cloudera

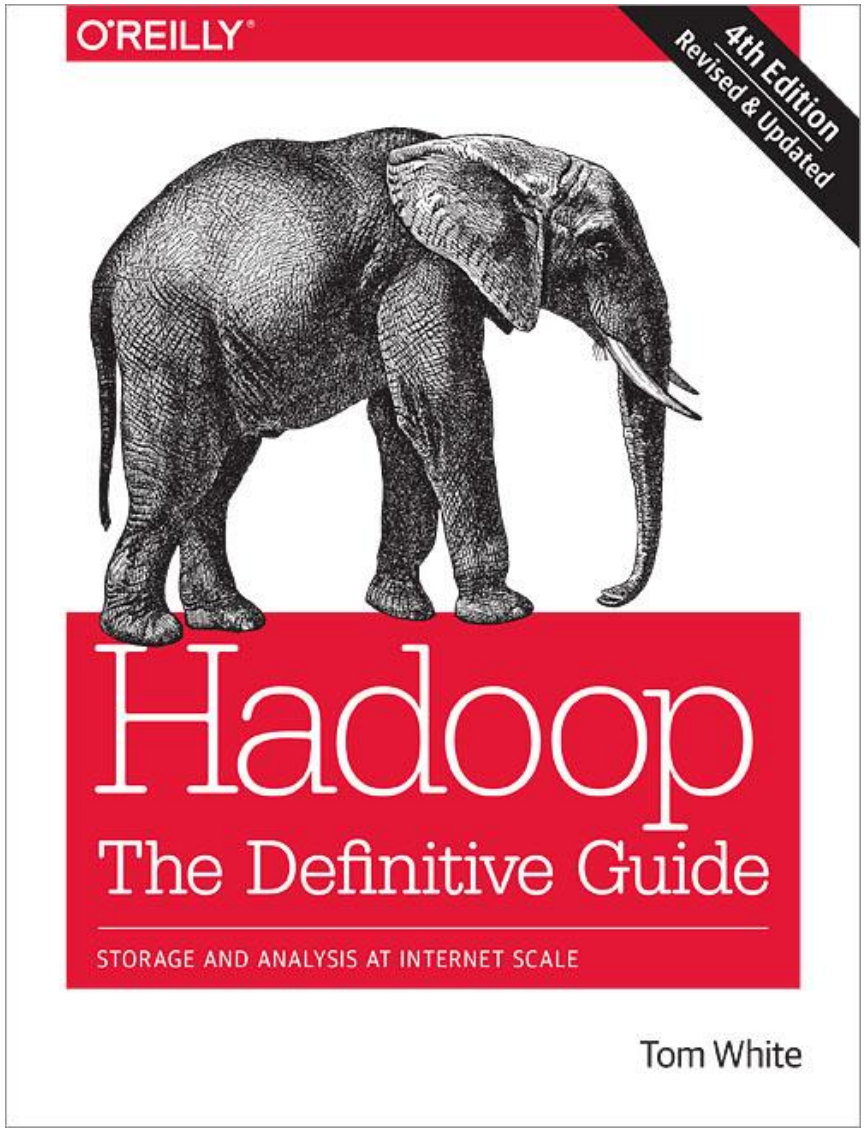
Machine Learning and Data Analytics at Cloudera

29 April 2016, CERN openlab

Tom White | @tom_e_white

About Me

- Data Science Team at Cloudera
- Apache Hadoop Committer, PMC Member, Apache Member
- Author of “Hadoop: The Definitive Guide”



cloudera

What is Data Science?

cloudera

Josh Wills
@josh_wills

Following

Data Scientist (n.): Person who is better at statistics than any software engineer and better at software engineering than any statistician.

RETWEETS
1,379

LIKES
852

5:55 p.m. - 3 May 2012

50 years of Data Science

David Donoho

Sept. 18, 2015
Version 1.00

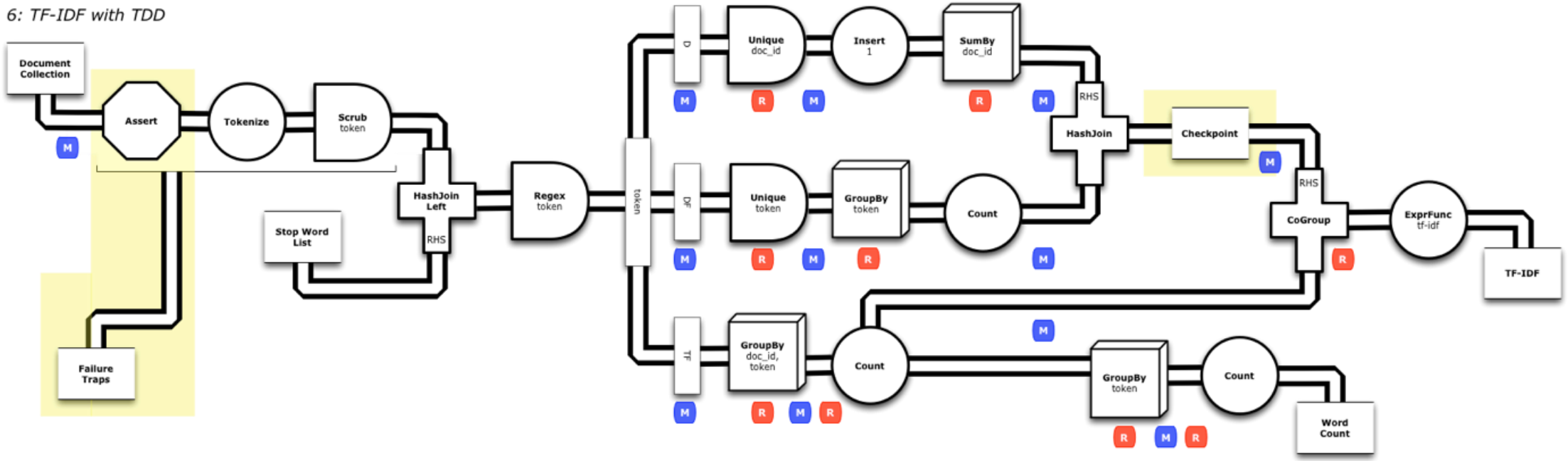
Abstract

More than 50 years ago, John Tukey called for a reformation of academic statistics. In 'The Future of Data Analysis', he pointed to the existence of an as-yet unrecognized *science*, whose subject of interest was learning from data, or 'data analysis'. Ten to twenty years ago, John Chambers, Bill Cleveland and Leo Breiman independently once again urged academic statistics to expand its boundaries beyond the classical domain of theoretical statistics; Chambers called for more emphasis on data preparation and presentation rather than statistical modeling; and Breiman called for emphasis on prediction rather than inference. Cleveland even suggested the catchy name "Data Science" for his envisioned field.

A recent and growing phenomenon is the emergence of "Data Science" programs at major universities, including UC Berkeley, NYU, MIT, and most recently the Univ. of Michigan, which on September 8, 2015 announced a $100M "Data Science Initiative" that will hire 35 new faculty. Teaching in these new programs has significant overlap in curricular subject matter with traditional statistics courses; in general, though, the new initiatives steer away from close involvement with academic statistics departments.

This paper reviews some ingredients of the current "Data Science moment", including recent commentary about data science in the popular media, and about how/whether Data Science is

Data Science is the Full Pipeline



Big Data

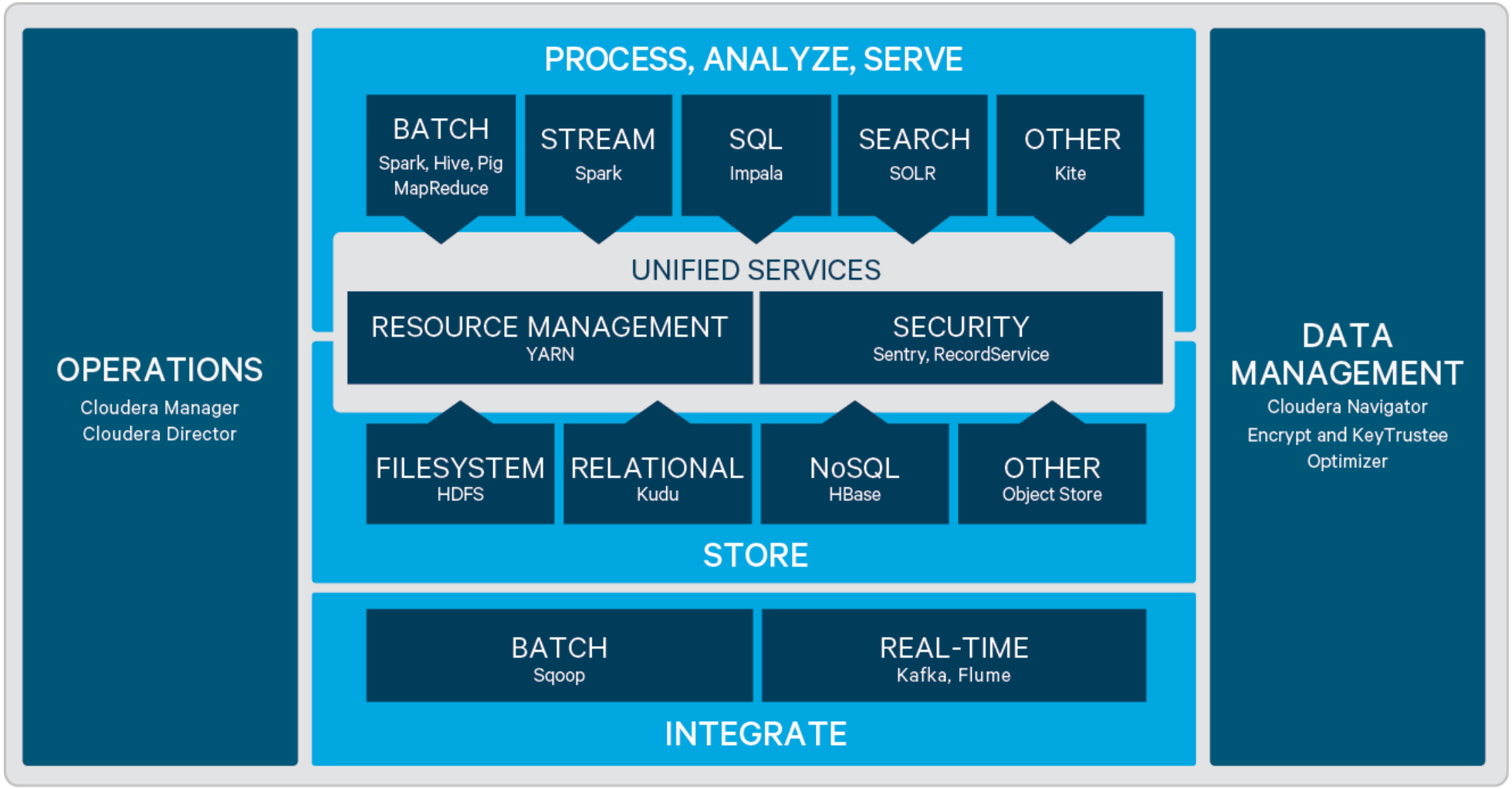
cloudera

Why Hadoop?

- High volume, low-cost shared storage
- Run compute local to the data
- Scale out, not just scale up
- Fault tolerant
- Multiple applications on a common enterprise data hub

cloudera

Cloudera's Hadoop Platform

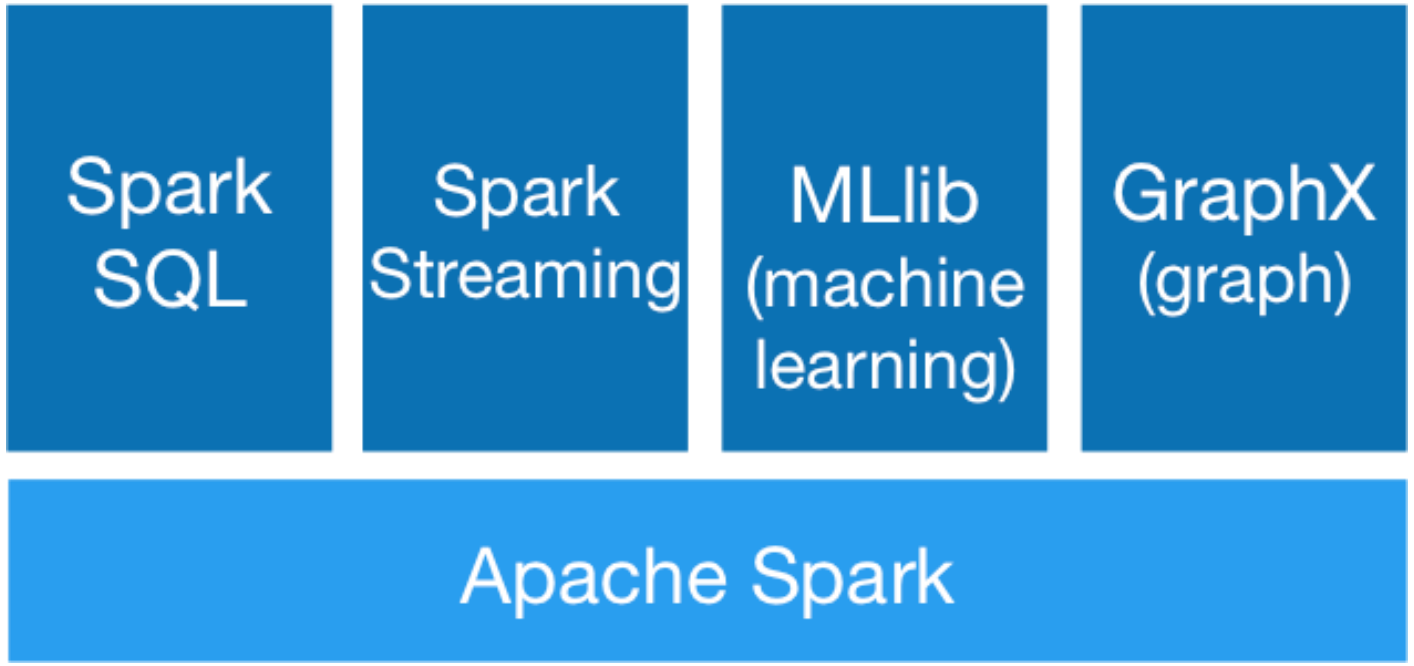


cloudera

Spark: the next-gen generalized compute framework

- Distributed, fault-tolerant data flow processing engine
- Replacing MapReduce in many workflows and systems
- Diverse ecosystem of tools
  - Streaming data processing (Spark Streaming)
  - SQL (Spark SQL)
  - Machine learning (MLLib)
- Varying levels of API / driver support for Python and R

Spark architecture



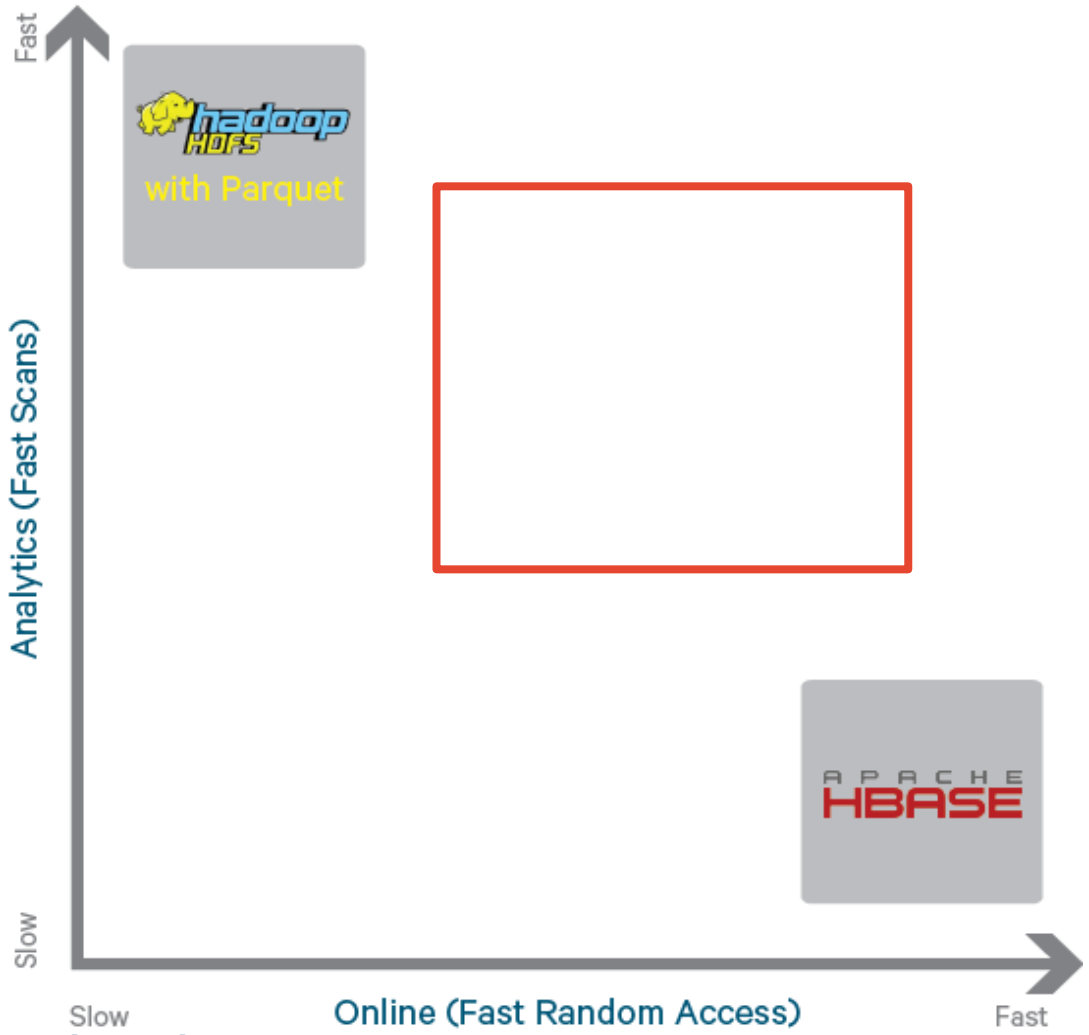
http://spark.apache.org

cloudera

Real-Time Analytics

cloudera

Current Storage Landscape in Hadoop



HDFS excels at:

- Efficiently scanning large amounts of data
- Accumulating data with high throughput

HBase excels at:

- Efficiently finding and writing individual rows
- Making data mutable

Gaps exist when these properties are needed *simultaneously*

cloudera

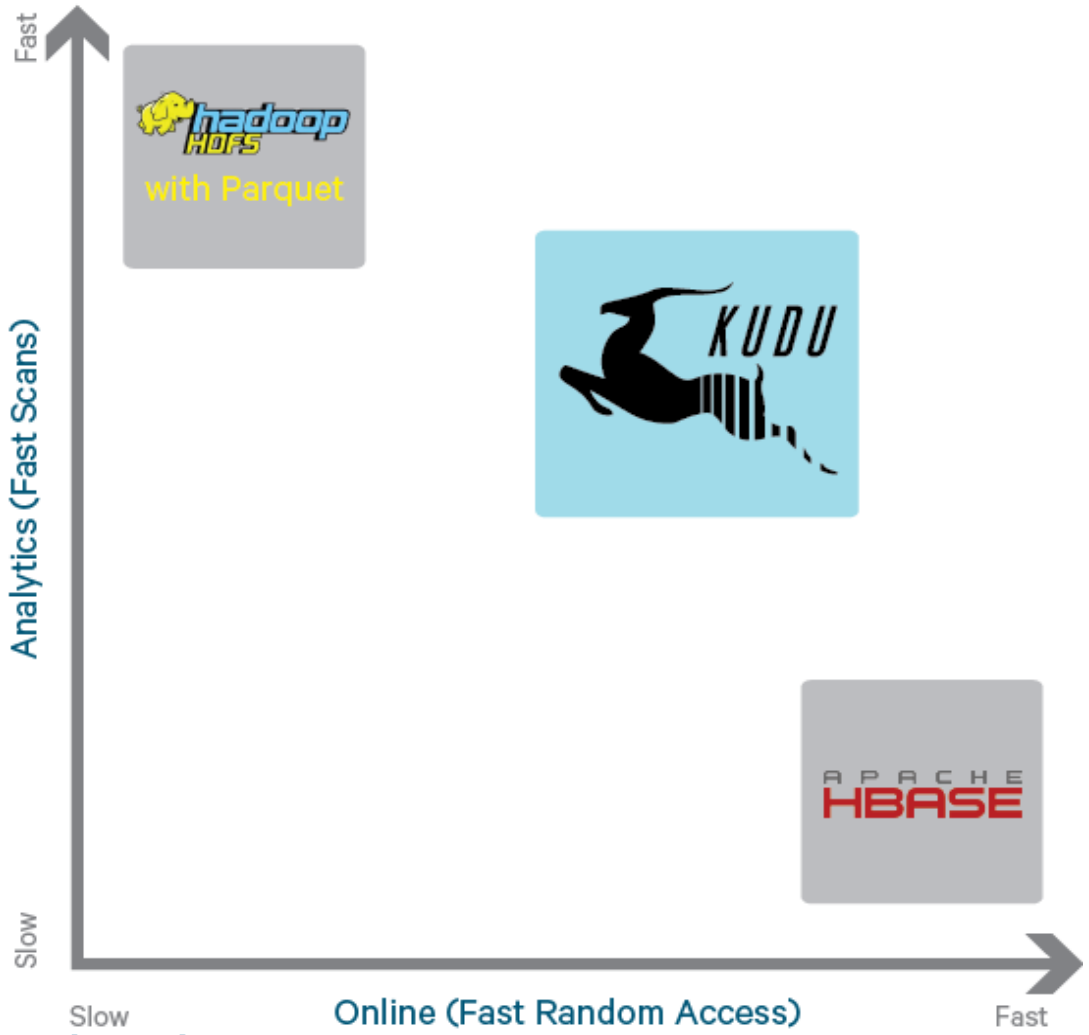
Changing Hardware landscape

- **Spinning disk** -> **solid state storage**
  - **NAND flash**: Up to 450k read 250k write iops, about 2GB/sec read and 1.5GB/sec write throughput, at a price of less than $3/GB and dropping
  - **3D XPoint memory** (1000x faster than NAND, cheaper than RAM)

- **RAM** is cheaper and more abundant:
  - 64->128->256GB over last few years
- ***Takeaway 1***: The **next bottleneck is CPU**, and current storage systems weren't designed with CPU efficiency in mind.
- ***Takeaway 2:*** Column stores are feasible for random access

cloudera

Kudu Design Goals



- **High throughput** for big scans (columnar storage and replication)

  Goal: Within 2x of Parquet

- **Low-latency** for short accesses (primary key indexes and quorum replication)

  Goal: 1ms read/write on SSD

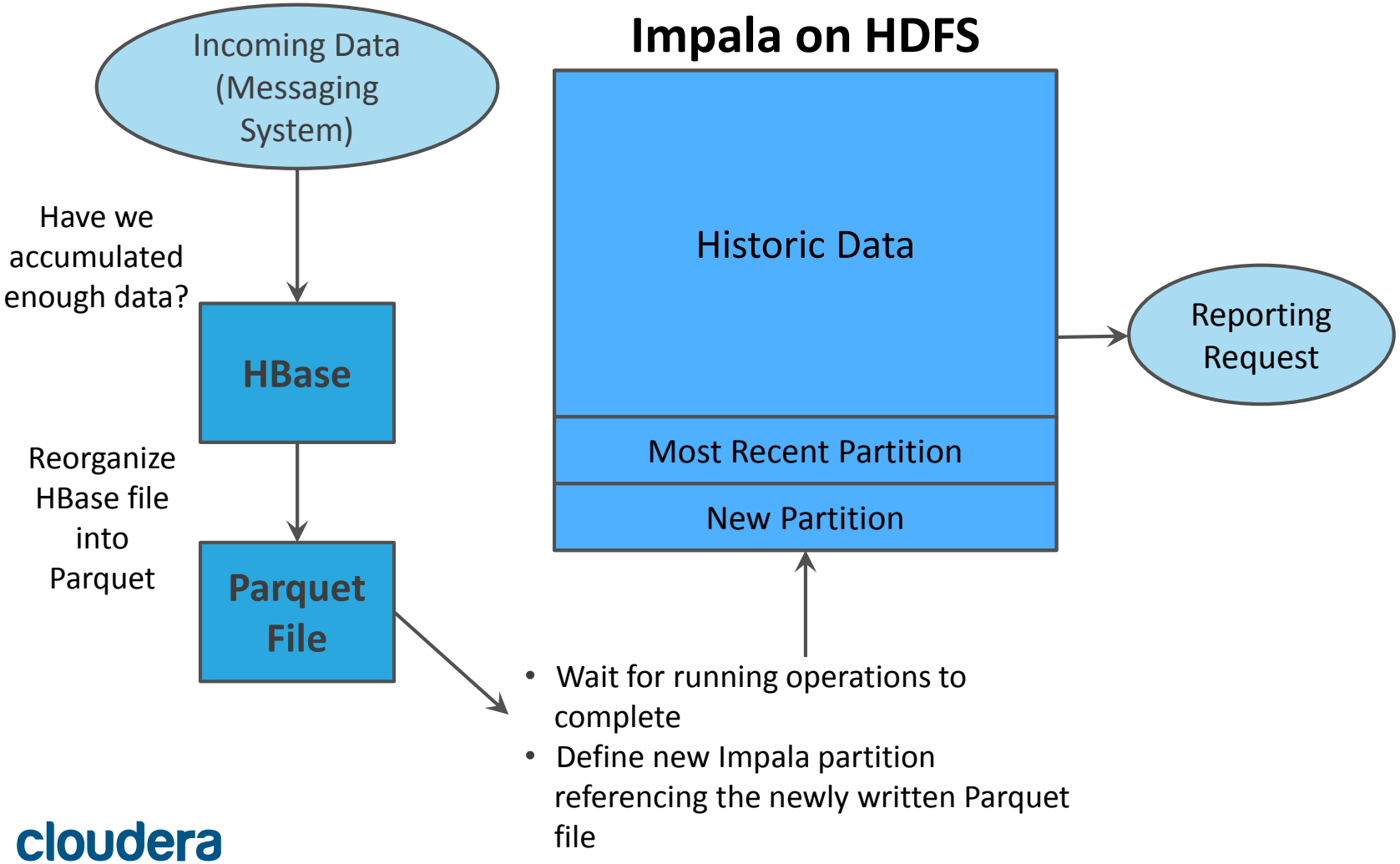
- **Database-like** semantics (initially single-row ACID)

- **Relational data model**
  - SQL query
  - "NoSQL" style scan/insert/update (Java client)

cloudera

Real-Time Analytics in Hadoop Today

Fraud Detection in the Real World = Storage Complexity

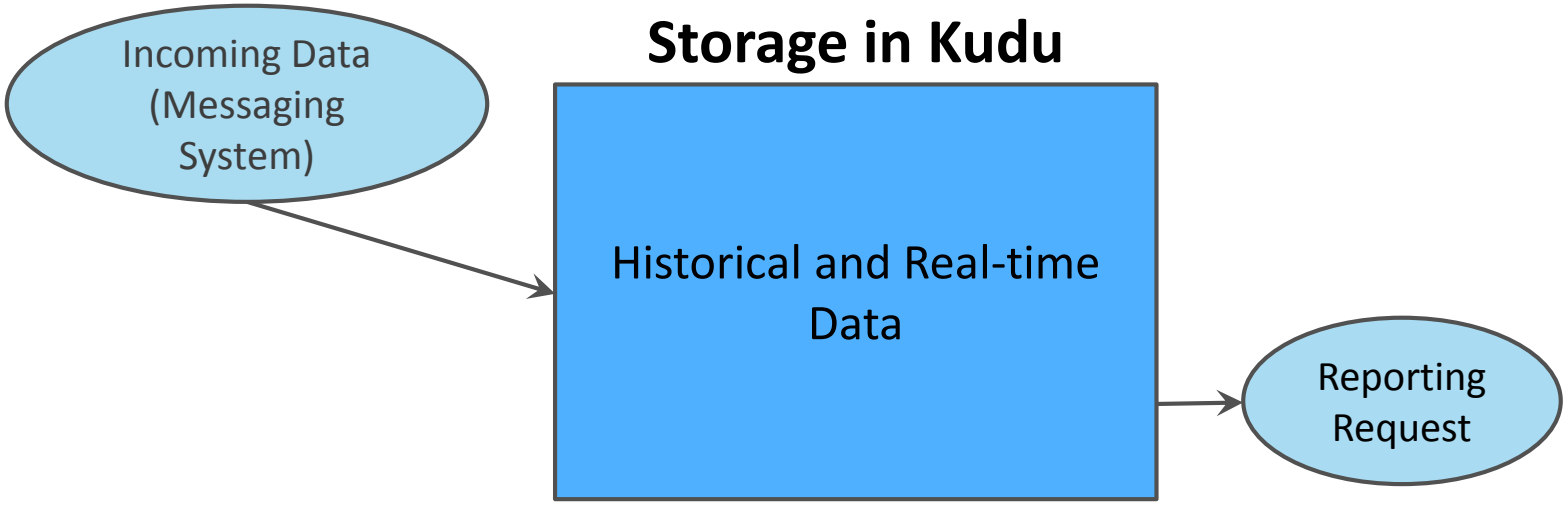


Considerations:

- How do I handle failure during this process?
- How often do I reorganize data streaming in into a format appropriate for reporting?
- When reporting, how do I see data that has not yet been reorganized?
- How do I ensure that important jobs aren't interrupted by maintenance?

cloudera

Real-Time Analytics in Hadoop with Kudu



Improvements:

- **One system** to operate
- **No cron jobs or background processes**
- **Handle late arrivals or data corrections with ease**
- **New data available immediately for analytics or operations**

cloudera

Tools for Data Science

cloudera

Data Science Use Cases

Exploratory Analytics
(understanding data)

Operational Analytics
(improving products for customers)

- **User:** Data scientists and analysts
- **Data:** New and changing; often sampled
- **Environment:** Local machine, dev cluster
- **Tools:** R, Python, SAS, SQL; notebooks; data wrangling/discovery, …
- **Goal:** Understand data, develop and test hypotheses, share results

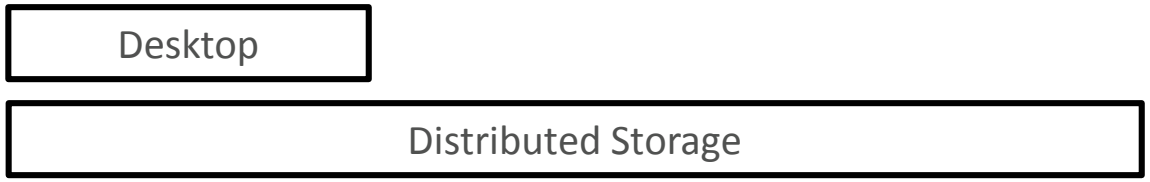
- **User:** Application developers
- **Data:** Known data; full scale
- **Environment:** Production clusters
- **Tools:** Java/Scala, C++; IDEs; continuous integration, source control, …
- **Goal:** Build applications, keep applications running within SLAs

cloudera

Scaling data science on Hadoop

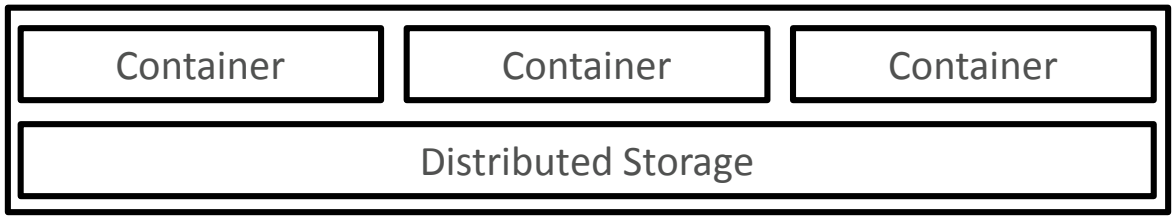
1. Desktop / Local Resources

Jobs run on user's local machine, so can use traditional single-machine libraries. Connects to shared data.



2. Desktop / Shared Resources

Same as #1, except as containers on a shared cluster; shared metadata, security, governance, management.

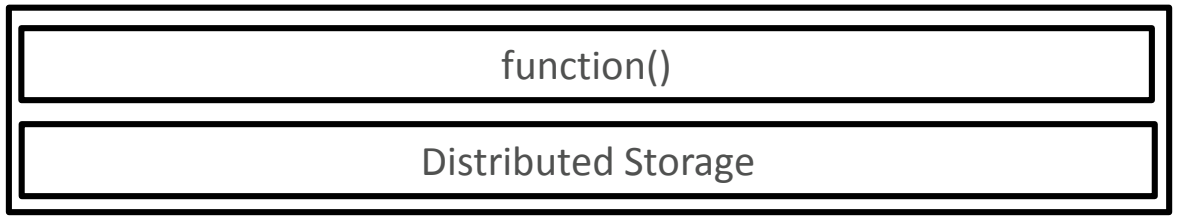


3. Embarrassingly Parallel / Shared Resources

Jobs run in parallel over partitioned data, supporting traditional single-machine libraries. Examples: PySpark UDFs/lambdas, Hadoop Streaming. Use cases: Cross-validation, hyperparameter optimization

4. Distributed Applications / Shared Resources

Jobs call libraries/algorithms designed for distributed computation. User does not necessarily need to be aware of data distribution. Examples: Spark MLlib; H2O



cloudera

R

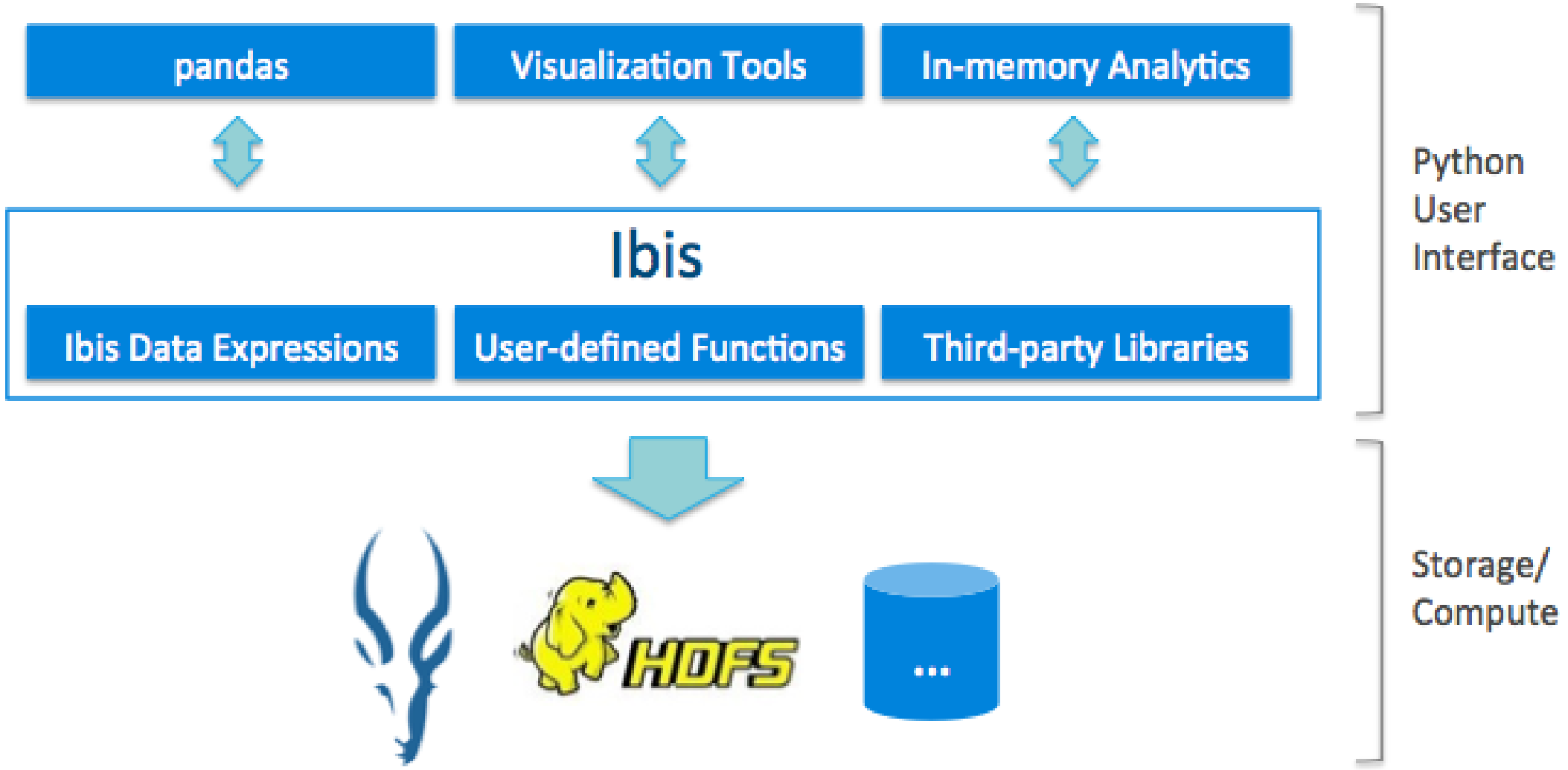
- dplyr on Impala
  - Data wrangling
- Microsoft R Server
  - Library of work-alike R primitives
  - Custom execution engine on Hadoop
  - Not a way to speed up existing R code
- H2O / 0xdata
  - Same architecture
  - Open Source

cloudera

Python: Ibis

- Crafting a compelling Python-on-Hadoop user experience
- Apache-licensed, open source http://github.com/cloudera/ibis
- Pandas and Scikit Learn integration on the roadmap

cloudera



cloudera

Java/Scala: Spark MLlib

- Model building on Spark
  - http://spark.apache.org/docs/latest/mllib-guide.html
- Basics covered
  - LR, SVM, Decision tree
  - PCA, SVD
  - K-means
  - ALS
  - PMML export
- Integrates easily with Spark-as-ETL tool
- Oryx – recommender system built on Spark
  - Model serving

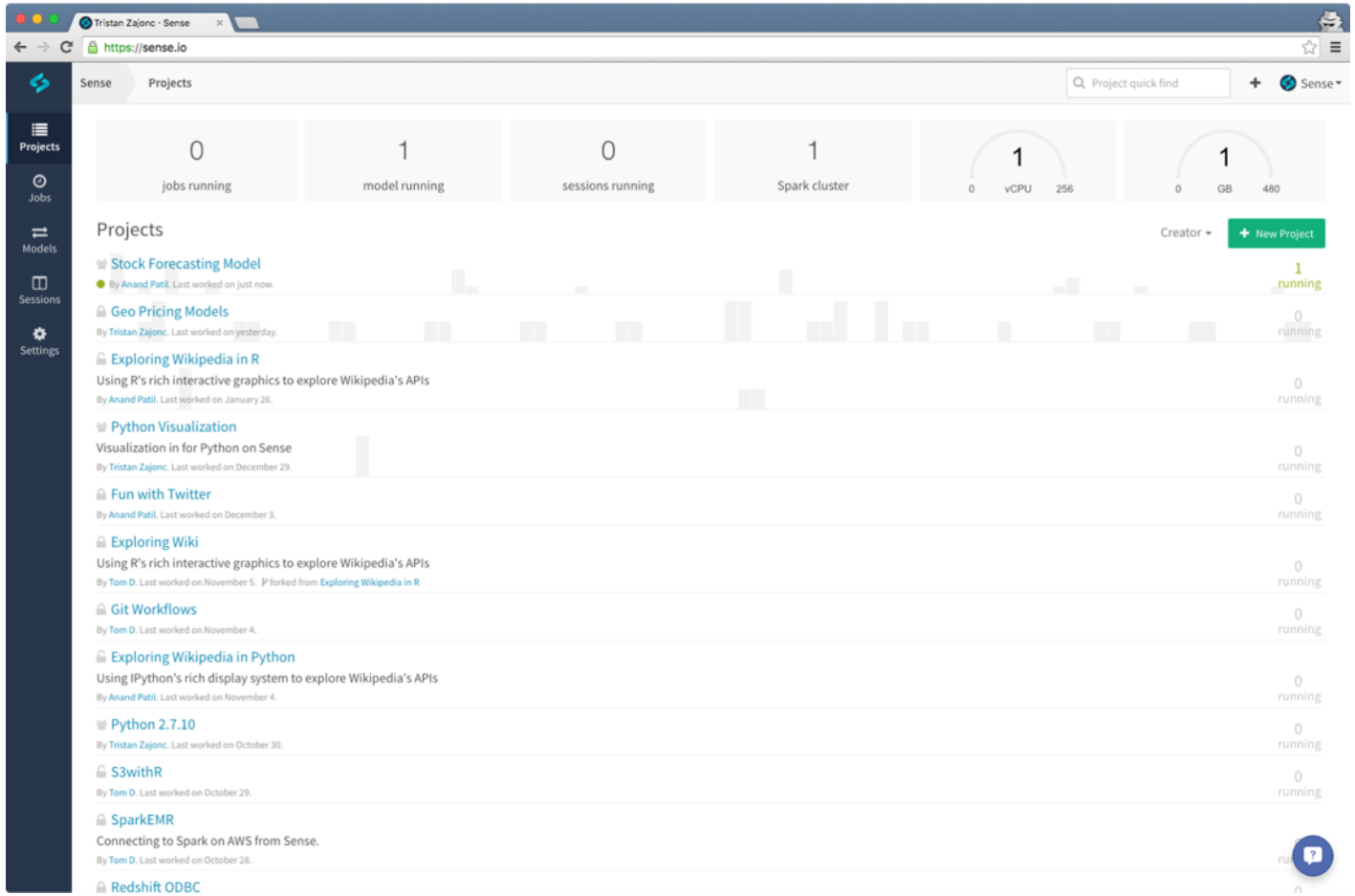
Cloudera Sense

A unified platform to accelerate data science from exploration to production.

1. Team Productivity
Sense Workbench

2. Automation
Sense Pipelines

3. Data Products
Sense Models



cloudera

cloudera

Thank you

@tom_e_white

tom@cloudera.com

Data	Modeling / insights	Production
Raw data collection / curation	Feature engineering	Model deployment
ETL / Data transformation	Model development / research	Model serving / scaling
Exploratory data analysis	Reproducible analysis development	Data application / product development

cloudera

Language Roadmap

Improving interoperability with Hadoop

- **Python**
  - Reference Python distribution (Anaconda parcel)
  - Read/write Parquet from Pandas (with Arrow)
  - Improved PySpark UDF performance (with Arrow)
  - Data frame interoperability (Feather)
- **R**
  - Partners: H2O, Microsoft R Server
  - Data frame interoperability (Feather)
- **SAS**
  - Easier SAS ACCESS setup/installation

cloudera