Spark SQL: Relational Data Processing in Spark (SIGMOD 2015)

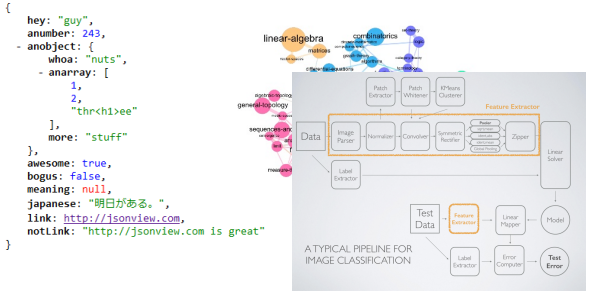
Presented by Ankur Dave

CS294-110, Fall 2015

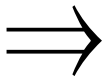
Problem

Imperative

Declarative



Semi-Structured Data
& Advanced Analytics



?

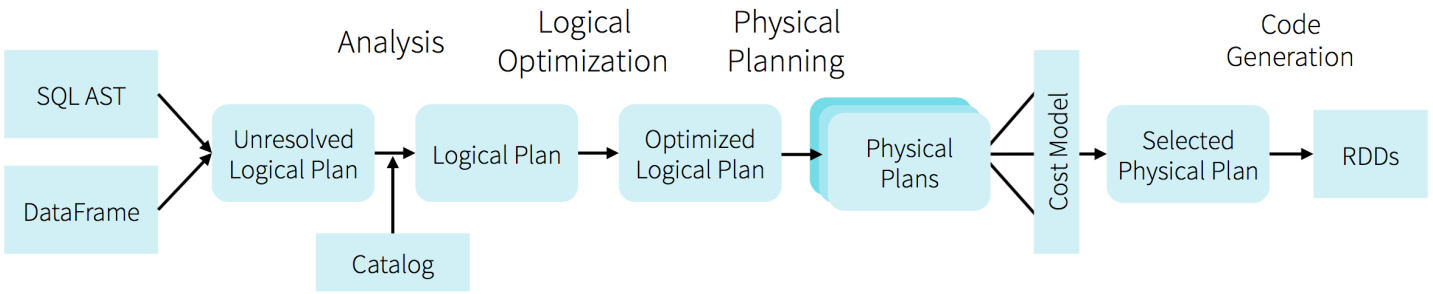
No support in existing systems

Spark SQL

1. DataFrame API

```
sqlCtx.table("people") \
   .groupBy("name") \
   .agg("name", avg("age")) \
   .collect()
```

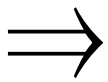
2. Catalyst Optimizer



DataFrames

- Language-integrated declarative API

```
data = sc.textFile(...).split("\t")
data.map(lambda x: (x[0], [int(x[1]), 1])) \
   .reduceByKey(lambda x, y: [x[0] + y[0], x[1] + y[1]]) \
   .map(lambda x: [x[0], x[1][0] / x[1][1]]) \
   .collect()
```



```
sqlCtx.table("people") \
   .groupBy("name") \
   .agg("name", avg("age")) \
   .collect()
```

- Schema inference for semi-structured data
- Allows direct access to JVM objects
  - Unlike other declarative systems
- Supports complex types like vectors
  - Easy to add new types

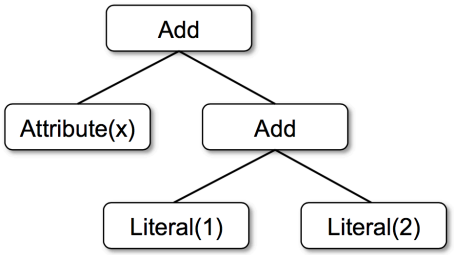
Catalyst Optimizer

- Rules written using Scala pattern matching
  - Allows writing complex rules
  - Example: join elimination

```
// Push down filter through EXCE
case Filter(condition, e @ Excep
  val (deterministic, nondetermi
  val rewrites = buildRewrites(e
  Filter(nondeterministic,
    Except(
```

- Exploits structure of data source
  - Example: predicate pushdown for Parquet
  - Easy to add new data sources (e.g., Succinct)

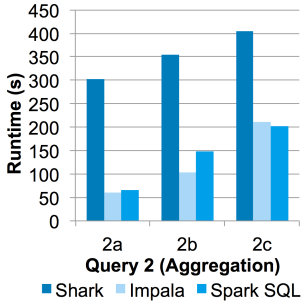
- Code generation for expression evaluation



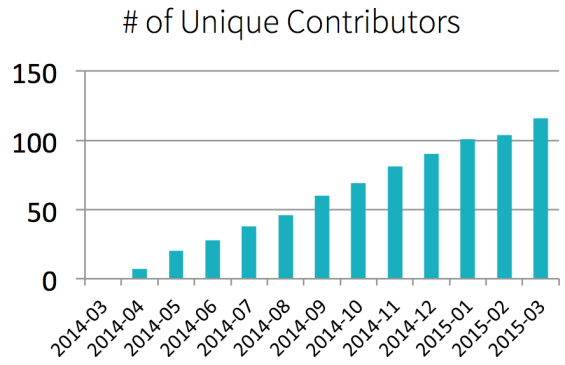
- Cost-based join algorithm selection (shuffle vs. BitTorrent broadcast)

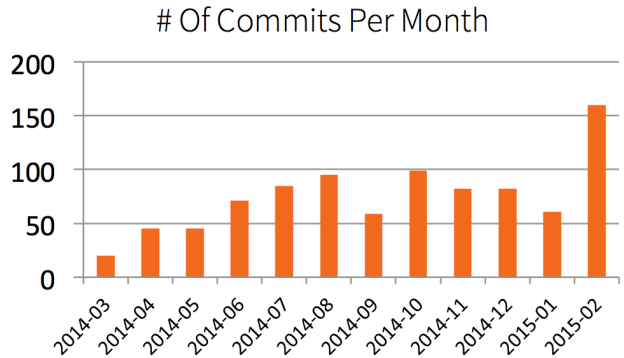
Evaluation

- Gains over Spark and Shark due to code generation
  - Optimizer less mature than Impala's

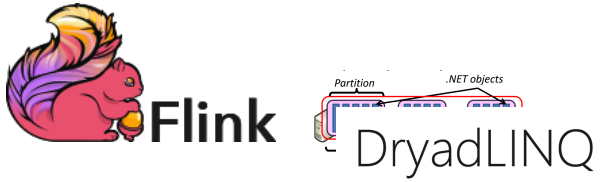


- More important: ease of use and extensibility





Why Spark SQL?

	Easy to Program?	Advanced Analytics?
	No	Yes
	Yes	No
	Yes	Yes
	?	?

Lasting Impact?

- Extensible optimizer

```
// Push down filter through EXCEPT
case Filter(condition, e @ Except(l
  val (deterministic, nondeterminis
  val rewrites = buildRewrites(e)
  Filter(nondeterministic,
    Except(
```

- Language integration

```
sqlCtx.table("people") \
   .groupBy("name") \
   .agg("name", avg("age")) \
   .collect()
```