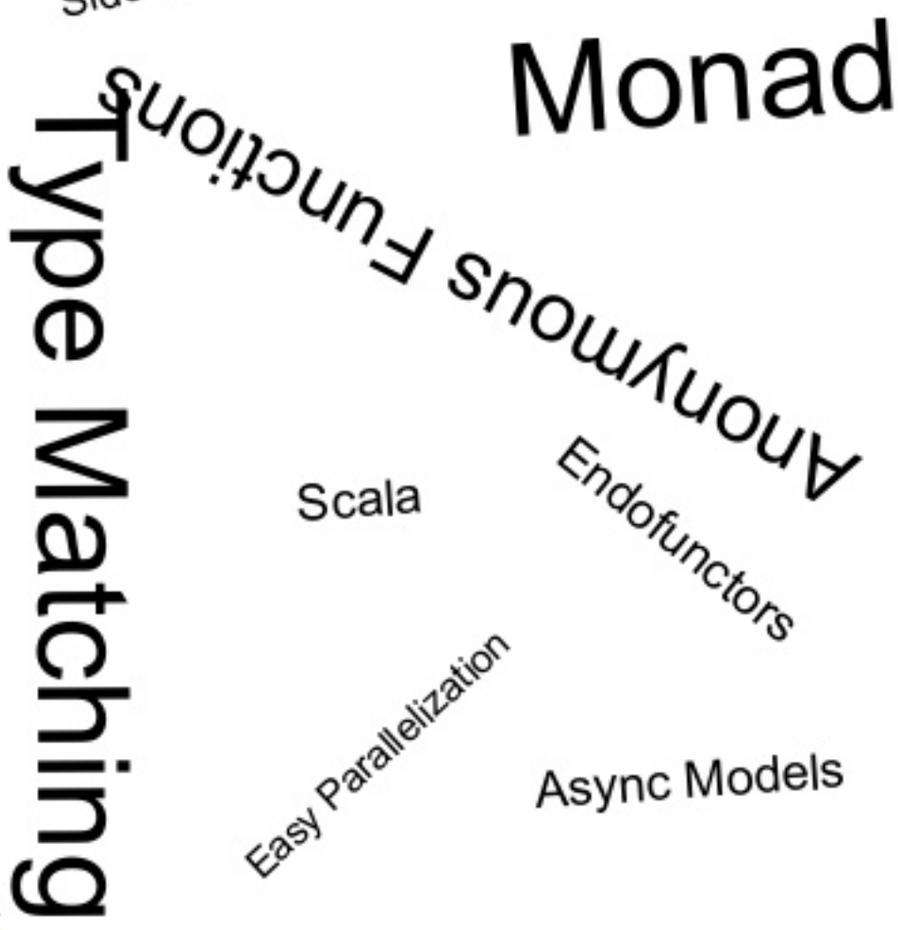
Spark is A Powerful Analytics Tool Built on Scala

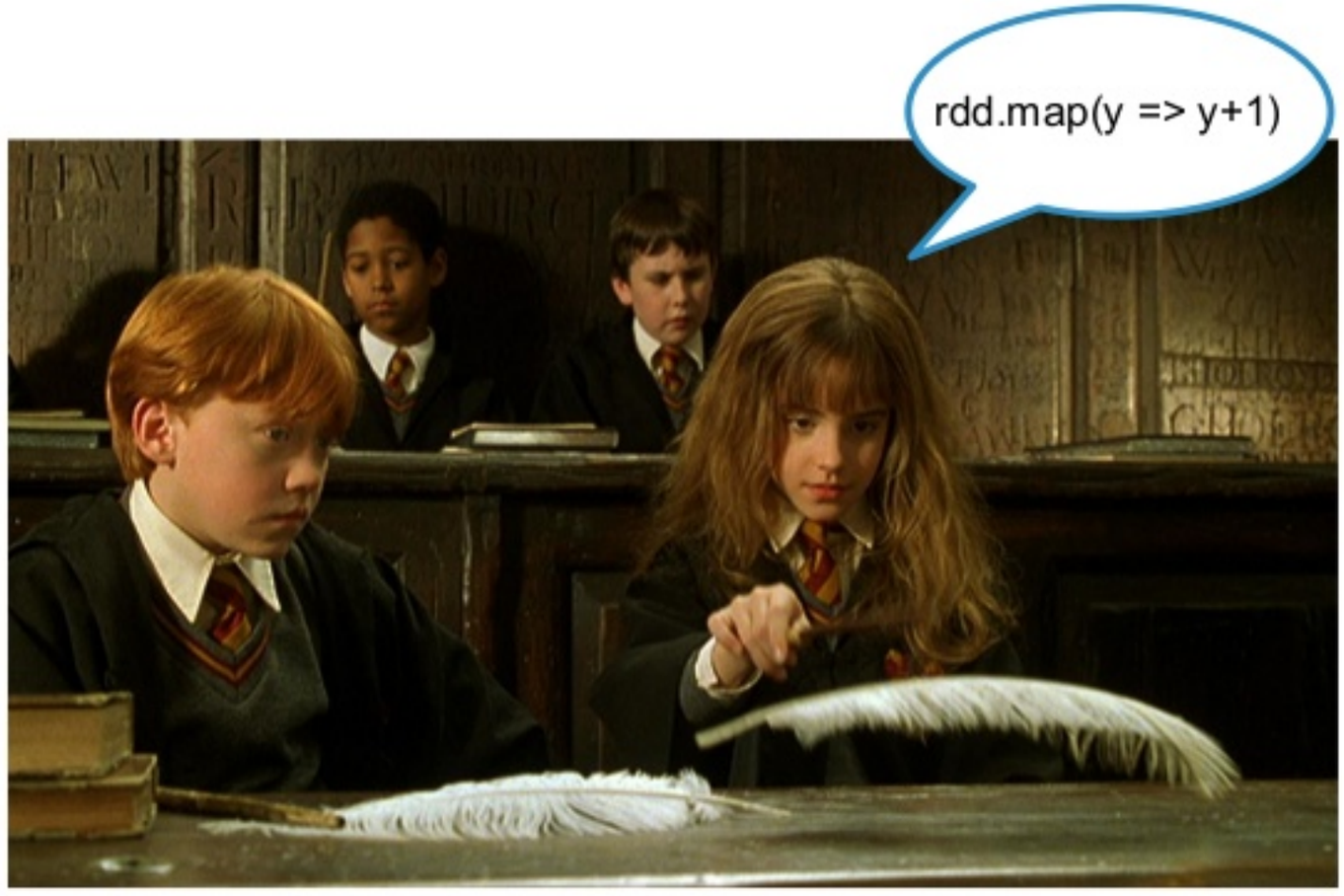
Distributed Analytics Platform
with In Memory Capabilities

Lots of new concepts:

- RDDs
- DataSets
- Streaming
- Serialization
- Functional Programming

Functional Programming Is Awesome





Functional Programming can be Hard

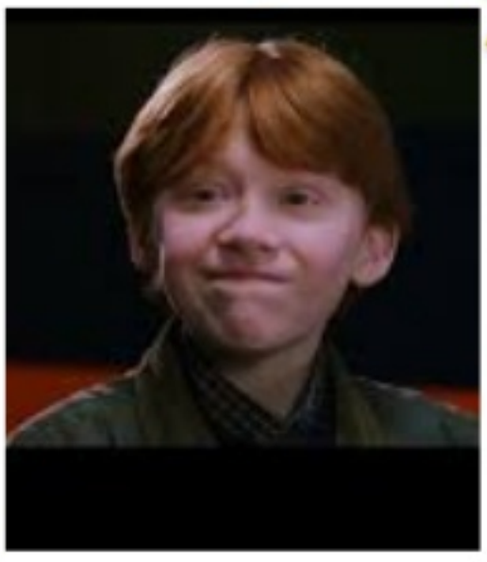
Learning Takes Time

Spark SQL Provides A Familiar and Easy API

Use SQL to access the Power of Spark

Spark Sql Provides A Familiar and Easy API

SQL



Codegen!
Optimization!
Predicate Pushdowns

Catalyst

Distributed Work

SPARK SUMMIT 2017

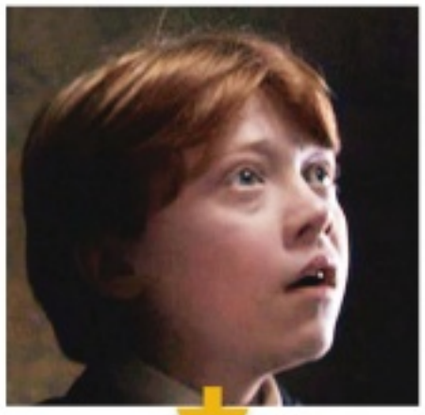
It still takes Scala/Java/Python/… Code.

```
import org.apache.spark.sql.cassandra._
val df = spark
  .read
  .cassandraFormat("tab", "ks")
  .load
 df.createTempView("tab")
spark.sql("SELECT * FROM tab").show
+---+---+---+
|  k|  c|  v|
+---+---+---+
|  1|  1|  1|
|  1|  2|  2|
```

It still takes Scala/Java/Python/… Code.

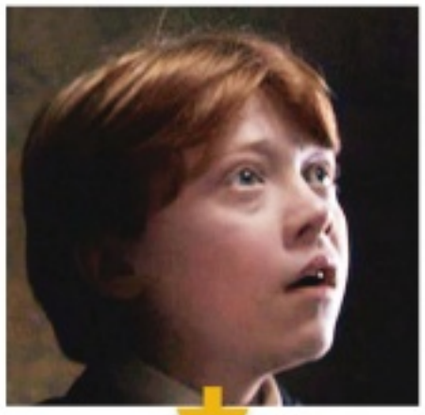
```
import org.apache.spark.sql.cassandra._
val df = spark
  .read
  .cassandraFormat("tab", "ks")
  .load
 df.createTempView("tab")
spark.sql("SELECT * FROM tab").show
+---+---+---+
|  k|  c|  v|
+---+---+---+
|  1|  1|  1|
|  1|  2|  2|
```

For exploration we have the Spark-SQL Shell



```
spark-sql> SELECT * FROM ks.tab;
1    2    2
1    3    3
```

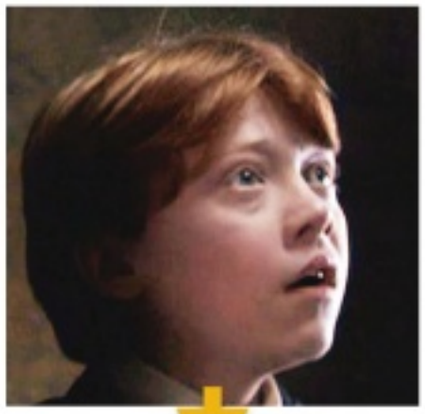
For exploration we have the Spark-SQL Shell



```
spark-sql> SELECT * FROM ks.tab;
1    2    2
1    3    3
```

SparkSession

For exploration we have the Spark-SQL Shell

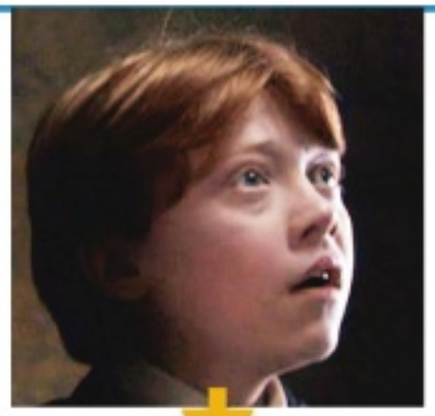


```
spark-sql> SELECT * FROM ks.tab;
1    2    2
1    3    3
```

SparkSession

Executor | Executor | Executor | Executor | Executor

Not really good for multiple-users



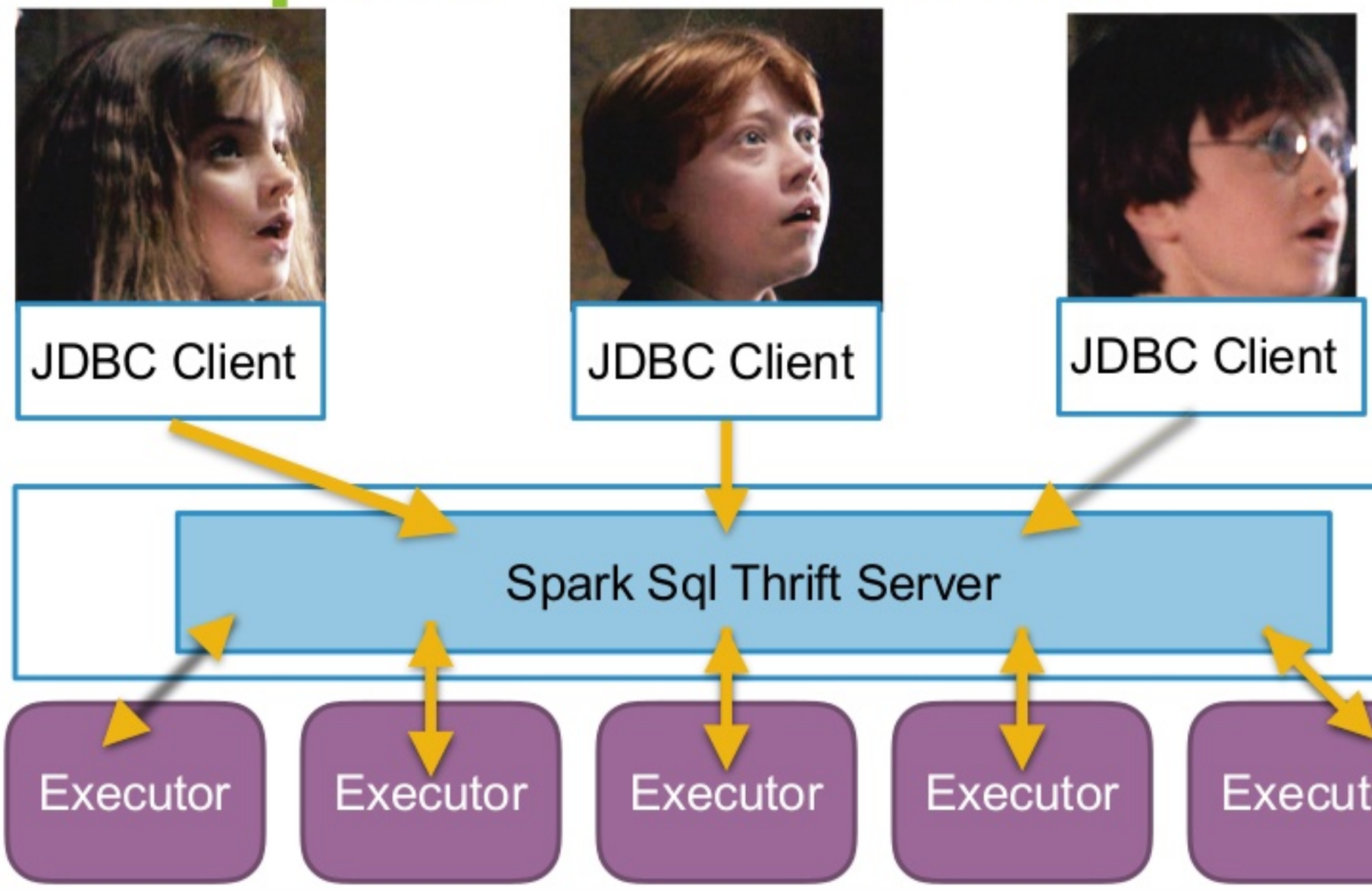
```
spark-sql> SELECT * FROM ks.tab;
1    2    2
1    3    3
```

SparkSession

Executor | Executor | Executor | Executor | Executor

SPARK SUMMIT 2017

Enter Spark Thrift Server



The Spark Sql Thrift Server is a Spark Application

- Built on HiveServer2
- Single Spark Context
- Clients Communicate with it via JDBC
- Can use all SparkSQL
- Fair Scheduling
- Clients can share Cached Resources
- Security

The Spark Sql ThriftServer is a Spark Application

- Built on HiveServer2
- Single Spark Context
- Clients Communicate with it via JDBC
- Can use all SparkSQL
- **Fair Scheduling**
- **Clients can share Cached Resources**
- Security

Fair Scheduling is Sharing

FIFO

Time

Fair Scheduling is Sharing

FIFO

Time

Fair Scheduling is Sharing

FIFO

Time

Fair Scheduling is Sharing

FIFO

Time

Fair Scheduling is Sharing

FIFO

Time


FAIR

Fair Scheduling is Sharing

FIFO

Time


FAIR

Fair Scheduling is Sharing

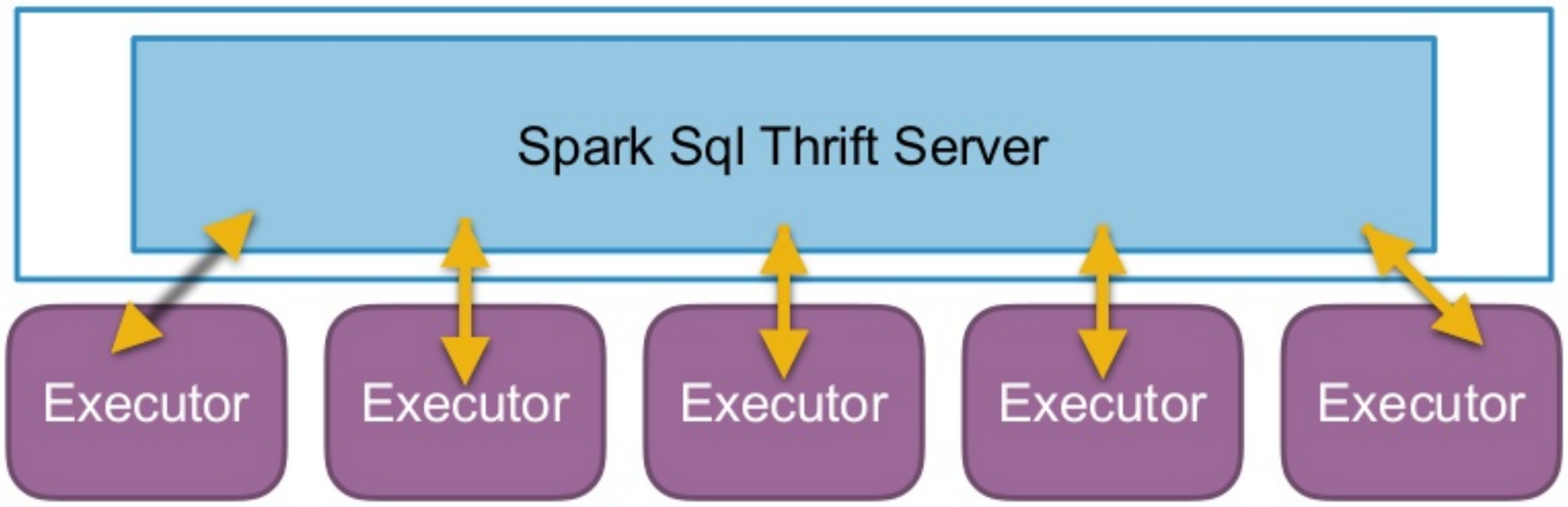
FIFO

Time


FAIR

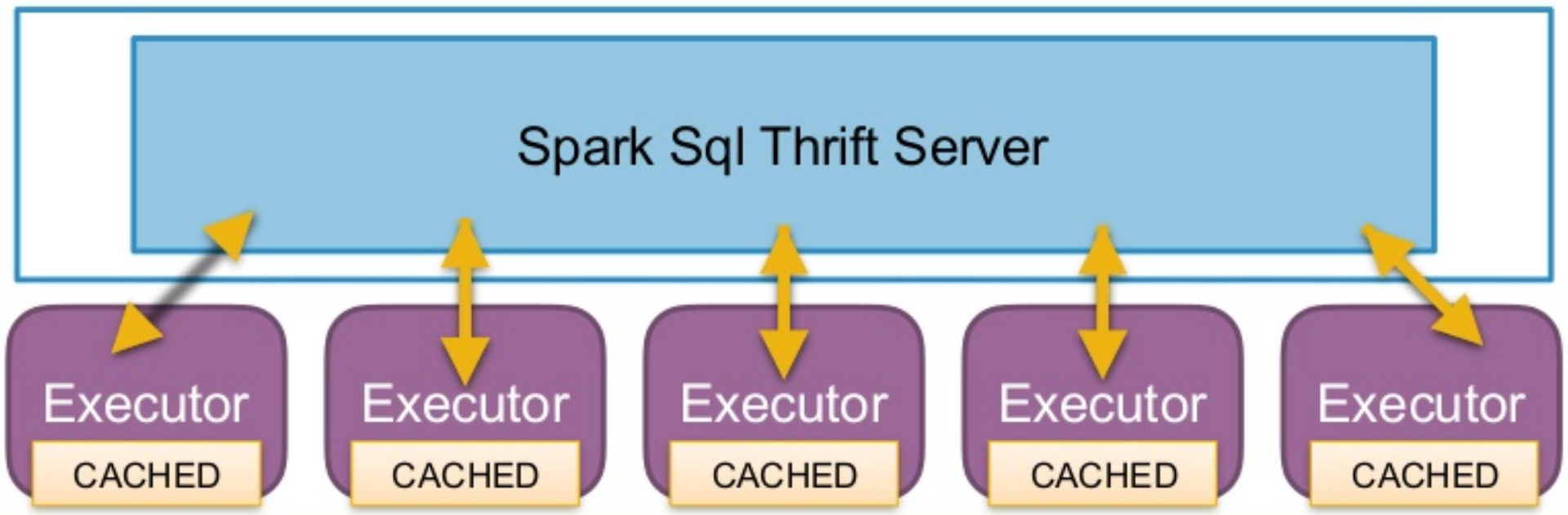
SingleContext can Share Cached Data

cache TABLE today select * from ks.tab where date = today;



SingleContext can Share Cached Data

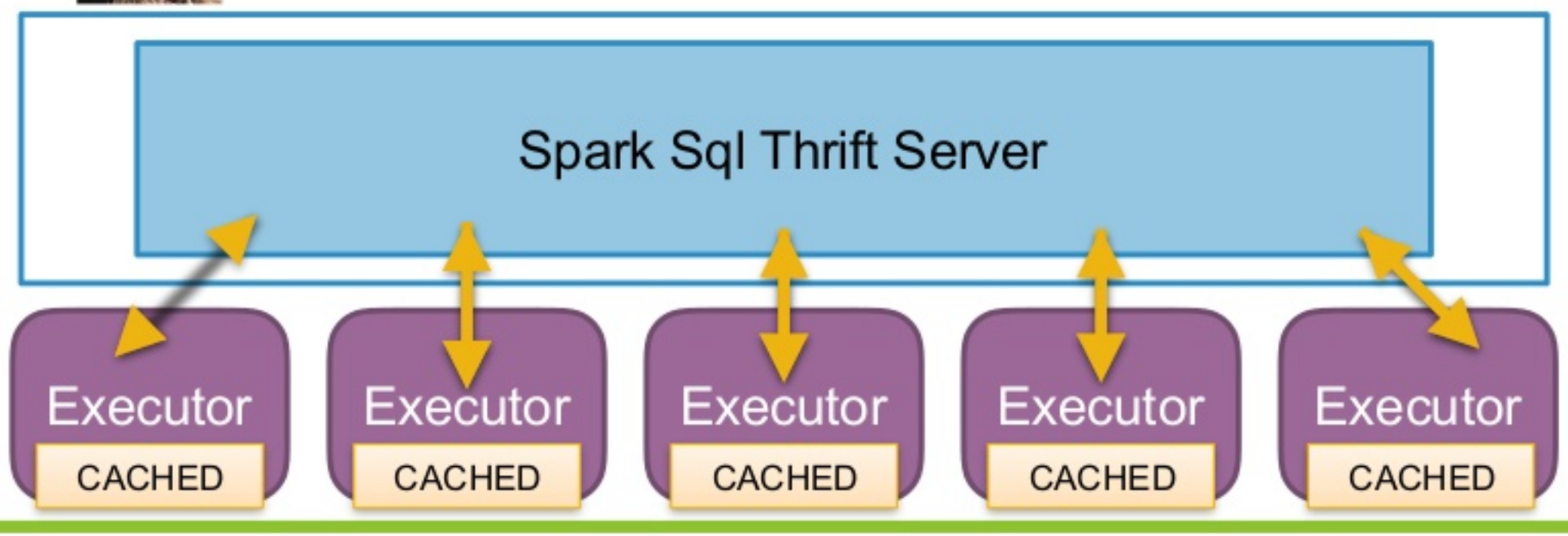
cache TABLE today select * from ks.tab where date = today;



SingleContext can Share Cached Data

cache TABLE today select * from ks.tab where date = today;

SELECT * from TODAY where age > 5



How to use it

Starts from the command line and can use all Spark Submit Args

- ./sbin/start-thriftserver.sh
- dse spark-sql-thriftserver start

Use with all of your favorite Spark Packages like the Spark Cassandra Connector!

--packages com.datastax.spark:spark-cassandra-connector_2.11:2.0.2
--conf spark.cassandra.connection.host=127.0.0.1

How to use it

Starts from the command line and can use all Spark Submit Args

- ./sbin/start-thriftserver.sh
- dse spark-sql-thriftserver start

```
starting org.apache.spark.sql.hive.thriftserver.HiveThriftServer2
```

Hive? Wait I though we were Doing Spark

- Built on HiveServer2

Why does it say Hive everywhere?

```
starting org.apache.spark.sql.hive.thriftserver.HiveThriftServer2
```

A Brief History of the Spark Thrift Server

- Thrift?
- Hive?

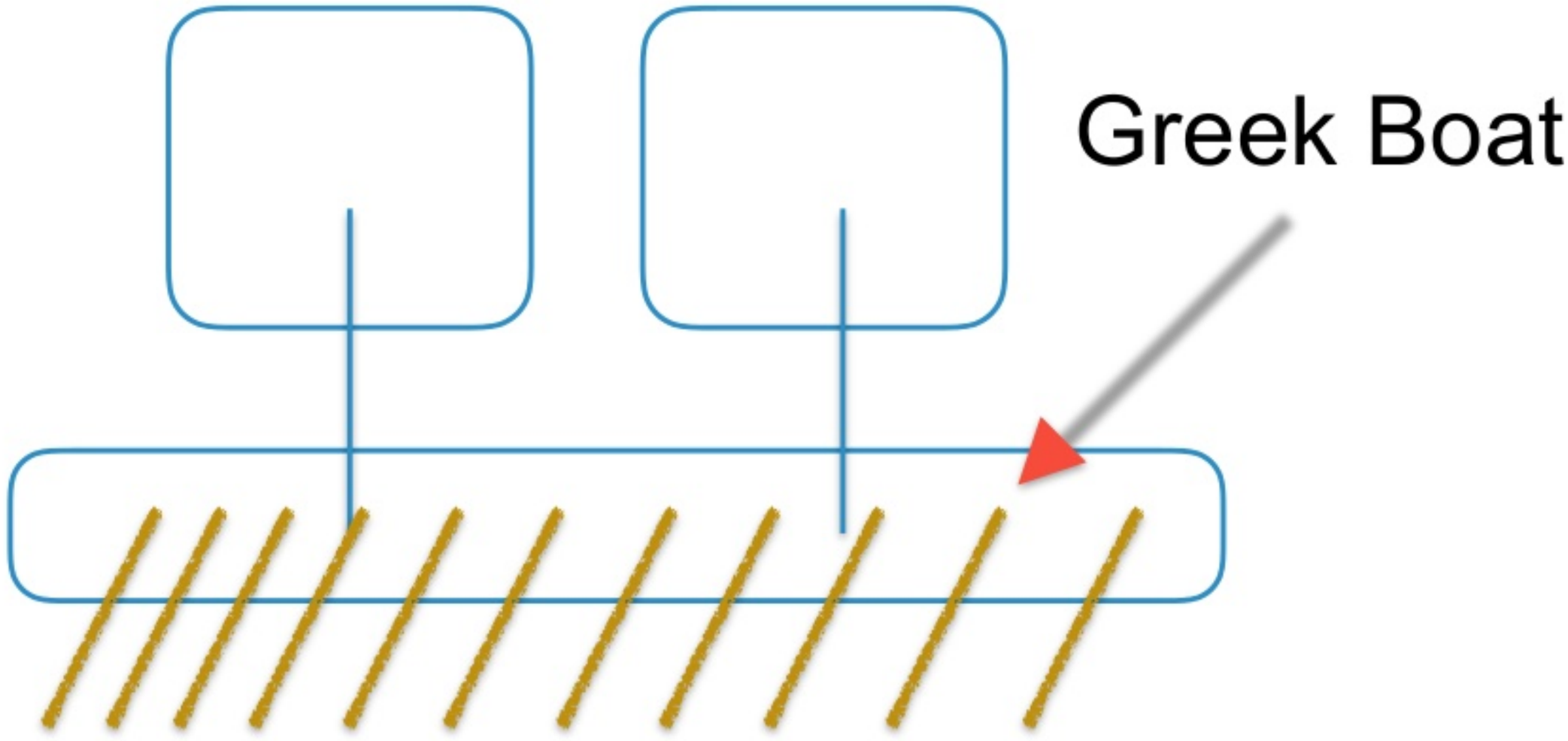
They are not the Same

Time for a quick history

Have you heard of the "Ship of Theseus?"

More Greek stuff ..

When you replace all the parts of a thing Does it Remain the Same?



When you replace all the parts of a thing Does it Remain the Same?

SharkServer

Hive Parser

Hive Optimization

Map-Reduce

Spark Execution

JDBC Results

When you replace all the parts of a thing Does it Remain the Same?

~~SharkServer~~

ThriftServer

Hive Parser

Hive Optimization

Map-Reduce

Spark Execution

JDBC Results

When you replace all the parts of a thing Does it Remain the Same?

ThriftServer

Hive Parser

Catalyst

Schema RDDs

Spark Execution

JDBC Results

When you replace all the parts of a thing Does it Remain the Same?

ThriftServer

Hive Parser

Catalyst

DataFrames

Spark Execution

JDBC Results

When you replace all the parts of a thing Does it Remain the Same?

ThriftServer

Hive Parser

Catalyst

Datasets

Spark Execution

JDBC Results

SPARK SUMMIT 2017

When you replace all the parts of a thing Does it Remain the Same?

ThriftServer

JDBC Results

Spark Parser

Catalyst

DataSets

Spark Execution

Almost all Spark now

ThriftServer

JDBC Results

Spark Parser

Catalyst

DataSets

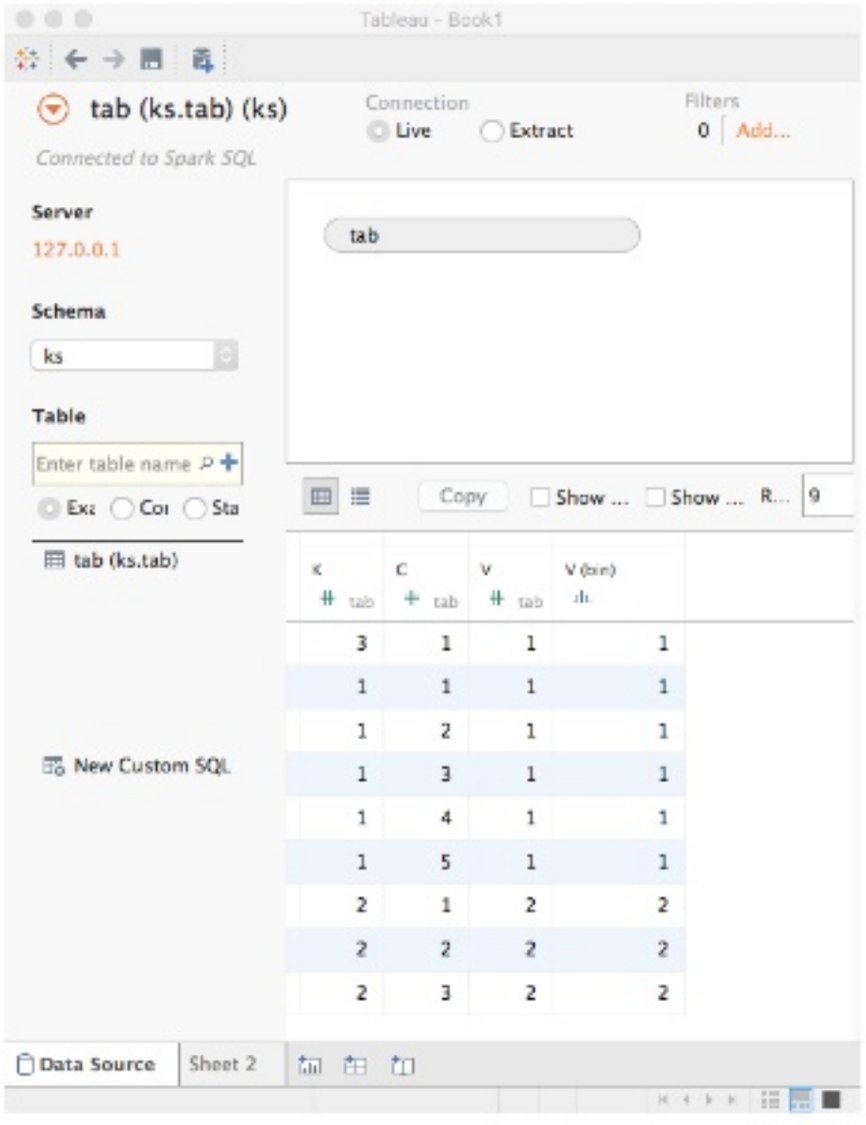
Spark Execution

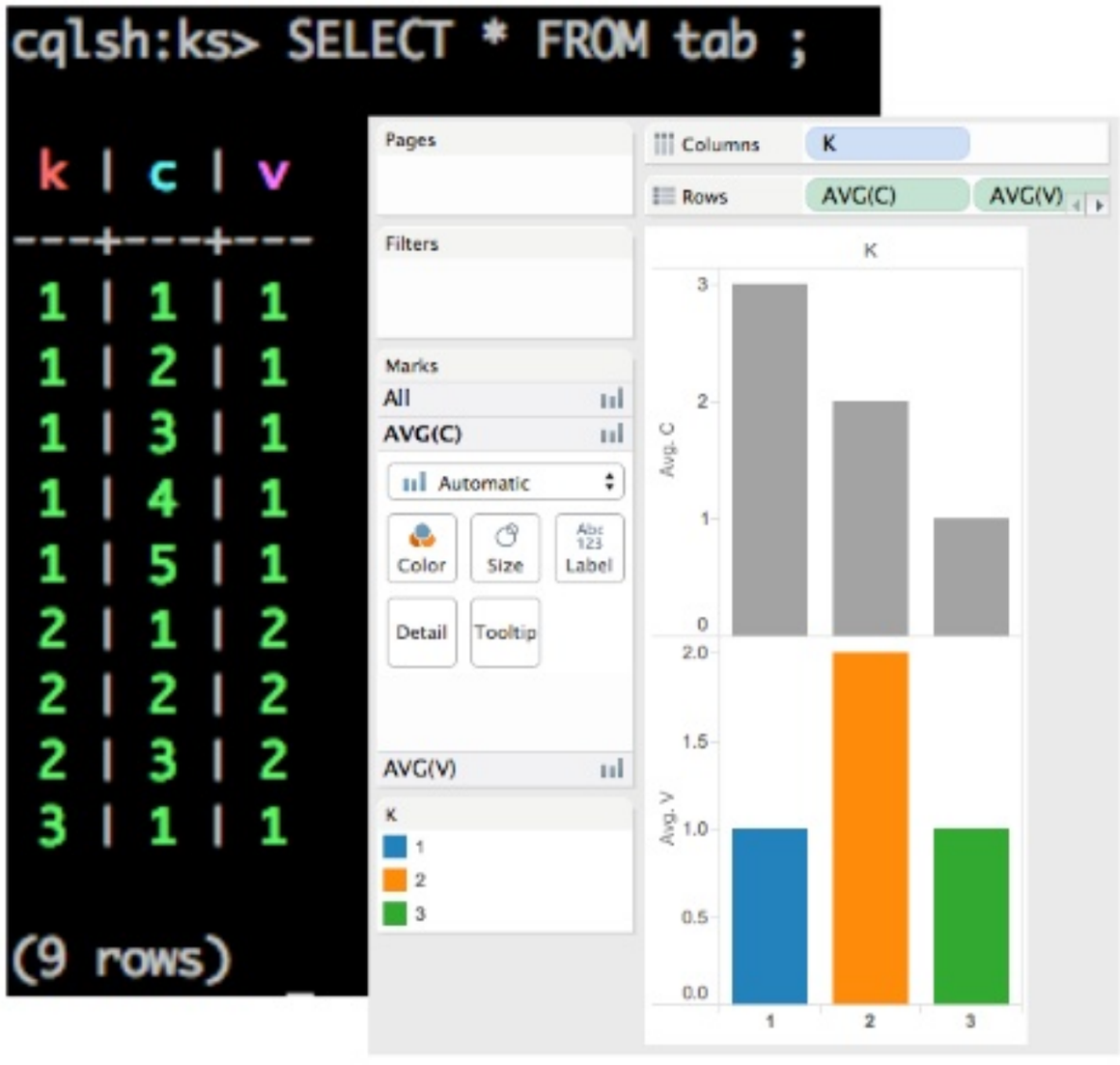
Connecting with Beeline (JDBC Client)

```
./bin/beeline
dse beeline
!connect jdbc:hive2://localhost:10000
```

Even More Hive!

Connect Tableau to Cassandra





The Full JDBC/ODBC Ecosystem Can Connect to ThriftServer

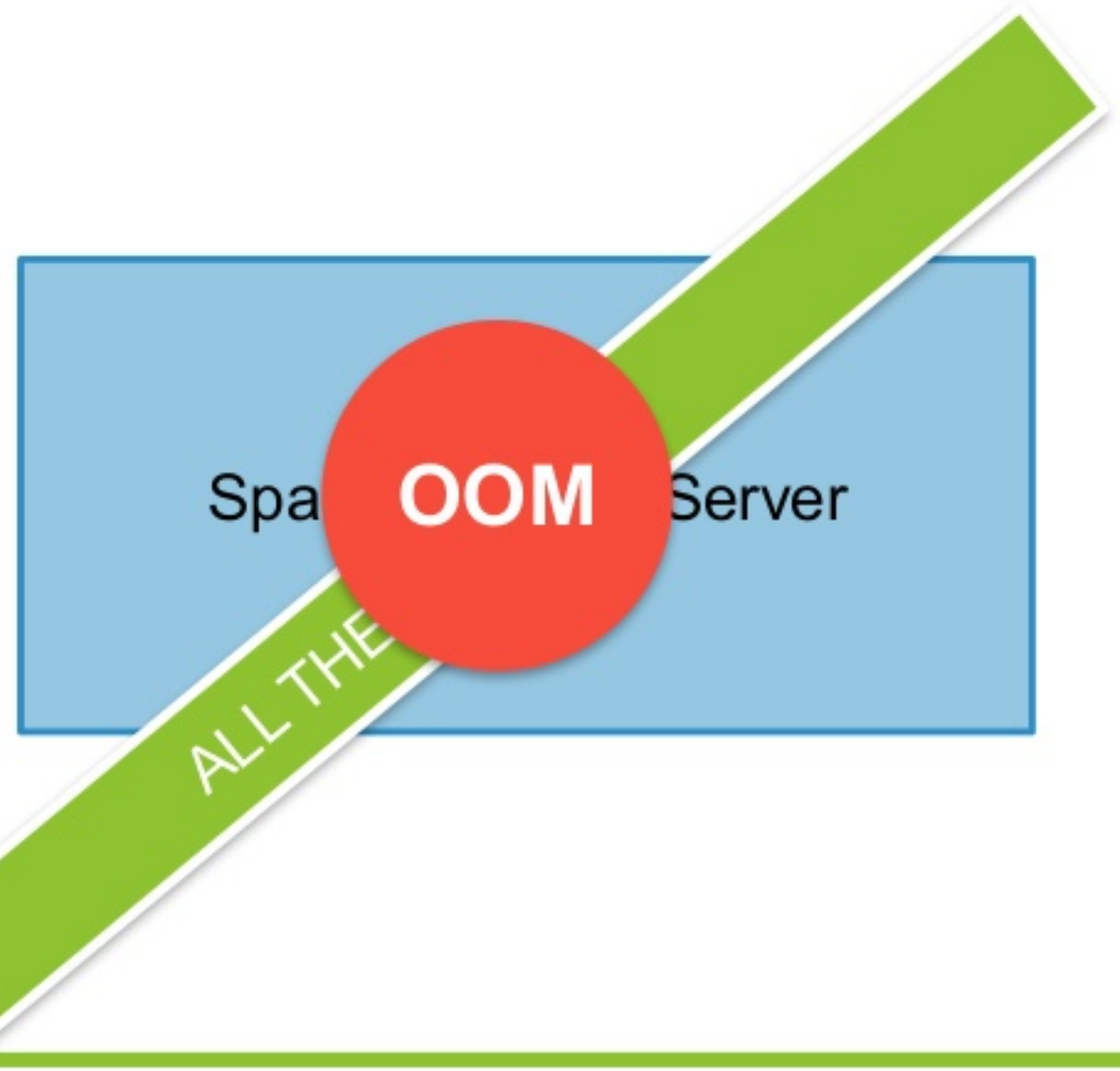
Incremental Collect - Because BI Tools are Mean

SELECT *
FROM TABLE

Spark Sql Thrift Server

ALL THE DATA

Incremental Collect - Because BI Tools are Mean



Incremental Collect - Because BI Tools are Mean

Spark Sql Thrift Server
spark.sql.thriftServer.incrementalCollect=true

Spark Partition 1 | Spark Partition 2 | Spark Partition 3

Incremental Collect - Because BI Tools are Mean

Spark Sql Thrift Server

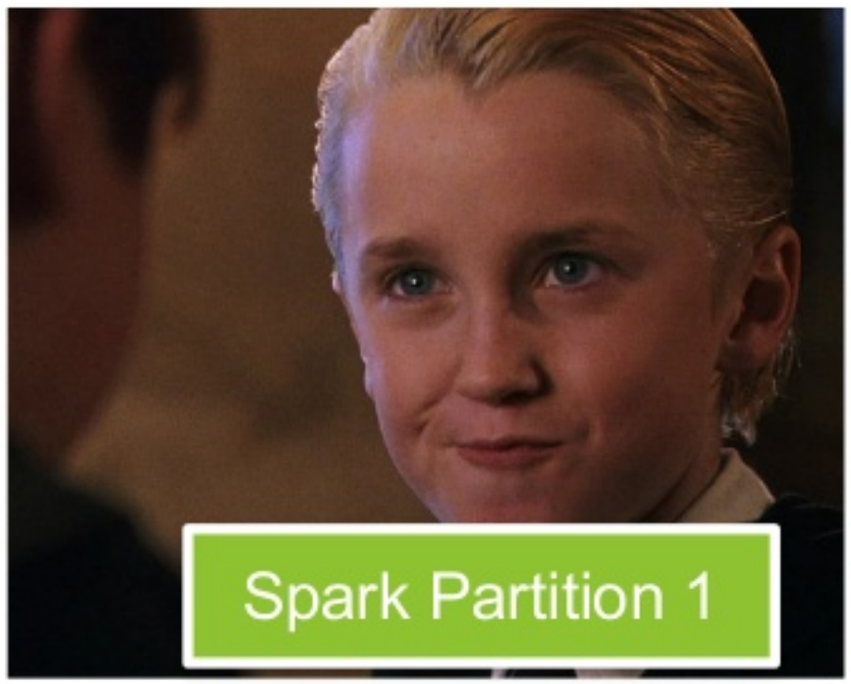
spark.sql.thriftServer.incrementalCollect=true

Spark Partition 1

Spark Partition 2

Spark Partition 3

Incremental Collect - Because BI Tools are Mean



Spark Sql Thrift Server

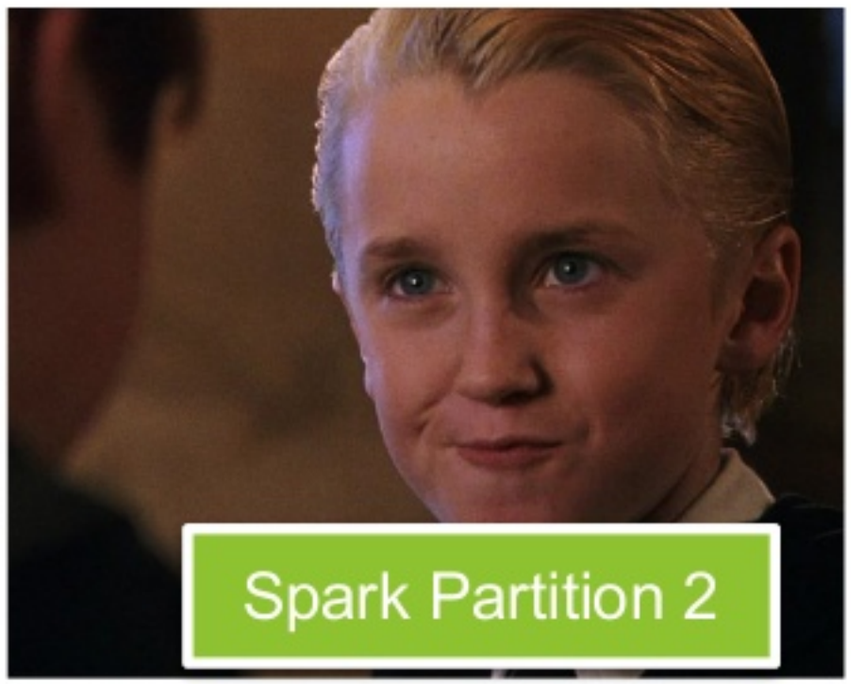
spark.sql.thriftServer.incrementalCollect=true

Spark Partition 2

ALL THE DATA

Spark Partition 3

Incremental Collect - Because BI Tools are Mean



Spark Sql Thrift Server

spark.sql.thriftServer.incrementalCollect=true

Spark Partition 3

ALL THE DATA

Getting things done with SQL

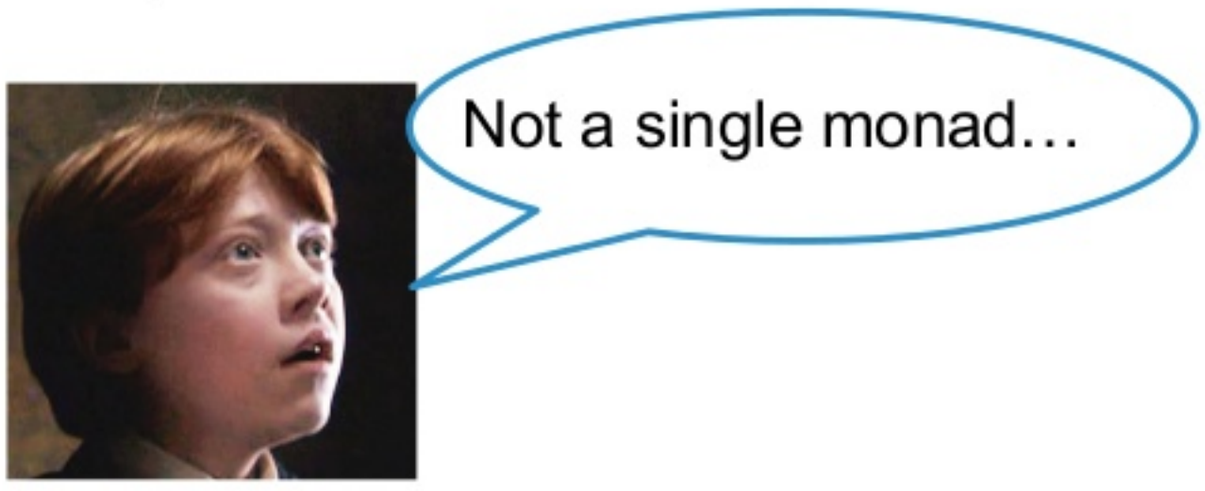
- Registering Sources
- Writing to Tables
- Examining Query Plans
- Debugging Predicate pushdowns
- Caching Views

Registering Sources using SQL

```
CREATE TEMPORARY VIEW words
      USING format.goes.here
      OPTIONS (
          key "value"
)
```

Registering Sources using SQL

```
CREATE TEMPORARY VIEW words
     USING org.apache.spark.sql.cassandra
     OPTIONS (
     table "tab",
     keyspace "ks")
```

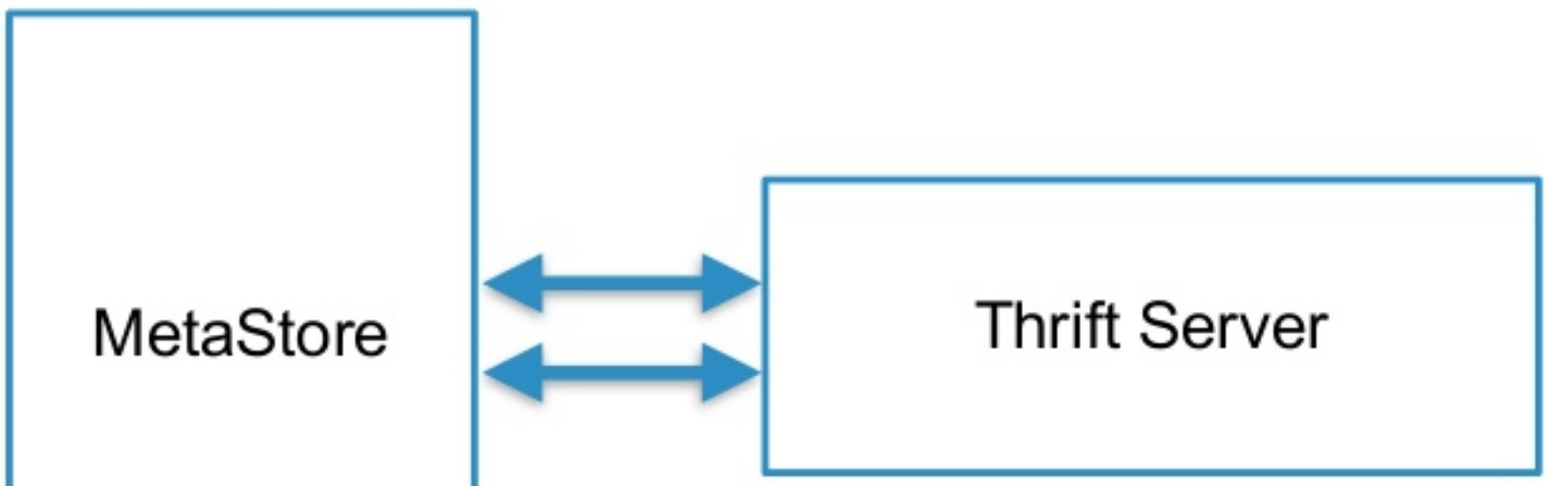


Registering Sources using SQL

```
CREATE TEMPORARY VIEW words
     USING org.apache.spark.sql.cassandra
     OPTIONS (
     table "tab",
     keyspace "ks")
```

CassandraSourceRelation

We Can Still Use a HiveMetaStore

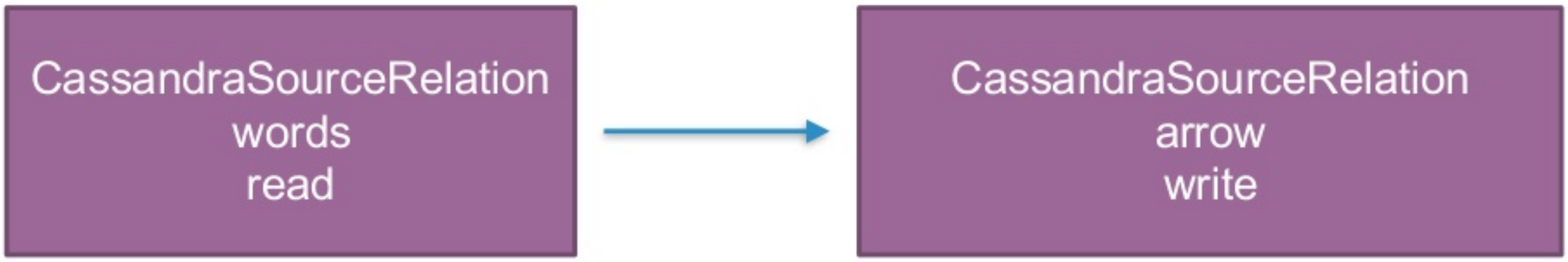


DATASTAX

DSE auto registers C* Tables in a C* based Metastore

Writing DataFrames using SQL

```
INSERT INTO arrow SELECT * FROM words;
```



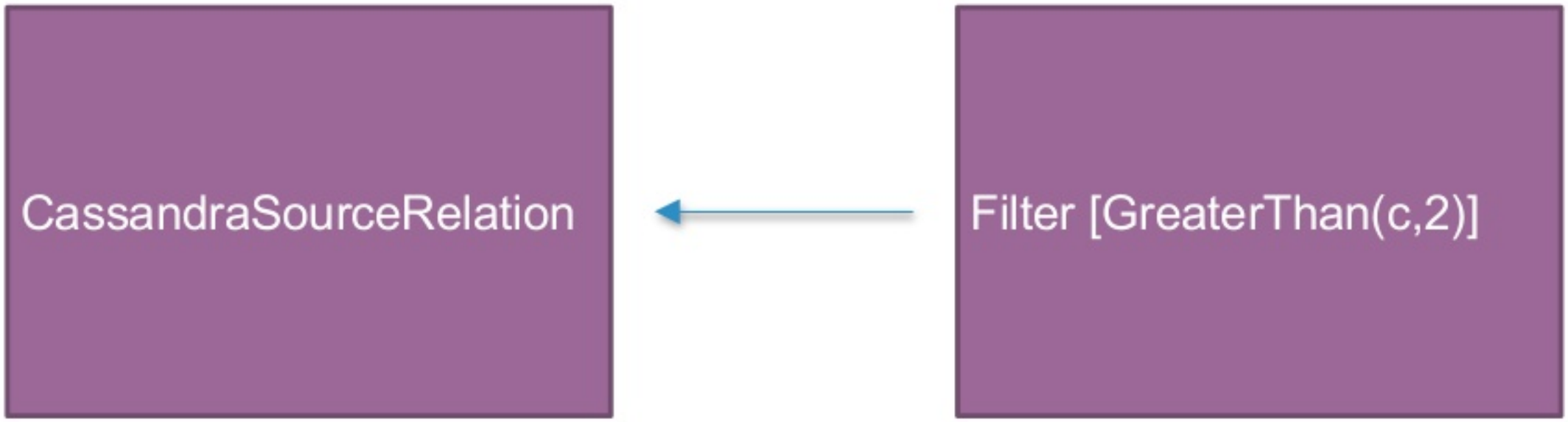
Explain to Analyze Query Plans

```
EXPLAIN SELECT * FROM arrow WHERE C > 2;

Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@6069193a
[k#18,c#19,v#20]
  PushedFilters: [IsNotNull(c), GreaterThan(c,2)],
  ReadSchema: struct<k:int,c:int,v:int>
```

We can analyze the inside
of the Catalyst just
like with Scala/Java/…

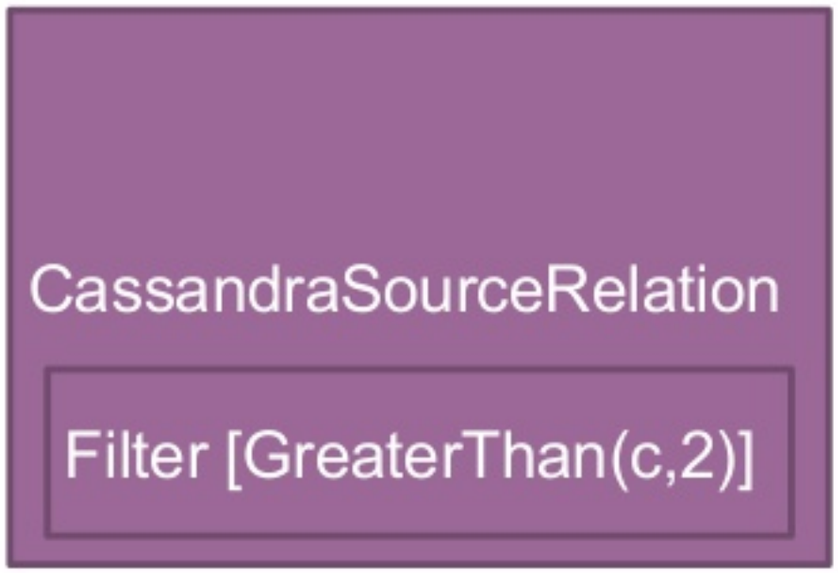
Predicates get Pushed Down Automatically

```
EXPLAIN SELECT * FROM arrow WHERE C > 2;

Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@6069193a
[k#18,c#19,v#20]
  PushedFilters: [IsNotNull(c), GreaterThan(c,2)],
  ReadSchema: struct<k:int,c:int,v:int>
```



Predicates get Pushed Down Automatically

```
EXPLAIN SELECT * FROM arrow WHERE C > 2;

Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@6069193a
[k#18,c#19,v#20]
  PushedFilters: [IsNotNull(c), GreaterThan(c,2)],
  ReadSchema: struct<k:int,c:int,v:int>
```

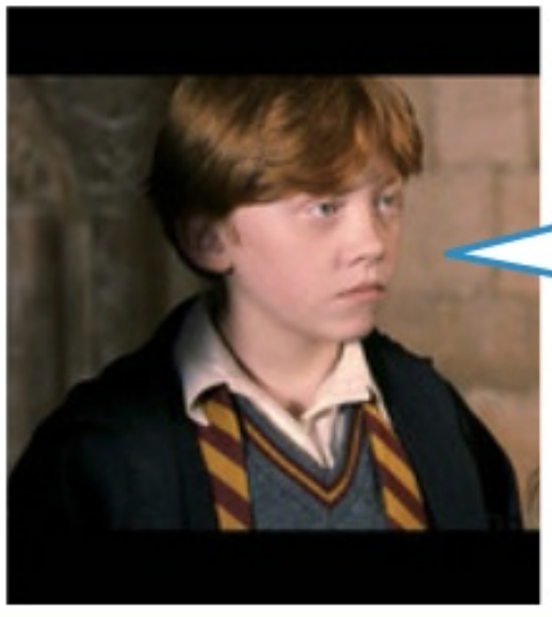


Internal Request to Cassandra: CQL
SELECT * FROM ks.bat WHERE C > 2

Automatic
Pushdowns!

Common Cases where Predicates Don't Push

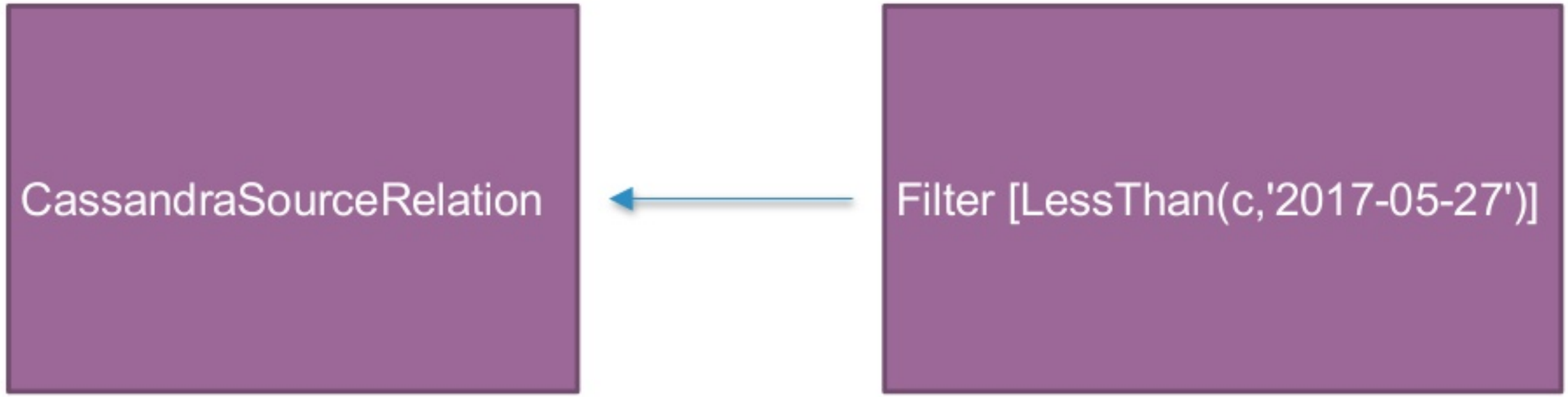
```
SELECT * from troubles WHERE c < '2017-05-27'
*Filter (cast(c#76 as string) < 2017-05-27)
+- *Scan CassandraSourceRelation@53e82b30 [k#75,c#76,v#77]
      PushedFilters: [IsNotNull(c)],
      ReadSchema: struct<k:int,c:date,v:int>
```



Why is my date clustering column not being pushed down.

Common Cases where Predicates Don't Push

```
SELECT * from troubles WHERE c < '2017-05-27'
*Filter (cast(c#76 as string) < 2017-05-27)
+- *Scan CassandraSourceRelation@53e82b30 [k#75,c#76,v#77]
      PushedFilters: [IsNotNull(c)],
      ReadSchema: struct<k:int,c:date,v:int>
```



Common Cases where Predicates Don't Push

```
SELECT * from troubles WHERE c < '2017-05-27'
*Filter (cast(c#76 as string) < 2017-05-27)
+- *Scan CassandraSourceRelation@53e82b30 [k#75,c#76,v#77]
      PushedFilters: [IsNotNull(c)],
      ReadSchema: struct<k:int,c:date,v:int>
```

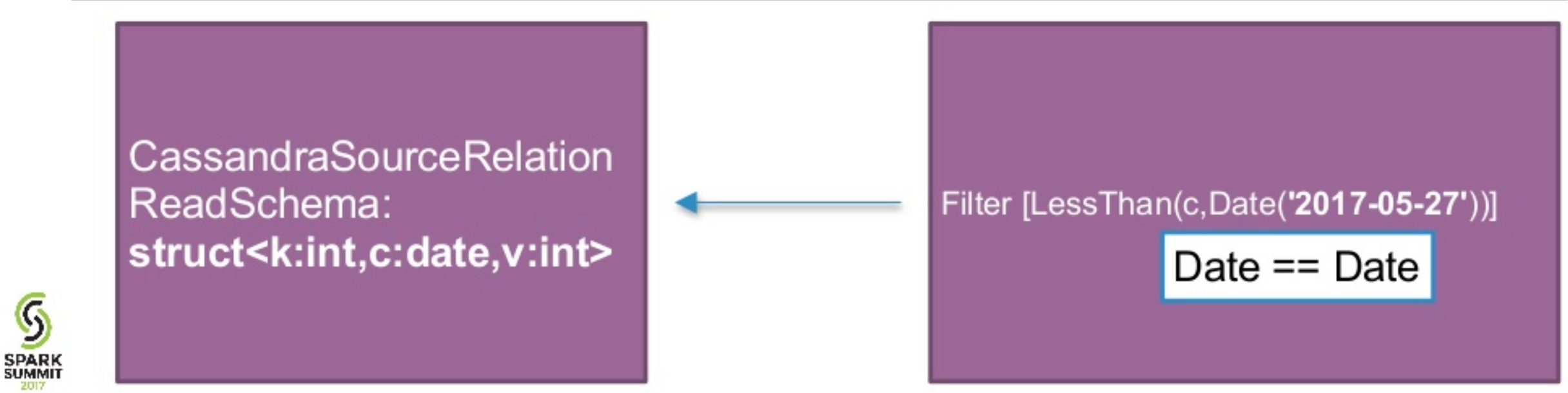
CassandraSourceRelation
ReadSchema:
struct<k:int,c:date,v:int>

Filter [LessThan(c,**'2017-05-27'**)]

Date != String

Make Sure we Cast Correctly

```
EXPLAIN SELECT * from troubles WHERE
  c < cast('2017-05-27' as date);
*Scan C*Relation PushedFilters: [IsNotNull(c), LessThan(c, 2017-05-27)]
```



Make Sure we Cast Correctly

```
EXPLAIN SELECT * from troubles WHERE
  c < cast('2017-05-27' as date);
*Scan C*Relation PushedFilters: [IsNotNull(c), LessThan(c, 2017-05-27)]
```

CassandraSourceRelation
ReadSchema:
struct<k:int,c:date,v:int>

Filter [LessThan(c,Date(**'2017-05-27'**))]

Automatic
Pushdowns!

DSE Search Automatic Pushdowns!

```
EXPLAIN SELECT * from troubles WHERE v < 6;
 *Scan C*Relation PushedFilters: [IsNotNull(v), LessThan(v, 6)]
```

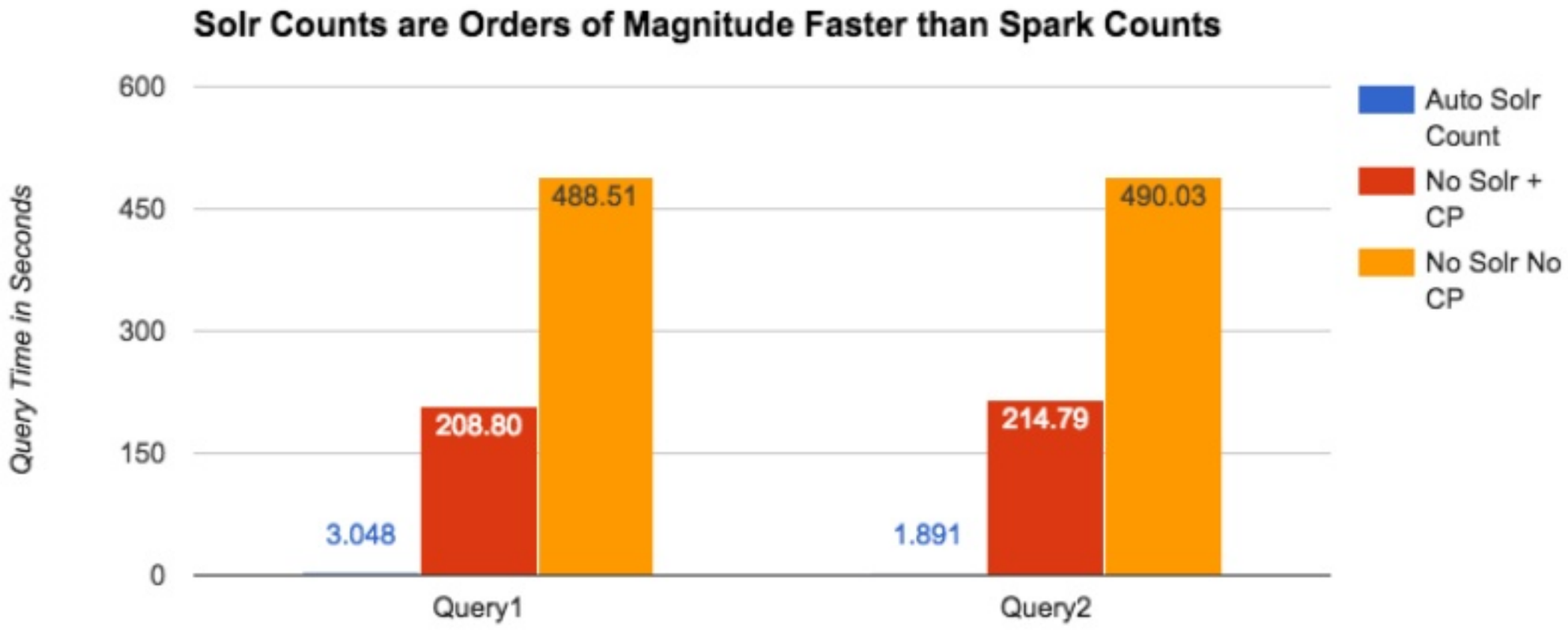
DATASTAX

Solr_Query

CassandraSourceRelation
ReadSchema:
struct<k:int,c:date,v:int>

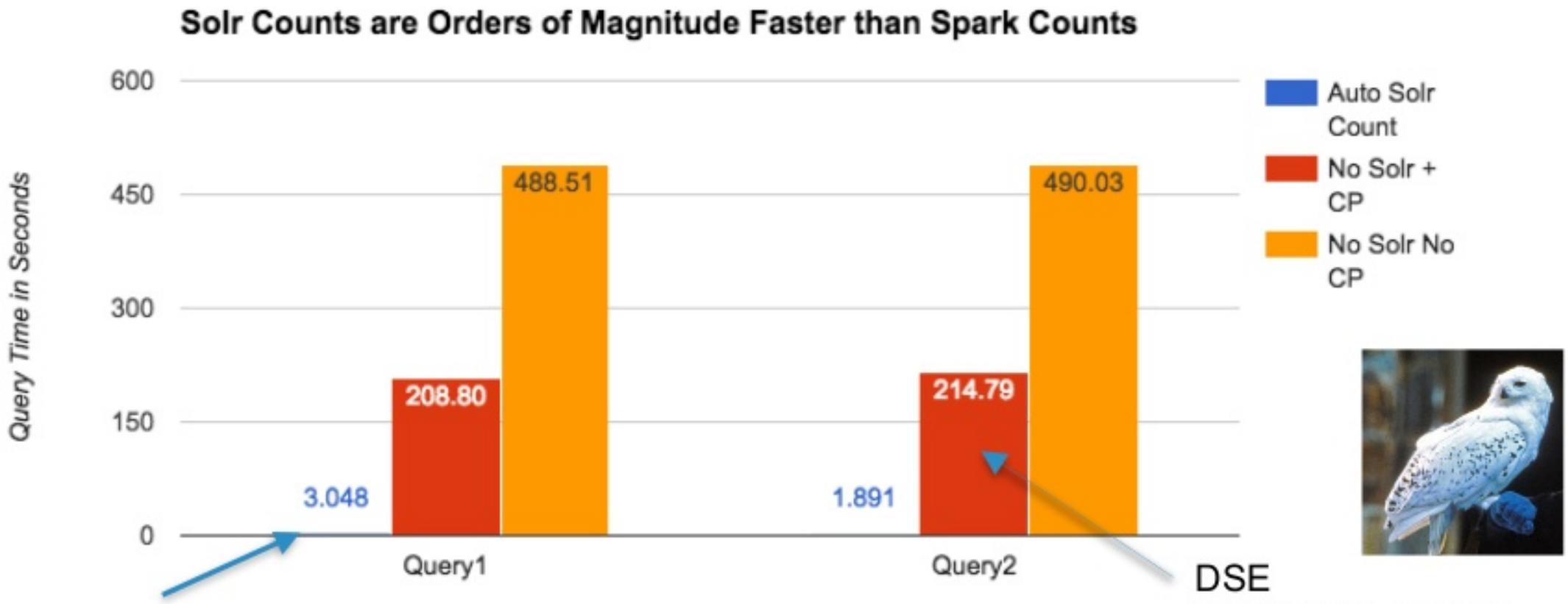
SPARK SUMMIT 2017

DSE Search Automatic Pushdowns!



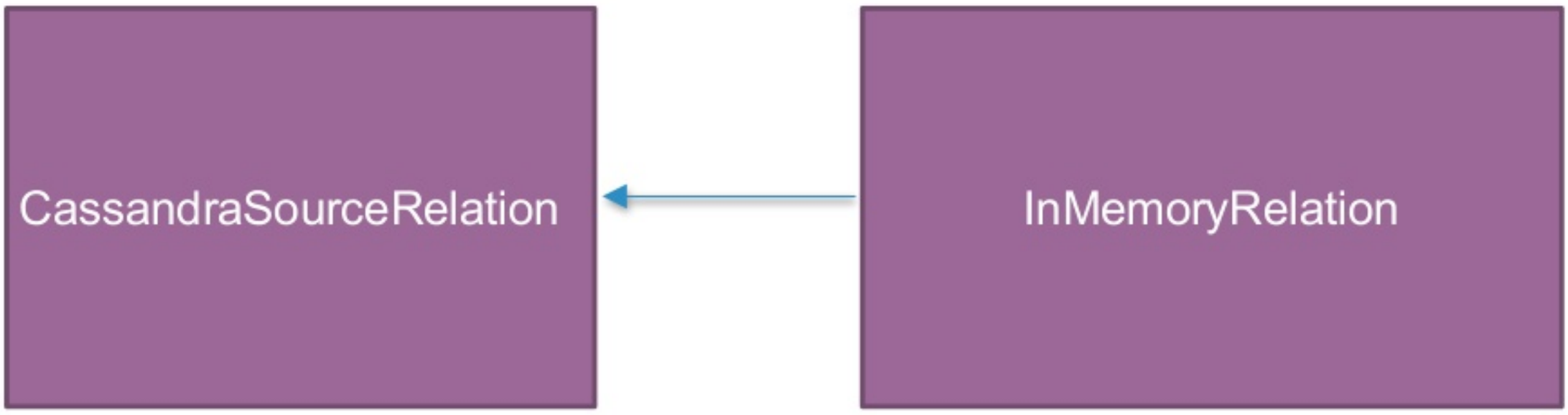
SPARK SUMMIT 2017

DSE Search Automatic Pushdowns!



Cache a whole table

```
CACHE TABLE ks.tab;
explain SELECT * FROM ks.tab;
== Physical Plan ==
InMemoryTableScan [k#0, c#1, v#2]
:  +- InMemoryRelation StorageLevel(disk, memory, deserialized, 1 replicas), `ks`.`tab`
:     :  +- *Scan CassandraSourceRelation
```



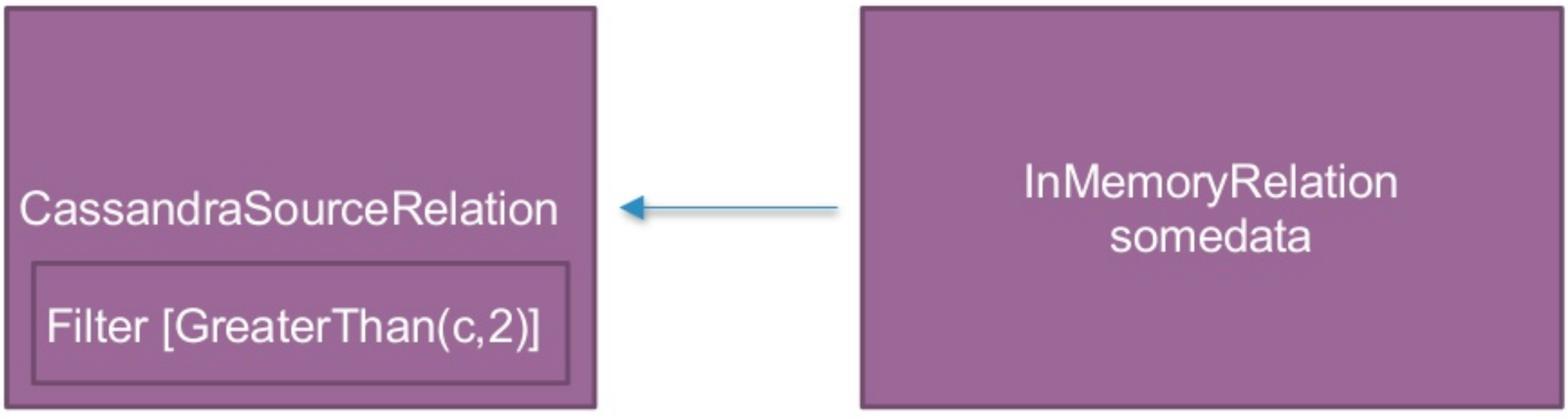
Uncache

```
UNCACHE TABLE ks.tab;
explain SELECT * FROM ks.tab;
== Physical Plan ==
*Scan CassandraSourceRelation
```

CassandraSourceRelation

Cache a fraction of Data

```
CACHE TABLE somedata SELECT * FROM ks.tab WHERE c > 2;
explain SELECT * from somedata;
== Physical Plan ==
InMemoryTableScan
:  +- InMemoryRelation `somedata`
:     :  +- *Scan CassandraSourceRelation PushedFilters: [IsNotNull(c), GreaterThan(c,2)]
```



Let this be a starting point

- https://github.com/datastax/spark-cassandra-connector
- https://github.com/datastax/spark-cassandra-connector/blob/master/doc/14_data_frames.md
- https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-sql-thrift-server.html
- http://docs.datastax.com/en/dse/5.1/dse-dev/datastax_enterprise/spark/sparkSqlThriftServer.html
- https://spark.apache.org/docs/latest/sql-programming-guide.html#distributed-sql-engine
- https://www.datastax.com/dev/blog/dse-5-1-automatic-optimization-of-spark-sql-queries-using-dse-search
- https://www.datastax.com/dev/blog/dse-continuous-paging-tuning-and-support-guide

Thank You.

http://www.russellspitzer.com/
@RussSpitzer

Come chat with us at DataStax Academy:
https://academy.datastax.com/slack