Apache Spark DataFrames

Markus Dale

2015

Spark Ecosystem

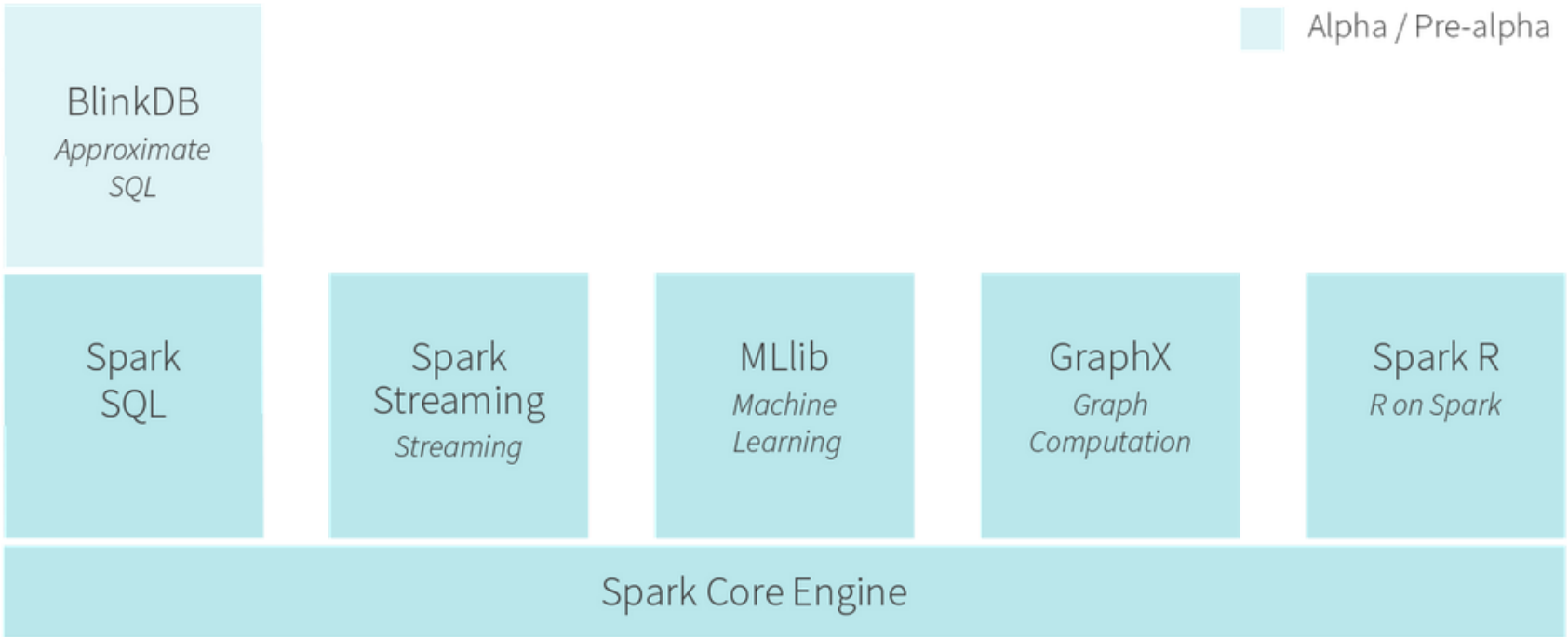

Figure : Databricks Spark 1.4.0 Ecosystem (2015)

Spark SQL

- Structured/semi-structured data on Spark
- Can write SQL-like queries or
- DataFrame DSL language
- Michael Armbrust (Databricks Spark SQL lead):
  - Write less code
  - Read less data
  - Let [Catalyst query] optimizer do the hard work

Emails per user - RDD

```
val mailRecordsAvroRdd =
  sc.newAPIHadoopFile("enron.avro",
        classOf[AvroKeyInputFormat[MailRecord]],
        classOf[AvroKey[MailRecord]],
        classOf[NullWritable], hadoopConf)
val recordsRdd = mailRecordsAvroRdd.map {
  case (avroKey, _) => avroKey.datum()
}
val emailsPerUserRdd =
    recordsRdd.flatMap(mailRecord => {
      val userNameOpt =
        mailRecord.getMailFieldOpt("UserName")
    if (userNameOpt.isDefined) Some(userNameOpt.get, 1)
      else None
    }).reduceByKey(_ + _).
    sortBy(((t: (String, Int)) => t._2),
         ascending = false)
```

Emails per user - DataFrame

```
import org.apache.spark.sql.functions.udf

val recordsDf = sqlContext.avroFile("enron.avro")

val getUserUdf = udf((mailFields: Map[String, String])
                => mailFields("UserName"))

import sqlContext.implicits._
val recordsWithUserDf =
  recordsDf.withColumn("user",
       getUserUdf($"mailFields"))
recordsWithUserDf.groupBy("user").
  count().
  orderBy($"count".desc)
```

Spark SQL in Context

- Complete re-write/superset of Shark announced April 2014
- Not Hive on Spark
- Leverages Spark Core infrastructure/RDD abstractions
- Separate library (in addition to Spark Core): spark-sql, spark-hive

DataFrame

- Introduced in Spark 1.3 March 2015 (presentation uses 1.4.0)
- Replacement/evolution of SchemaRDD
- Inspired by data frames in Python Data Analysis (pandas) and R
- Distributed collection of Row objects (with known schema/columns)
- Abstractions for selecting, filtering, aggregation

Catalyst Query Optimizer Pipeline

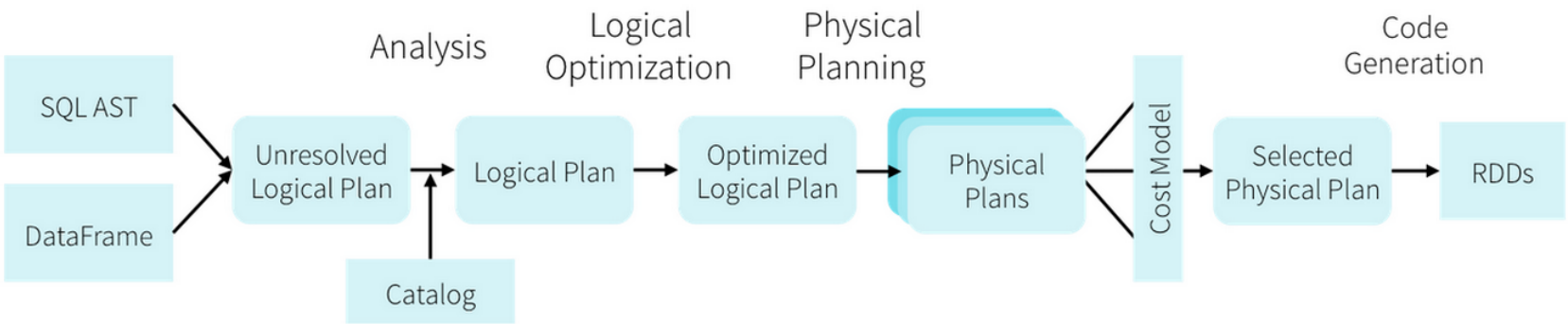

Figure : Catalyst Query Optimizer Pipeline Armbrust (2015a)

DataFrame Speed Up - Catalyst Query Optimizer

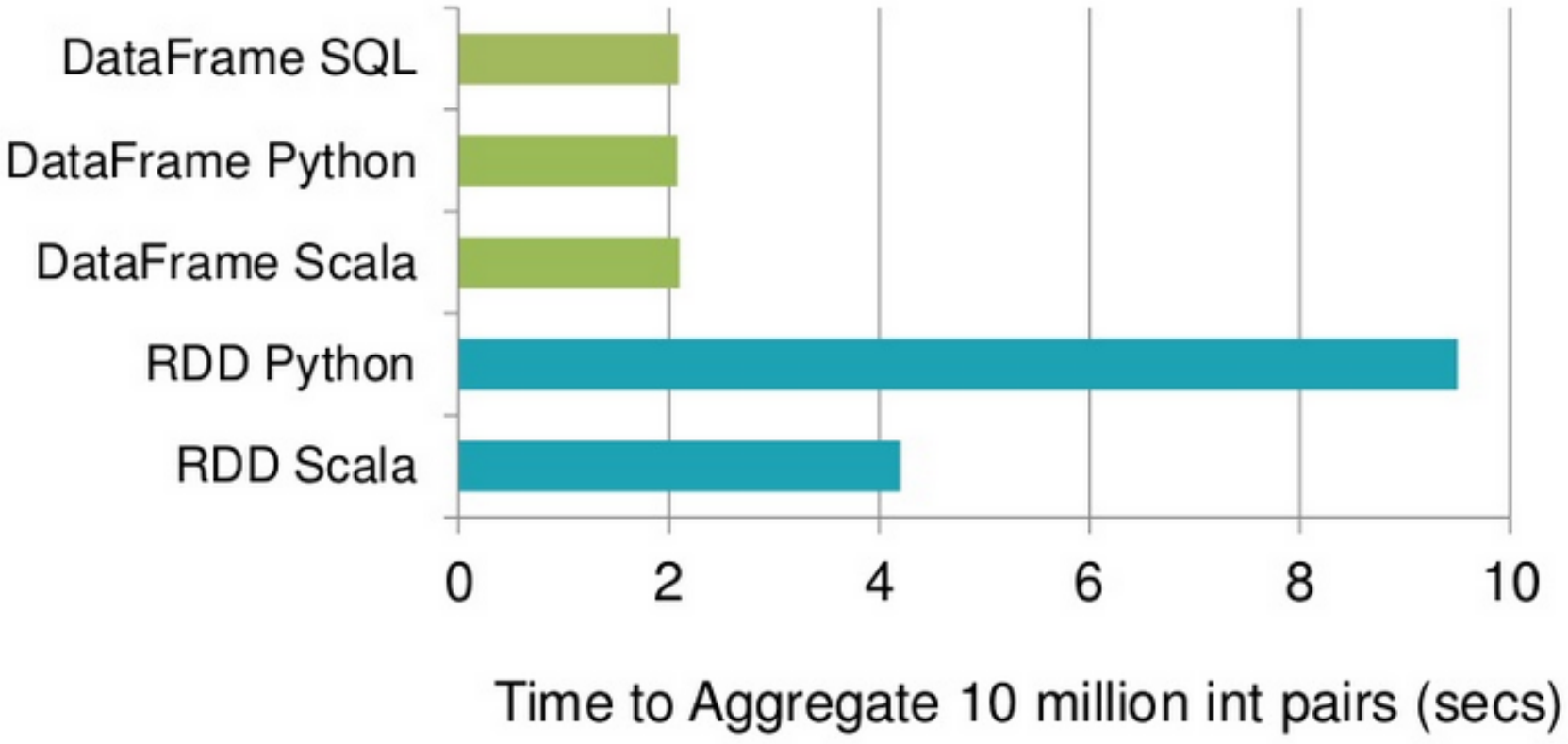

Figure : DataFrame Runtimes Armbrust (2015b)

Spark SQL Data Sources

Figure : Internal and external data sources Armbrust (2015a)

Spark Packages

- Aggregator site for third party Spark packages (http://spark-packages.org)
- spark-avro
- spark-redshift
- couchbase-spark-connector
- 10 more entries (as of June 21, 2015)

Apache Parquet

- Columnar storage format - store data by chunks of columns rather than rows
- Support complex nesting using algorithms from (Google Dremel Melnik et al. 2010).
- See (Apache Parquet docs Parquet 2014)

Parquet File Structure

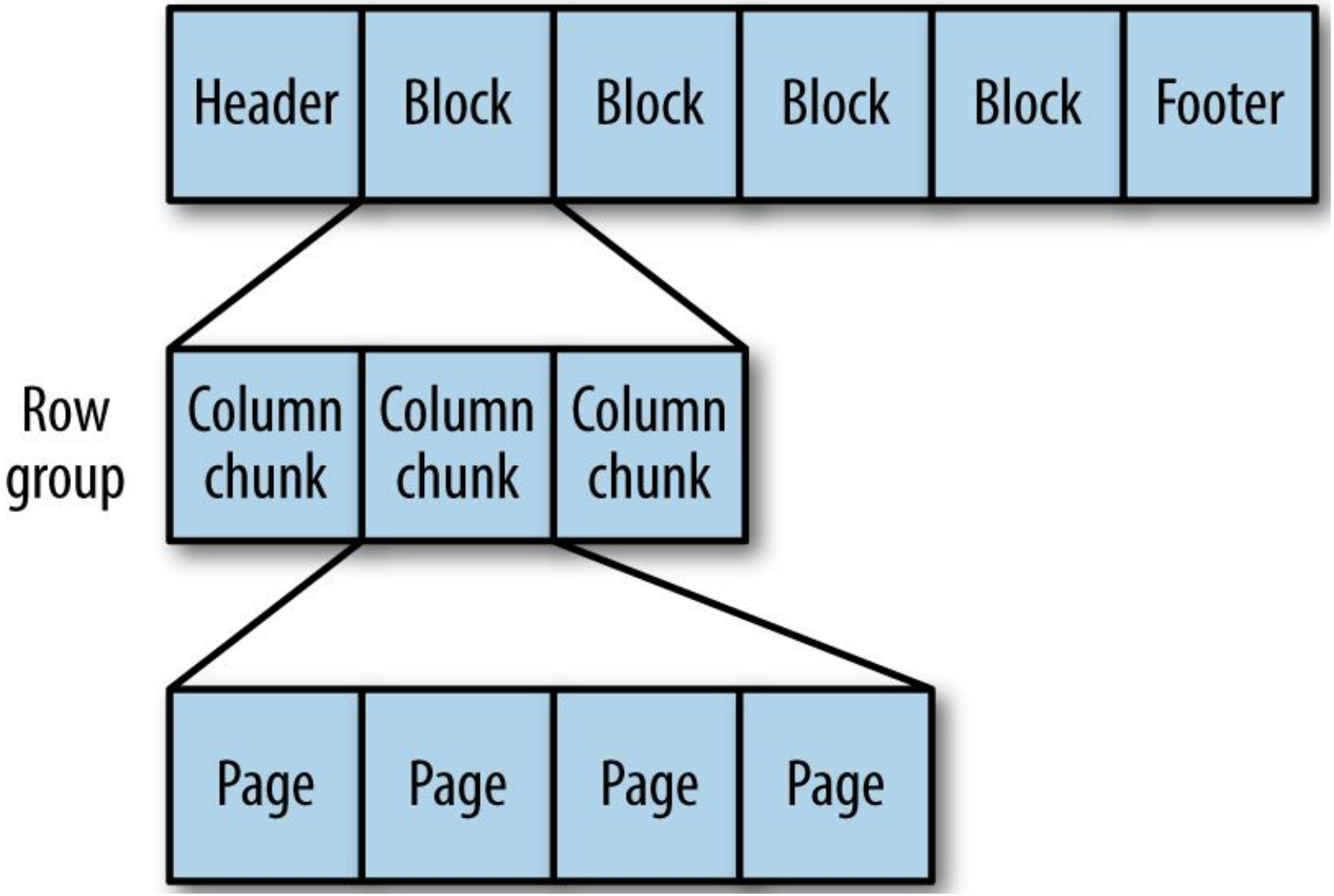

Figure : Parquet File Structure White (2015)

DataFrameReader (1.4)

```
val emails =
  sqlContext.read.format("parquet").load("enron.parquet")
//.read.parquet/json/jdbc

val rolesDf = sqlContext.read.
   format("com.databricks.spark.csv").
   option("header", "true").
   load("roles.csv")
```

DataFrameWriter (1.4)

```
emailsWithYearDf.write.format("parquet").
  partitionBy("year").
  save("/opt/rpm1/enron/parquet/out")

//year=0001  year=1986 ... year=2044
//part-r-00001.gz.parquet in each
```

References I

Armbrust, Michael. 2015a. "What's New for Spark SQL in Spark 1.3." https://databricks.com/blog/2015/03/24/spark-sql-graduates-from-alpha-in-spark-1-3.html.

———. 2015b. "Beyond SQL: Speeding up Spark with DataFrames." http://www.slideshare.net/databricks/spark-sqlsse2015public.

Ecosystem. 2015. "Databricks Spark Ecosystem." https://databricks.com/spark/about.

Melnik, Sergey, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis. 2010. "Dremel - Interactive Analysis of Web-Scale Datasets." In *Proc. of the 36th Int'l Conf on Very Large Data Bases*, 330–339. http://research.google.com/pubs/pub36632.html.

Parquet. 2014. "Apache Parquet Documentation." https://parquet.incubator.apache.org/documentation/latest/.

References II

White, Tom. 2015. *Hadoop The Definitive Guide*. 4th ed. O'Reilly.