gluent.

Hadoop for Database Professionals

NoCOUG Fall Conference 2017

About Gluent

Globally distributed team with a deep background in building high performance enterprise applications and systems …

Now on a mission to liberate enterprise data.

Dallas, TX USA

London, England UK

Leeds, England UK

gluent.

we liberate enterprise data

We connect enterprise applications

transparently

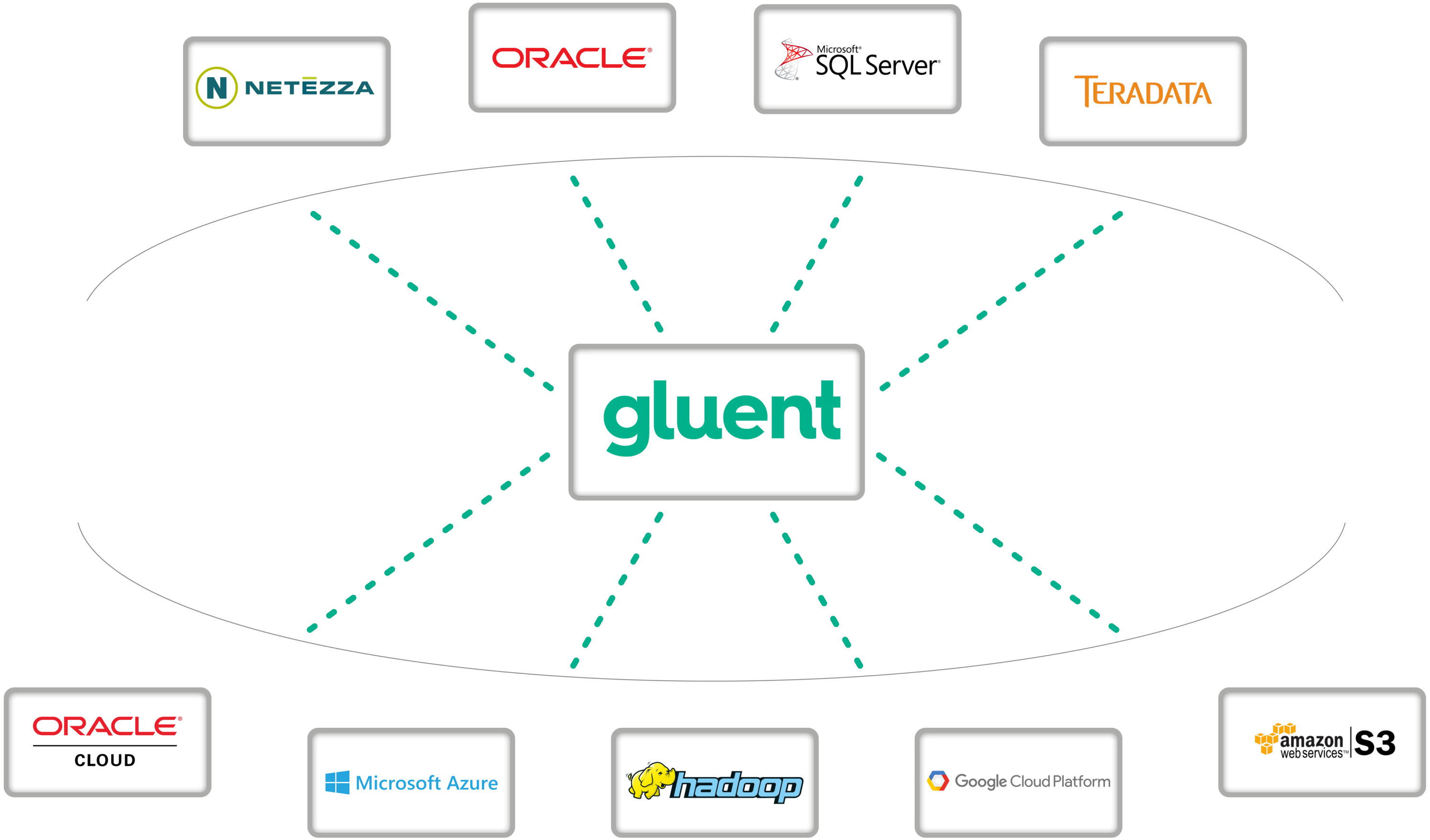
to cloud & big data platforms

Industry recognition for Gluent

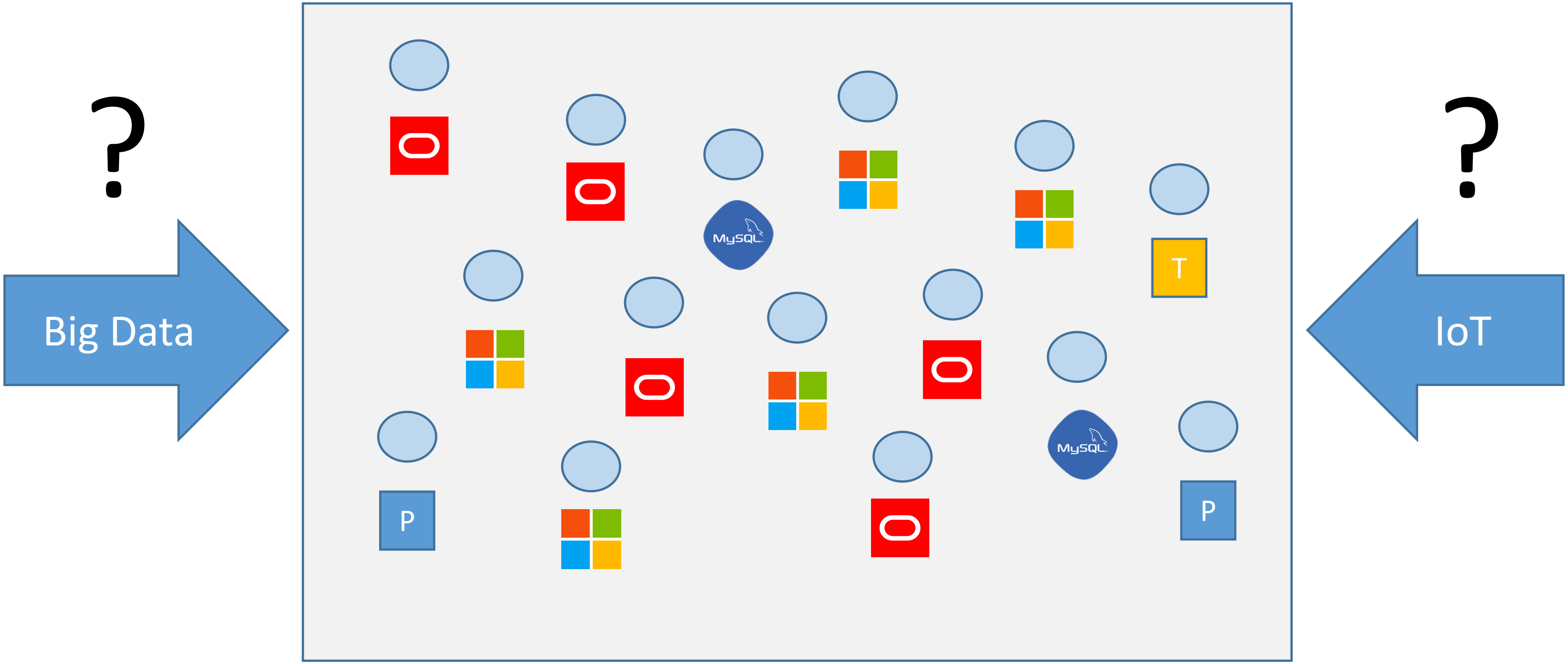
- Selected as a Gartner 2017 Cool Vendor in Data Management
  - https://gluent.com/cool-vendor-2017
- Recognized in “The 10 Coolest Big Data Startups of 2017” on CRN
  - https://gluent.com/gluent-recognized-in-top-10-coolest-startups-of-2017-on-crn/
- 2nd Place in Strata Startup Showcase
  - https://gluent.com/gluent-takes-2nd-place-in-strata-startup-showcase/

What is Gluent?

Gluent data sharing platform

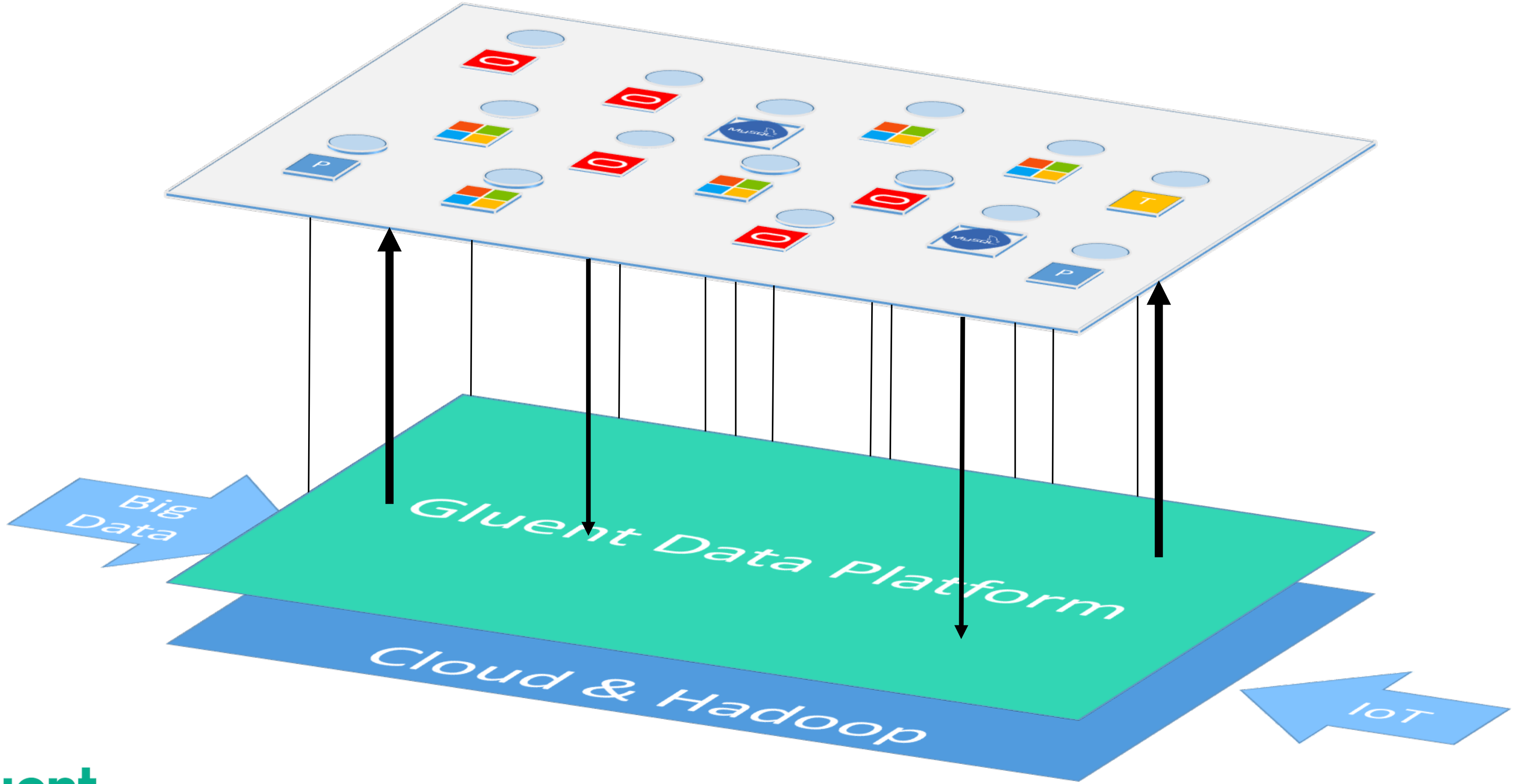


Enterprise applications run on enterprise databases



… but traditional databases don't cut it anymore!

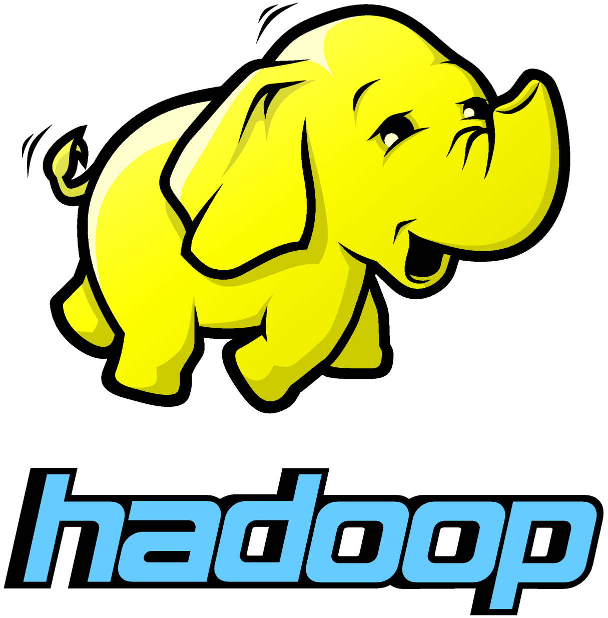
Gluent data virtualization



gluent.

Agenda

- Introduction and Fundamentals of Hadoop (11:30 - 12:15)
  - *Lunch break (45 min)*
- Hadoop Storage and Data Ingestion (13:00 - 13:45)
  - *Networking break (30 min)*
- SQL Processing in Hadoop (14:15 - 15:00)
  - *Break (15 min)*
- Hadoop in Action (15:15 - 16:00)
  - *Break (15 min)*
- Free beer! (NoCOUG networking event)
  - *Q&A after each section*

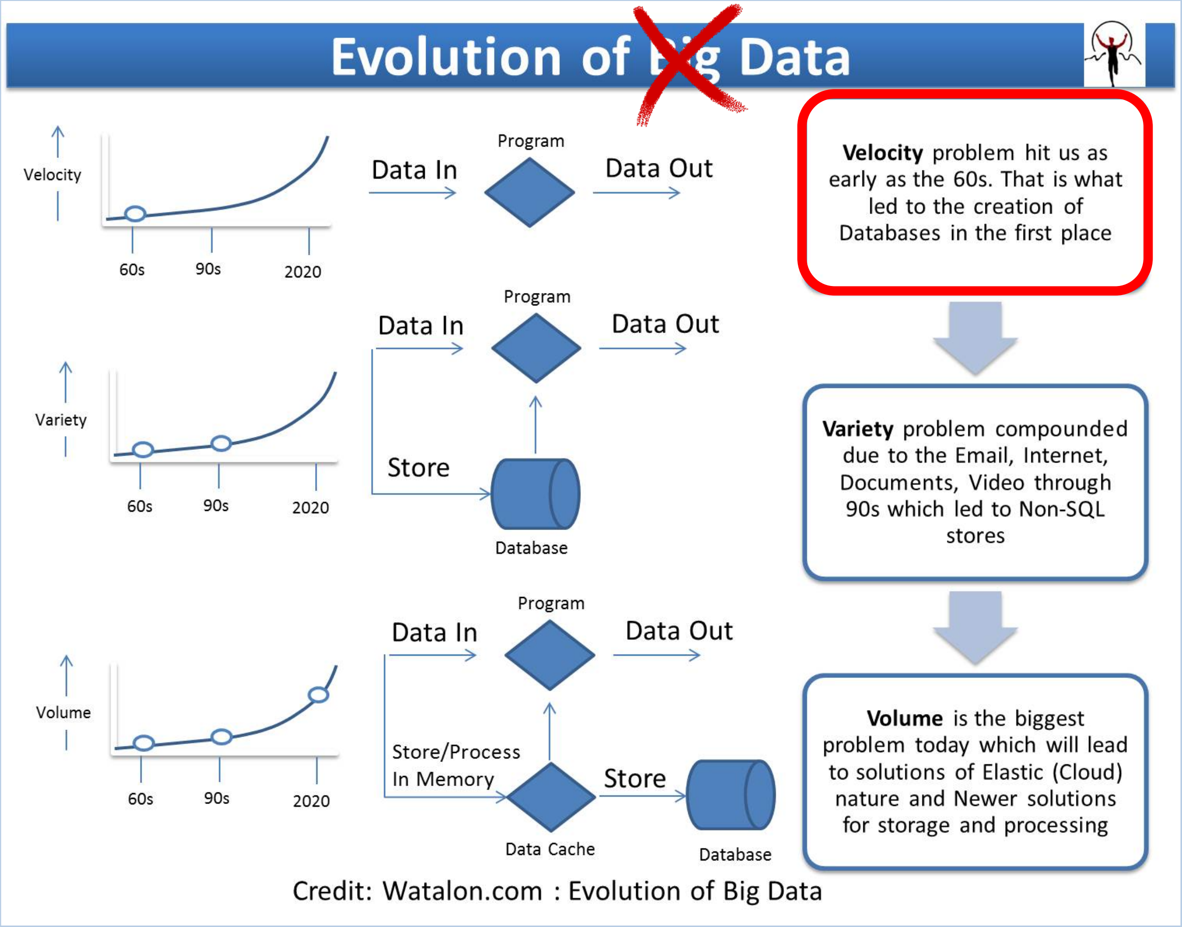


Introduction and Fundamentals of Hadoop

gluent.

Why Hadoop?

The evolution of data



Evolution of ~~Big~~ Data

Velocity problem hit us as early as the 60s. That is what led to the creation of Databases in the first place

Variety problem compounded due to the Email, Internet, Documents, Video through 90s which led to Non-SQL stores

Volume is the biggest problem today which will lead to solutions of Elastic (Cloud) nature and Newer solutions for storage and processing

Credit: Watalon.com : Evolution of Big Data

Source: http://watalon.com/?p=720

How we got here

- In the old days
  - If you run out of database server space, the only option was to buy a new, bigger server & bigger shared SAN storage arrays. This became expensive.

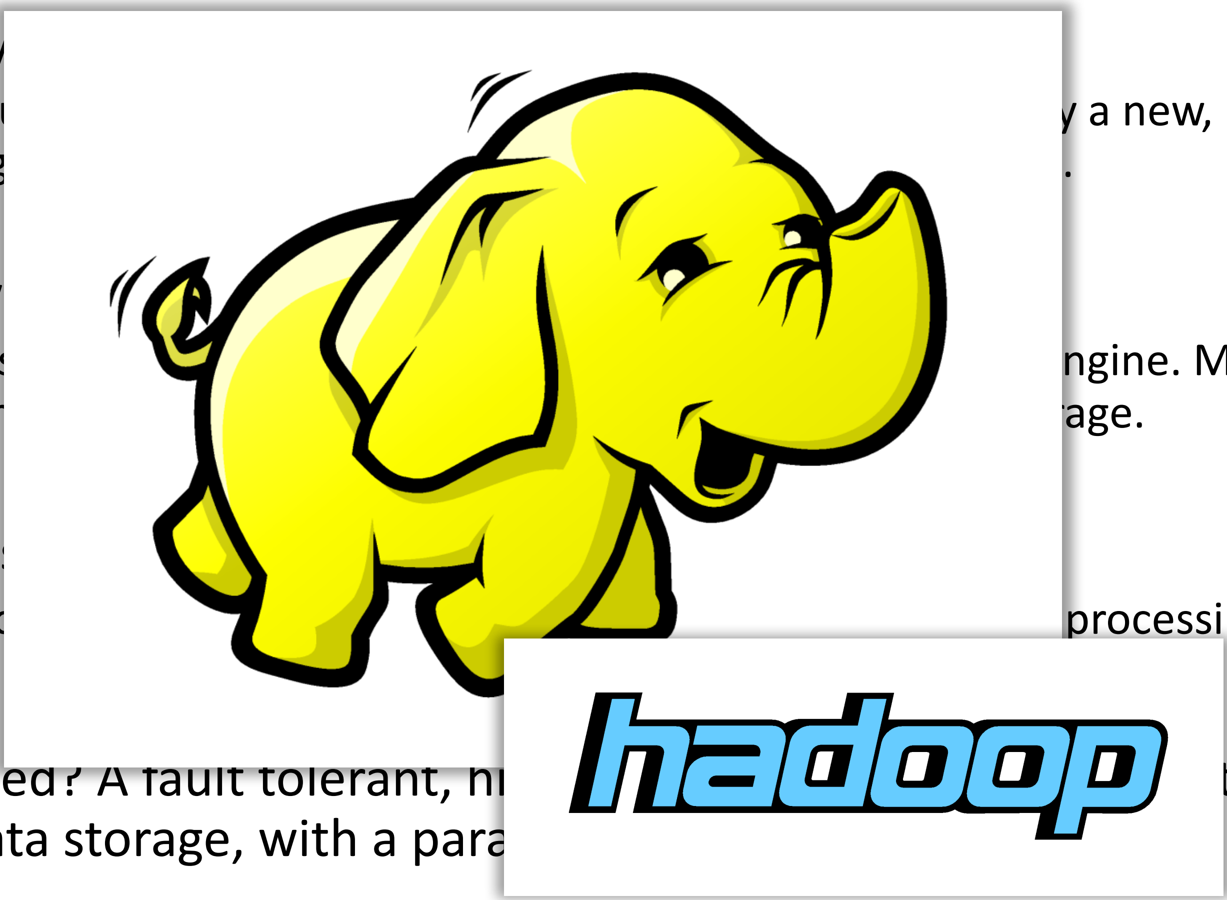
- Then… a new approach
  - Build out distributed systems. Multiple servers as the computing engine. Many disks in many separate servers logically used as large “shared” storage.

- But…these distributed systems can become complex
  - Write your own code to distribute data, handle failures, distribute processing, etc.

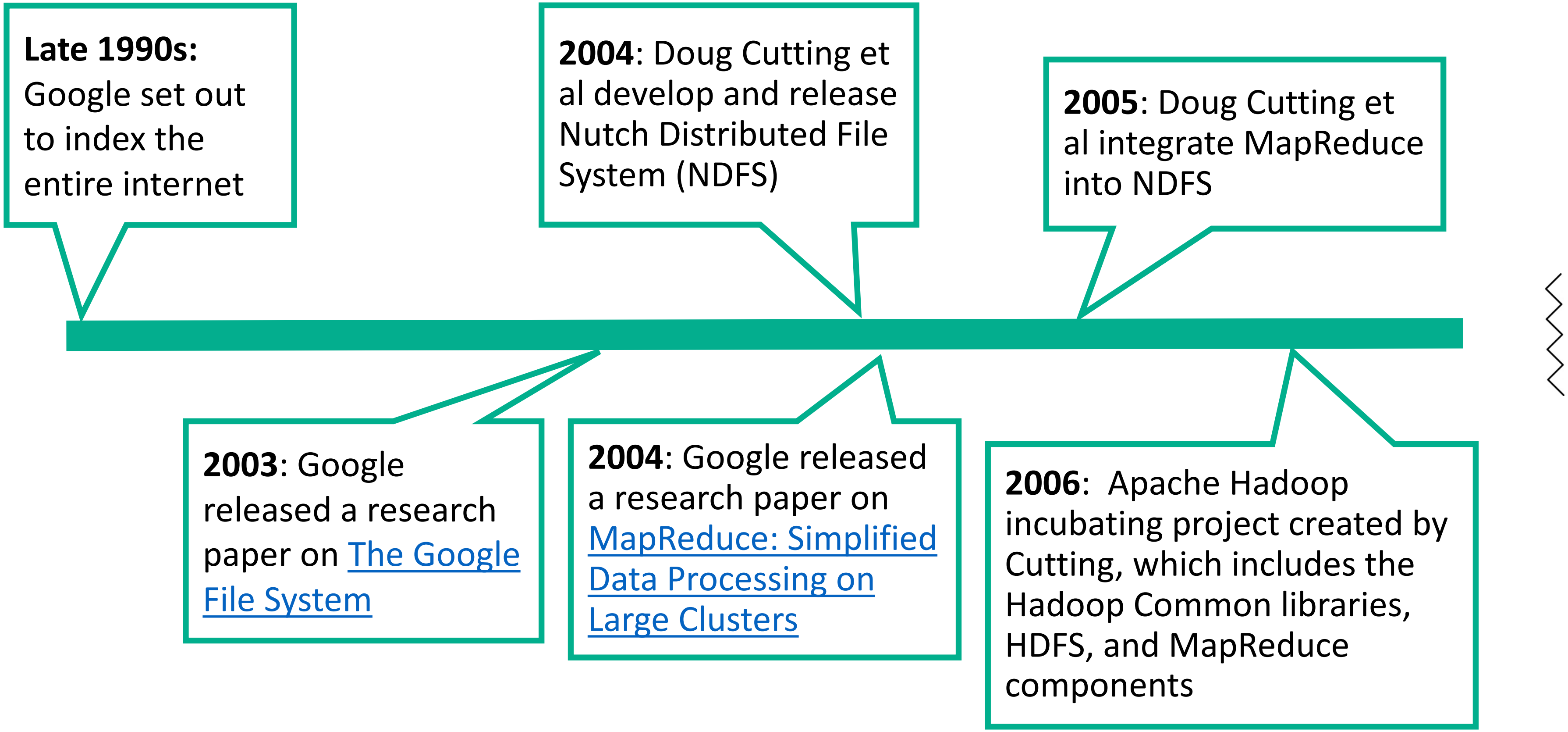
- What is needed? A fault tolerant, highly available, durable, distributed system for data storage, with a parallel processing framework.

How we got here

- In the old day
  - If you run o y a new, bigger server & big .
- Then… a new
  - Build out di ngine. Many disks in mar age.
- But…these di
  - Write your processing, etc.
- What is needed? A fault tolerant, h ted system for data storage, with a para



A brief history of Hadoop



Late 1990s: Google set out to index the entire internet

2003: Google released a research paper on The Google File System

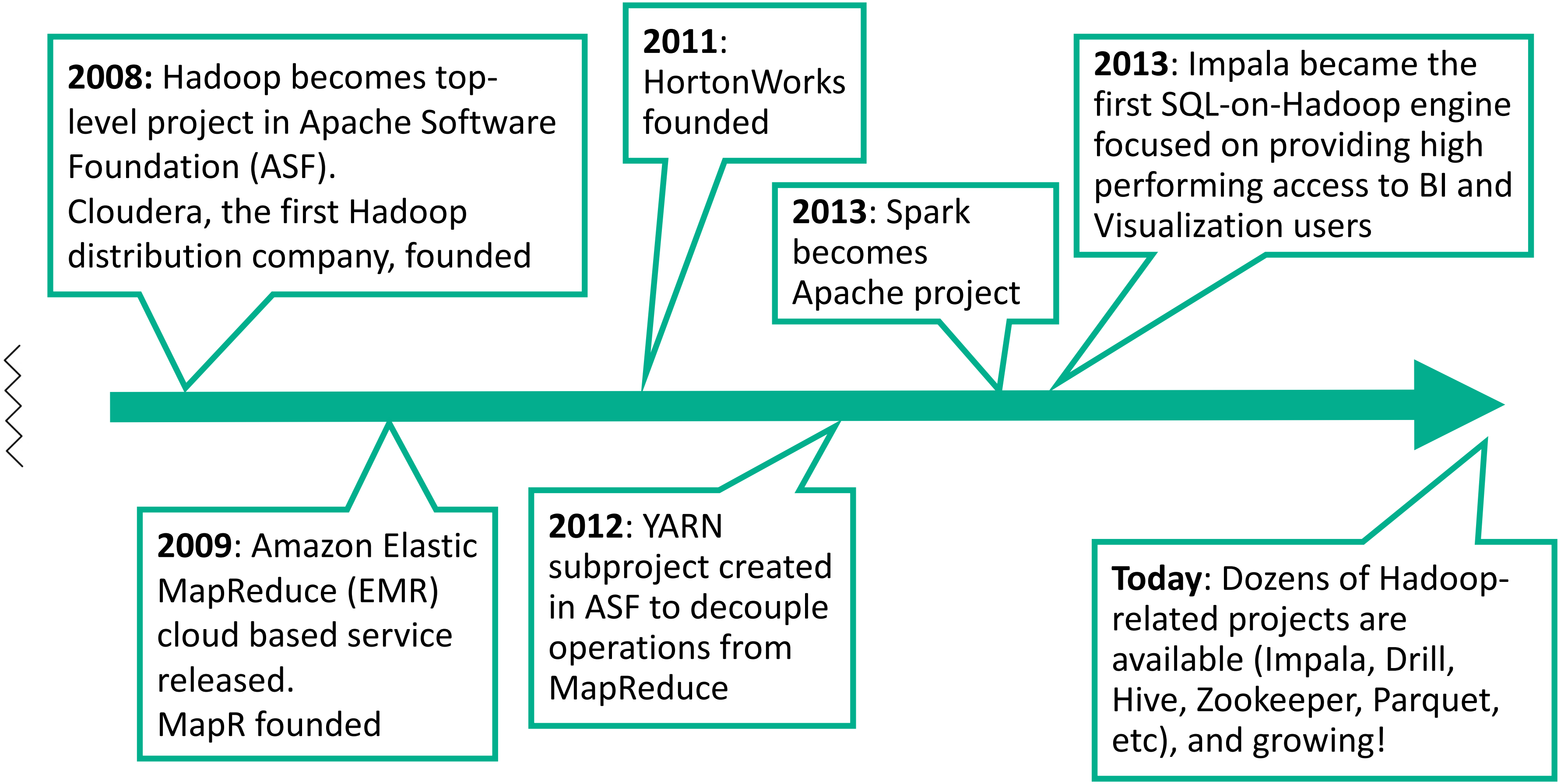
2004: Doug Cutting et al develop and release Nutch Distributed File System (NDFS)

2004: Google released a research paper on MapReduce: Simplified Data Processing on Large Clusters

2005: Doug Cutting et al integrate MapReduce into NDFS

2006: Apache Hadoop incubating project created by Cutting, which includes the Hadoop Common libraries, HDFS, and MapReduce components

A brief history of Hadoop



2008: Hadoop becomes top-level project in Apache Software Foundation (ASF).
Cloudera, the first Hadoop distribution company, founded

2009: Amazon Elastic MapReduce (EMR) cloud based service released.
MapR founded

2011: HortonWorks founded

2012: YARN subproject created in ASF to decouple operations from MapReduce

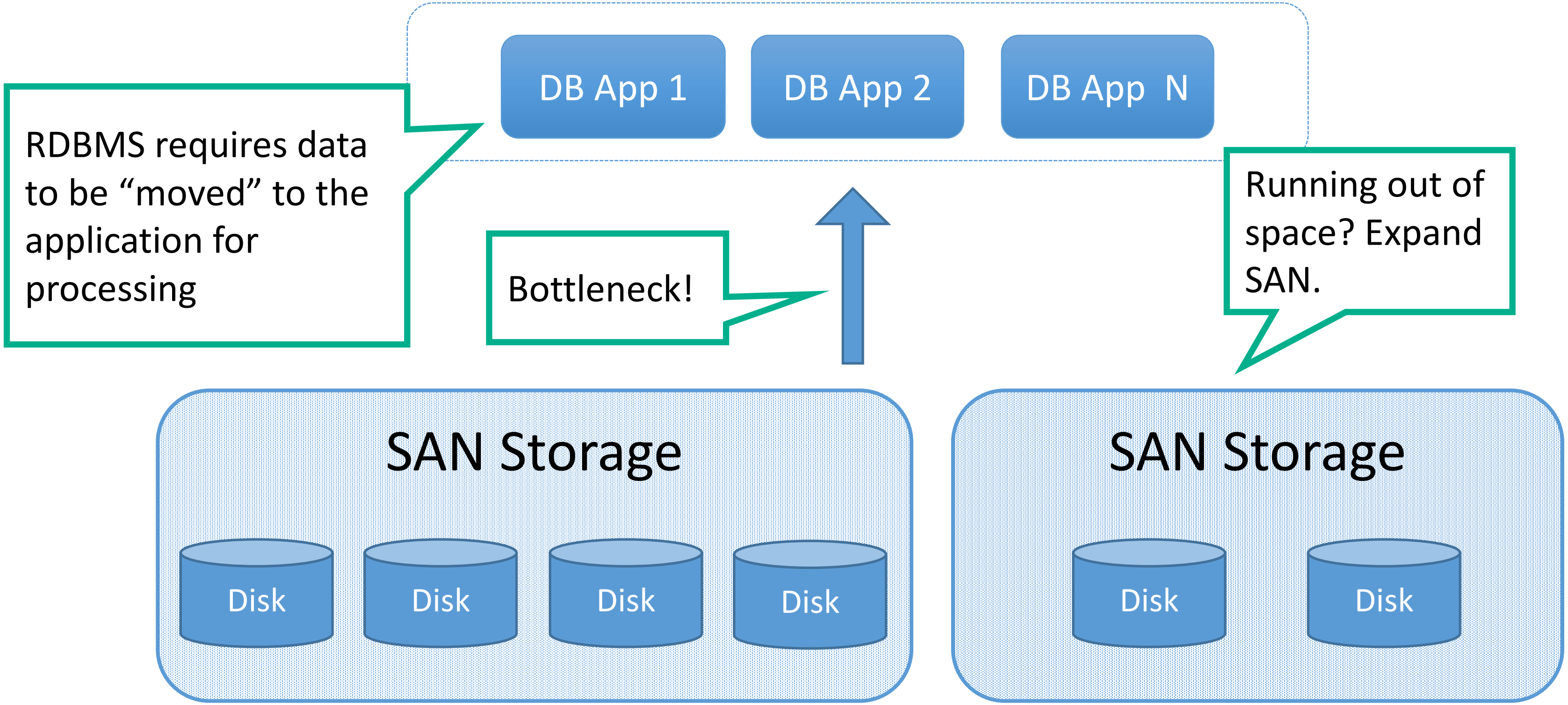
2013: Spark becomes Apache project

2013: Impala became the first SQL-on-Hadoop engine focused on providing high performing access to BI and Visualization users

Today: Dozens of Hadoop-related projects are available (Impala, Drill, Hive, Zookeeper, Parquet, etc), and growing!

What is so different about Hadoop?

RDBMS architecture

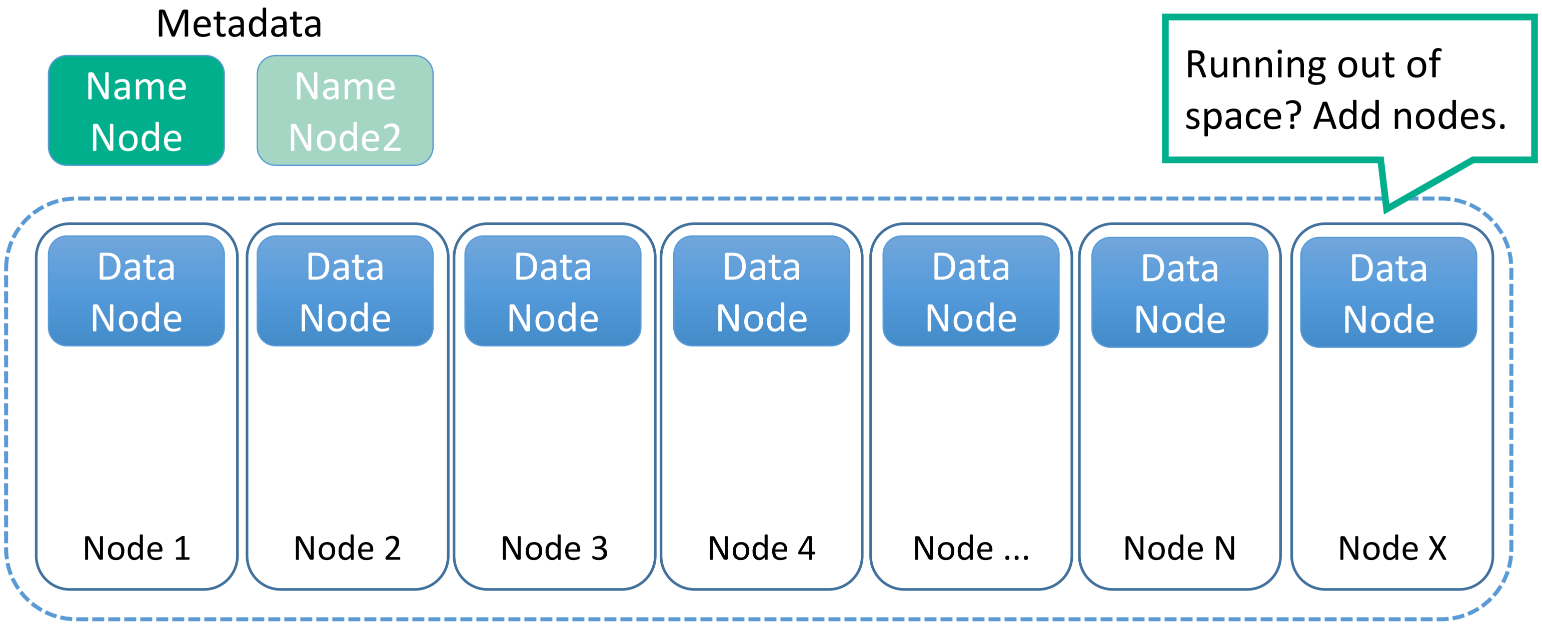


RDBMS requires data to be “moved” to the application for processing

Bottleneck!

Running out of space? Expand SAN.

HDFS architecture

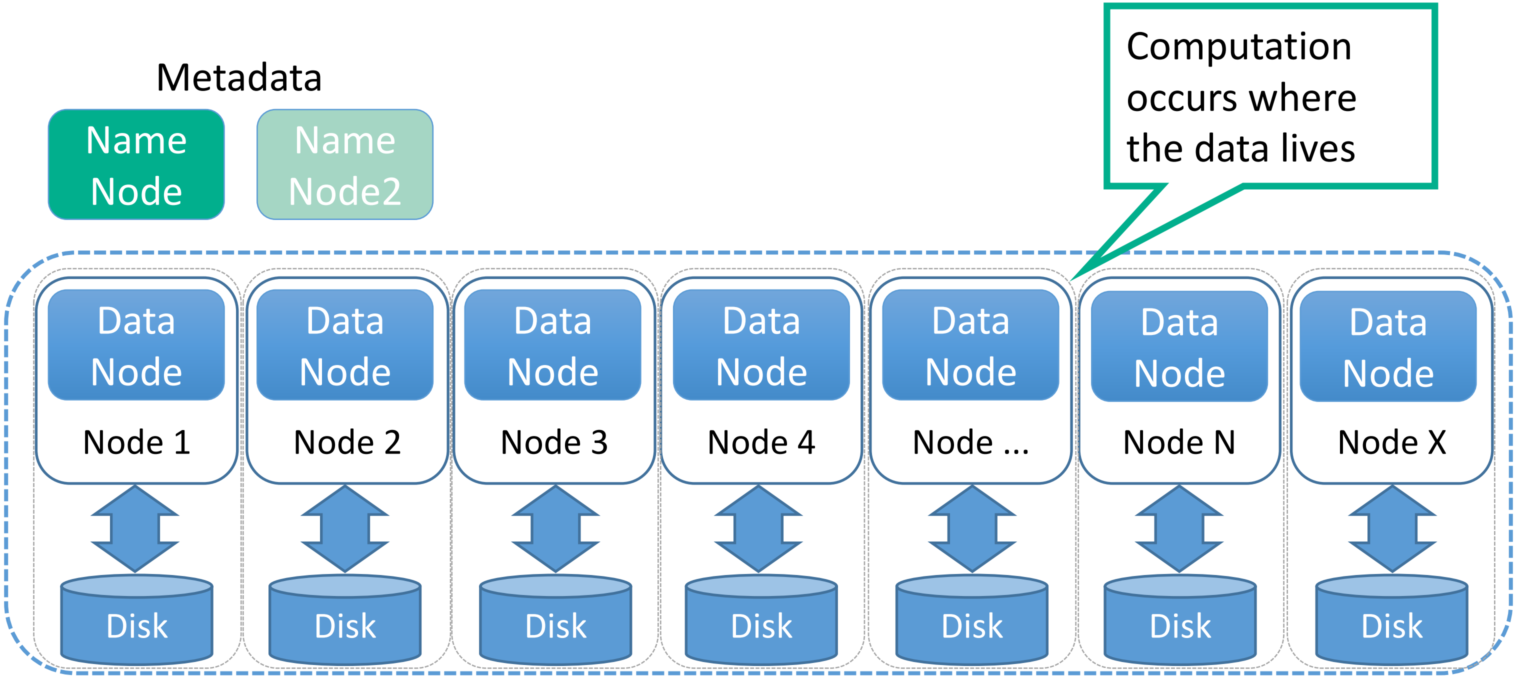


Running out of space? Add nodes.

Hadoop Distributed File System (HDFS)

gluent.

HDFS scalability and performance



Computation occurs where the data lives

Hadoop Distributed File System (HDFS)

“Moving Computation is Cheaper than Moving Data”

Scalable and open!

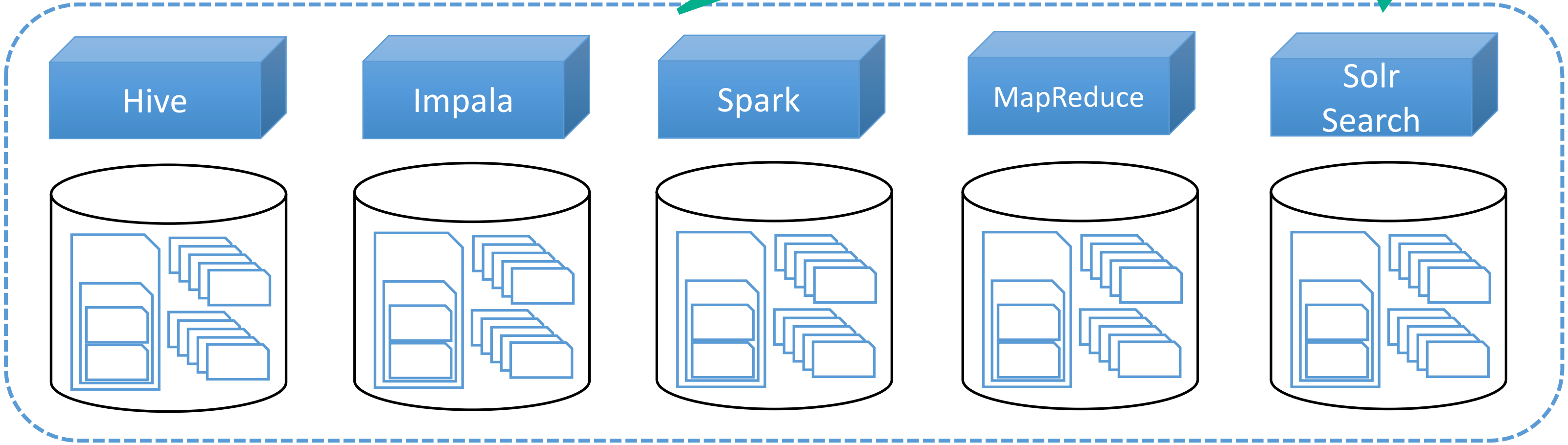
- Scalability in Software!

- Open Data Formats
  - Future-proof!

- **One Data, Many Engines!**

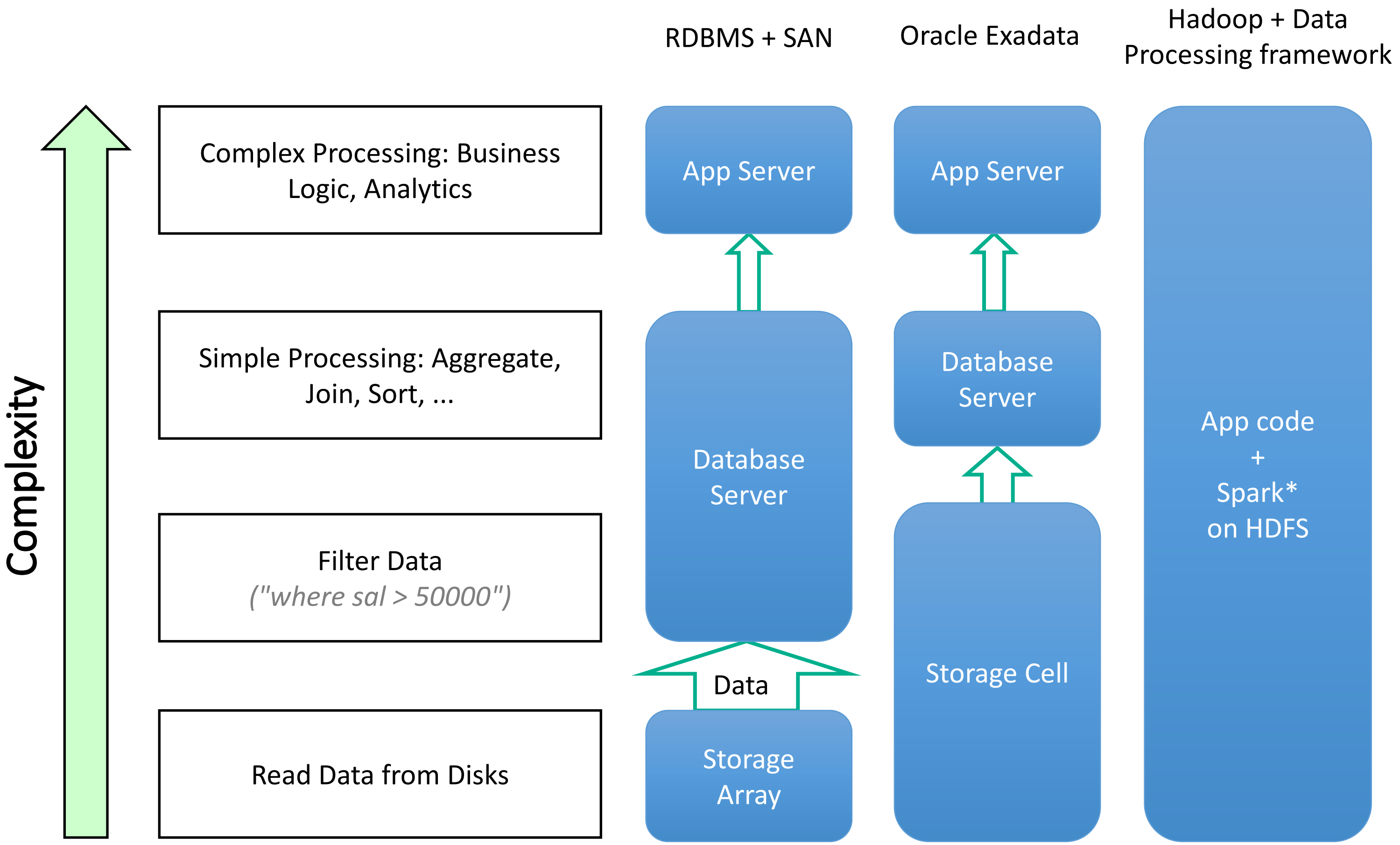
SQL-on-Hadoop is only one of many applications of Hadoop

- Full text search
- Image Processing
- Log processing
- Streams



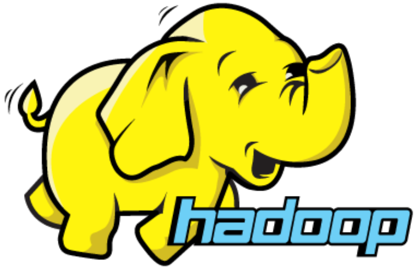
gluent.

Data processing examples



gluent.

What’s in the box? (Core Hadoop)



hadoop
Map Reduce

hadoop
HDFS

hadoop
YARN

Common Hadoop myths

Myth #1: Hadoop *is* MapReduce

- MapReduce is just *one* component of Hadoop
  - Hadoop v1, MapReduce was THE component for data retrieval and resource management
  - Hadoop v2 includes YARN and the open source community has built up MapReduce data access alternatives

- MapReduce is not the only way to access data
  - You don’t need an army of Java developers to use Hadoop
  - SQL engines on Hadoop make use of a familiar syntax for accessing data
  - MapReduce is rarely used in practice or by other software (ex. Hive on Spark)

Hadoop != MapReduce

Myth #1: Hadoop *is* MapReduce

- MapReduce is just *one* component of Hadoop
  - Hadoop v1, MapReduce was THE component for data retrieval and resource management
  - Hadoop v2 includes YARN and the open source community has built up MapReduce data access alternatives

BUSTED!

- MapReduce is not the only way to access data
  - You don’t need an army of Java developers to use Hadoop
  - SQL engines on Hadoop make use of a familiar syntax for accessing data
  - MapReduce is rarely used in practice or by other software (ex. Hive on Spark)

Hadoop != MapReduce

Myth #2: SQL on Hadoop is basically Hive

HIVE
Spark SQL
APACHE DRILL
TM
APACHE PHOENIX
presto

- There are many more SQL engines on Hadoop that can provide SQL-like data access, just as Hive does

SQL on Hadoop is rapidly evolving!

Myth #2: SQL on Hadoop is basically Hive

BUSTED!

- There are many more SQL engines on Hadoop that can provide SQL-like data access, just as Hive does

SQL on Hadoop is rapidly evolving!

Myth #3: Data Governance is non-existent

- Data classification
- Metadata management
- Data lineage, search, and data discovery
- Policy engine and auditing

- Many tools available, such as Cloudera Navigator

Data Governance on Hadoop is real

Myth #3: Data Governance is non-existent

- Data classification
- Metadata management
- Data lineage, search and data discovery
- Policy engine and auditing
- Many tools available, such as Cloudera Navigator

BUSTED!

Data Governance on Hadoop is real

gluent.

Myth #4: Hadoop is insecure

- Authentication
  - Kerberos, LDAP (for SQL engines)
- Authorization
  - SQL engines: Role Based Access Control (Sentry)
  - HDFS: Filesystem ACLs
- Encryption
  - Data at rest: HDFS encryption (every directory can have its own keys)
  - Data in-flight: SSL/TLS

Hadoop security is enterprise ready

Myth #4: Hadoop is insecure

- Authentication
  - Kerberos, LDAP (for SQL engines)
- Authorization
  - SQL engines: Role Based Access Control (Sentry)
  - HDFS: File System ACLs
- Encryption
  - Data at rest: HDFS encryption (every directory can have its own keys)
  - Data in-flight: SSL/TLS

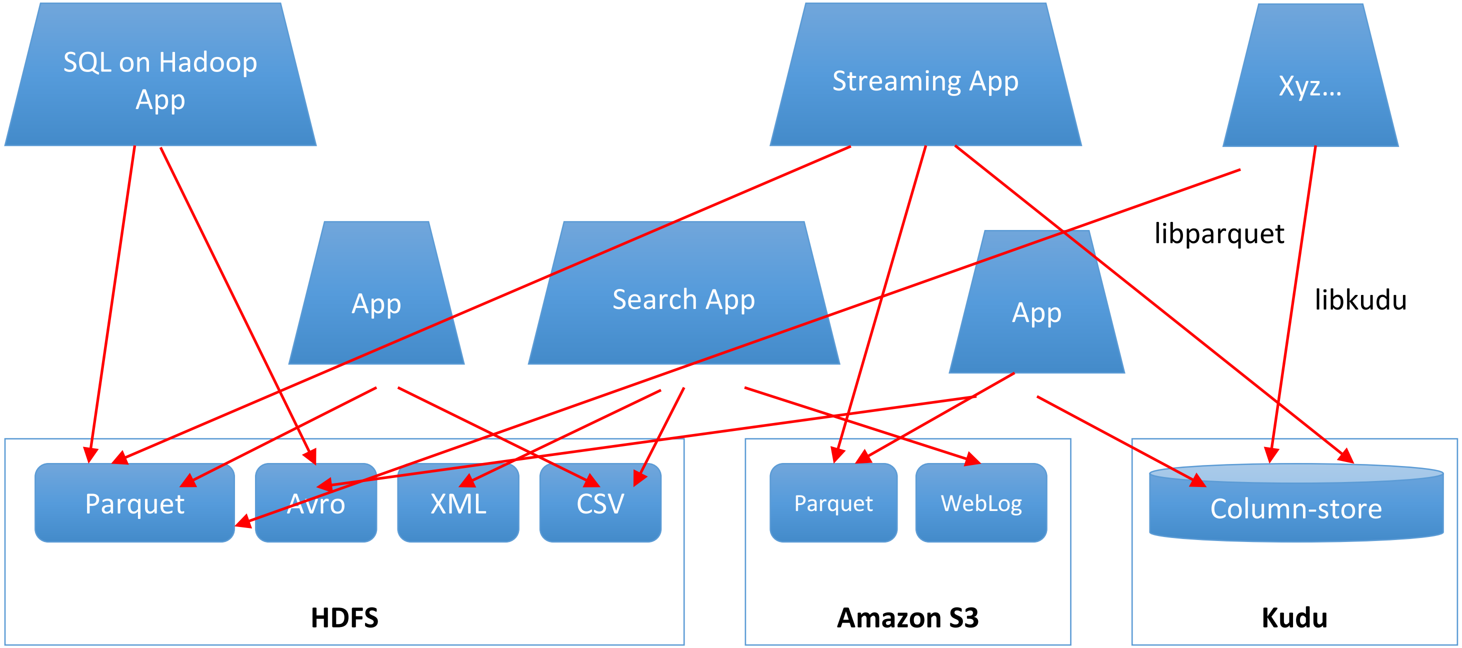
BUSTED!

Hadoop security is enterprise ready

What else can we do with Hadoop?

One Data, Many Engines!

flexible, future proof data systems!



gluent.

Hadoop ecosystem

- Hadoop is not a single product, but an ecosystem
  - Mostly (Apache) open source
  - Every component/layer has multiple alternatives
  - Different vendor distributions prefer different components

  - **SQL-on-Hadoop:** Hive, Impala, Drill, SparkSQL, Presto, Phoenix
  - **Storage subsystems:** HDFS, Kudu, S3, MapR-FS, EMC Isilon
  - **Resource management:** YARN, Mesos, Myriad, MapReduce v1

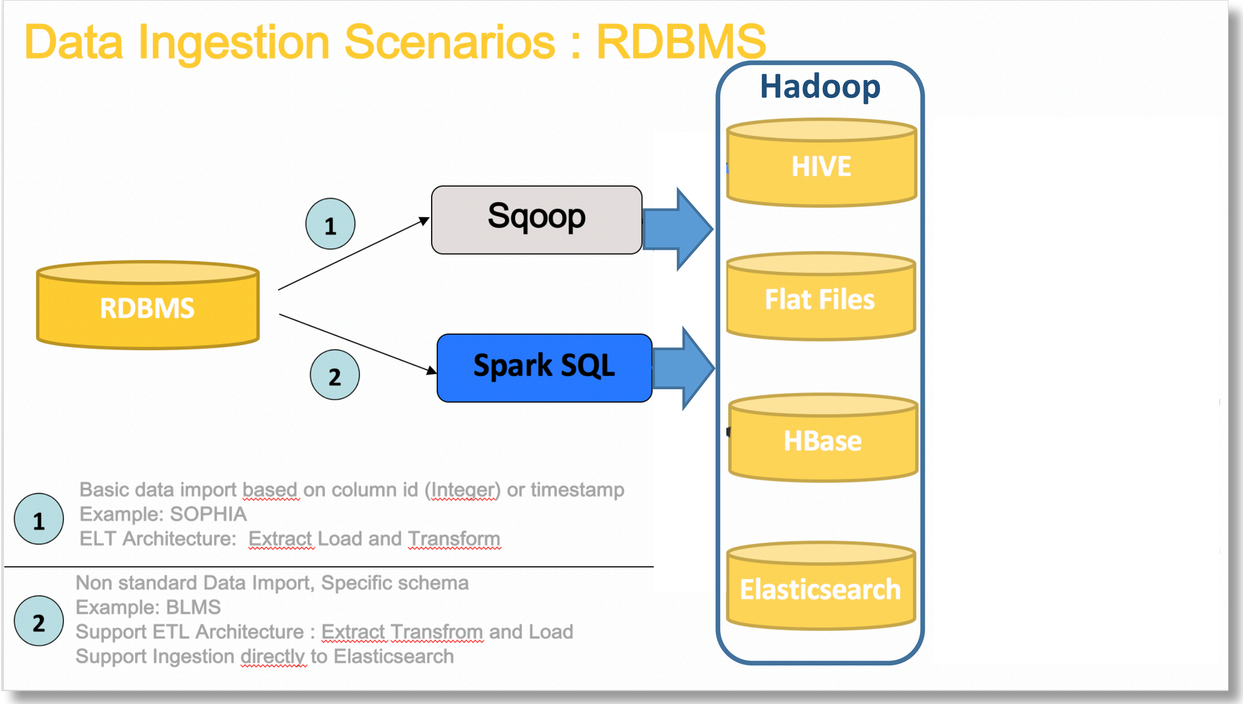
- Hadoop Ecosystem Table
  - A good reference for looking up component names & what they do
  - https://hadoopecosystemtable.github.io/

Two types of use cases for Hadoop

1. The game-changing super-awesome Big Data predictive analytics deep learning AI magic (*that somehow will make us money*)

2. Cheap storage and scalable computation

New World: This is what’s happening



Data Ingestion Scenarios : RDBMS

1 Basic data import based on column id (Integer) or timestamp
Example: SOPHIA
ELT Architecture: Extract Load and Transform

2 Non standard Data Import, Specific schema
Example: BLMS
Support ETL Architecture : Extract Transfrom and Load
Support Ingestion directly to Elasticsearch

Old World
Closed
Silo
Expensive

New World
Open
Cheap
Scalable

* Source: The Hadoop Deployment Strategy at Renault Group (Hadoop Summit 2016)

gluent.

RDBMS vs Hadoop – Technology comparison

	RDBMS	Hadoop
Storage format	Database tables / data files	File system
Schema	Structured Static Required on Write	Multi / semi-structured Dynamic Required on Read (schema-on-read)
Processing	Bring data to the processing	Bring processing to the data
Consistency	Immediate + multi-row ACID	Eventual or immediate + single-row
Hardware	High-end servers	Commodity boxes
Software cost	License and support	Subscription
Scalability	Possible to scale, but difficult & expensive	Clustered, built to scale
High availability	Yes (build your own)	Yes (built-in)

Challenges for database professionals diving into Hadoop

- Hadoop is not only a database
  - It's a filesystem with optional SQL layers on top

- Additional technical skills
  - OS administration (Unix/Linux)
  - Network administration (at least decent understanding)
  - Kerberos (security)

- Troubleshooting
  - Lots of Java/JVMs!
  - Lots of log files!

- Workload management
  - Running out of memory, historically a problem

Recap - Introduction and Fundamentals of Hadoop

- Hadoop was created to support what we call “Big Data”
  - High volumes of structured or semi-structured data
  - Distributed file systems provide fault tolerance and durability
  - Processing occurs where the data lives
- Hadoop is open
  - Storage formats, storage options, resource management, data ingestion and processing – all open source and community driven
  - Future proof your data systems!
- Many myths about Hadoop still persist – don’t believe the hype!
- The struggle is real – database systems are running out of space
  - Hadoop, ultimately, is cheap storage and scalable computation
  - Store everything!
- RDBMS vs Hadoop
  - Many high-level similarities, but different in many ways

Q&A

Lunch Break (45 minutes)

Hadoop Storage and Data Ingestion

Data storage in Hadoop

Hadoop Distributed File System (HDFS)

- Standard distributed data storage file system (HDFS)
- Hardware failure is assumed
  - Stores 3 copies (by default) of each block of data
  - Fault-tolerant by design!
- Data is processed on the same server which it is stored
  - *"Moving computation is cheaper than moving data"*
- Java based – can be run on any machine that supports Java
  - Accessed from command line or various GUI tools
  - Commands similar to Unix (ls, mkdir, chown, etc)
- Excellent for large files, but possible issues with very small files
- One “master” node, multiple “slave” nodes

But there are other Hadoop storage options

- Distributed file systems
  - HDFS (most common)
  - GlusterFS
  - Quantcast File System QFS

- Cloud storage
  - Amazon S3 (Simple Storage Service)
  - Google Cloud Storage
  - Microsoft Azure Data Lake Store

- Columnar storage
  - Apache Kudu

Amazon S3: the cloud

- Pay monthly for storage use and data transfer
- Data is preserved if Hadoop cluster shutdown
  - Disaster recovery backup, active-passive high-availability
- Cloud
  - Automatically scales
  - Backups handled by Amazon
  - Data is independent of Hadoop clusters. Any cluster can process data.

- **Important:** S3 is not a direct replacement for HDFS
  - Eventual consistency
  - S3 is not a filesystem
  - Amazon EMR & S3 are a bit different together (see Amazon EMR docs)

S3 is great for…

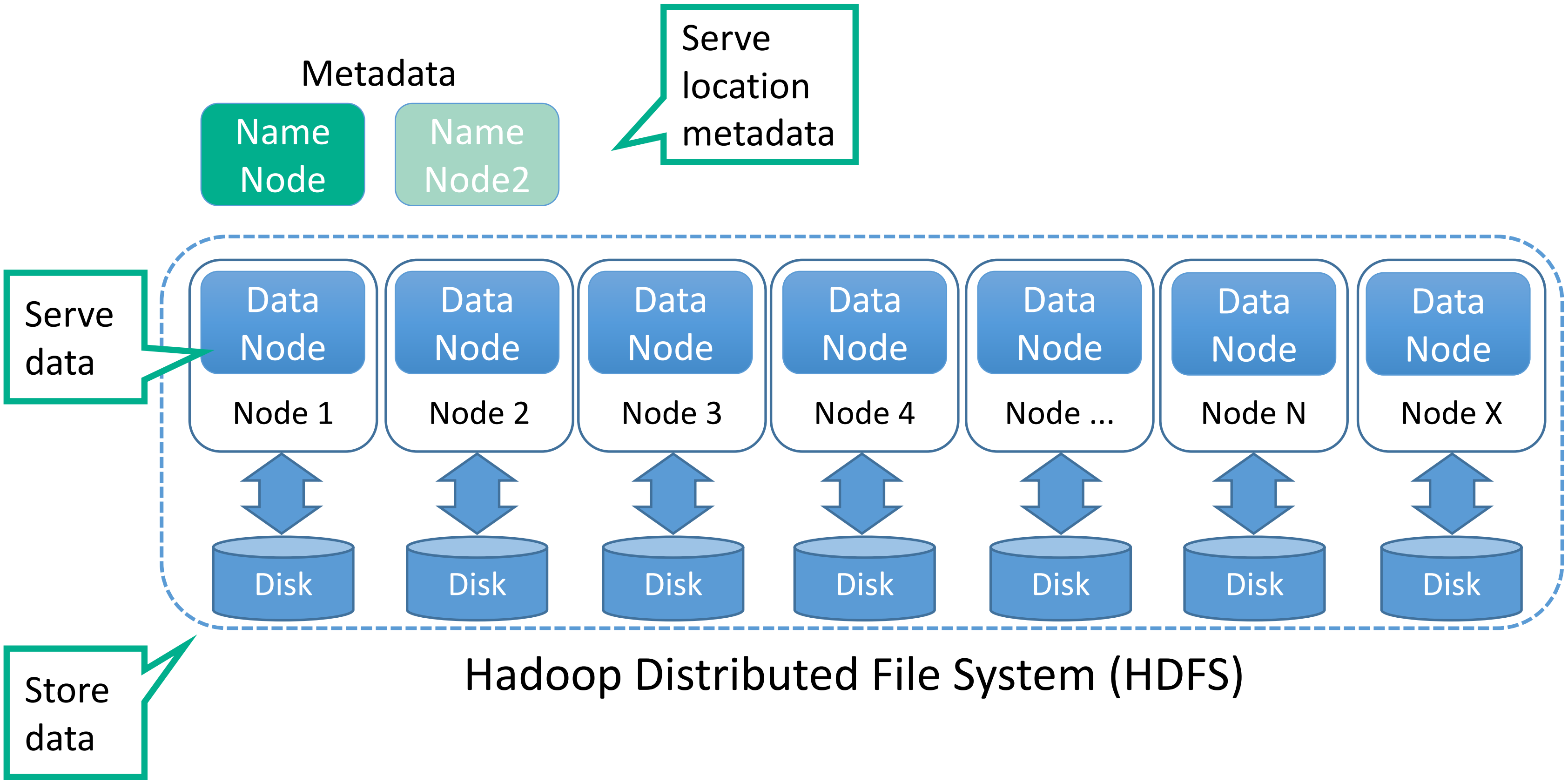
- …creating a backup of your Hadoop cluster
  - HDFS → S3 – Keep the cloud in-sync with local data for disaster recovery
  - Perform analytics / ML on backup copy without impacting production

- …Hadoop clusters running completely in the cloud
  - Amazon EMR (Elastic MapReduce) and S3 play nicely together

- …intermittent use of the data
  - Pay for data processing only when you need to use the data
  - Perfect example of a sandbox or PoC machine

More about HDFS

What’s inside the NameNode & DataNode?

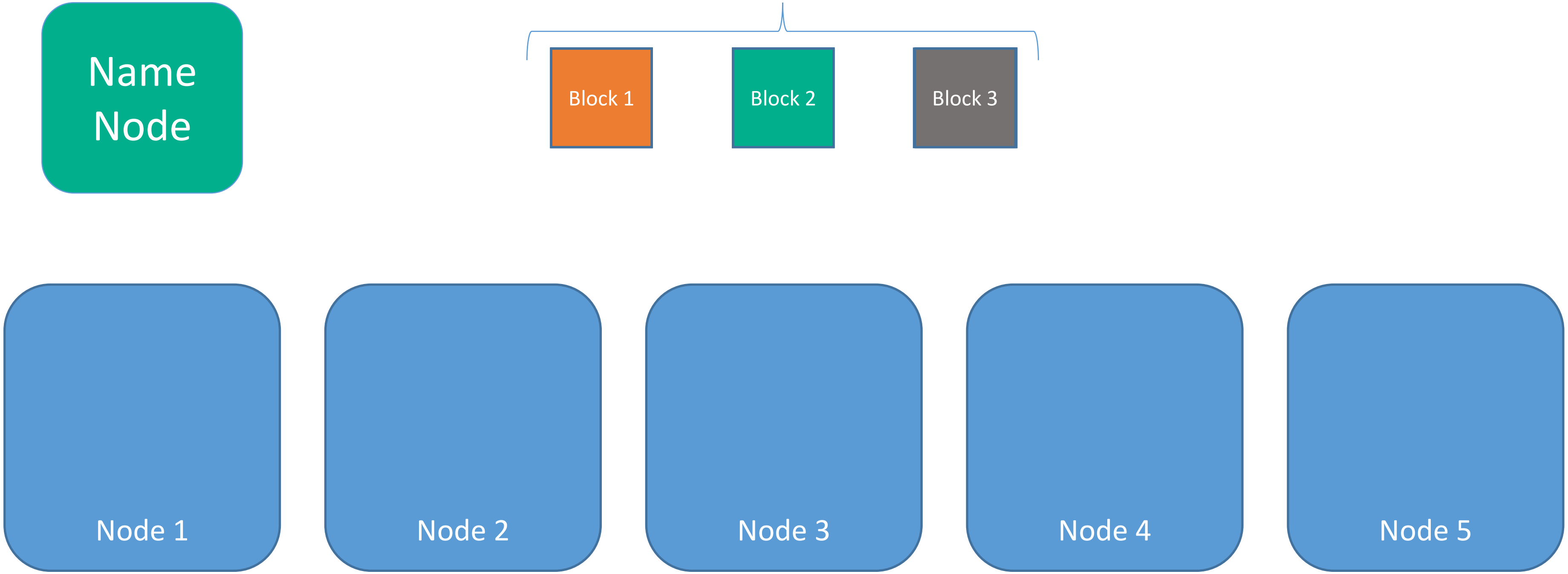


Hadoop Distributed File System (HDFS)

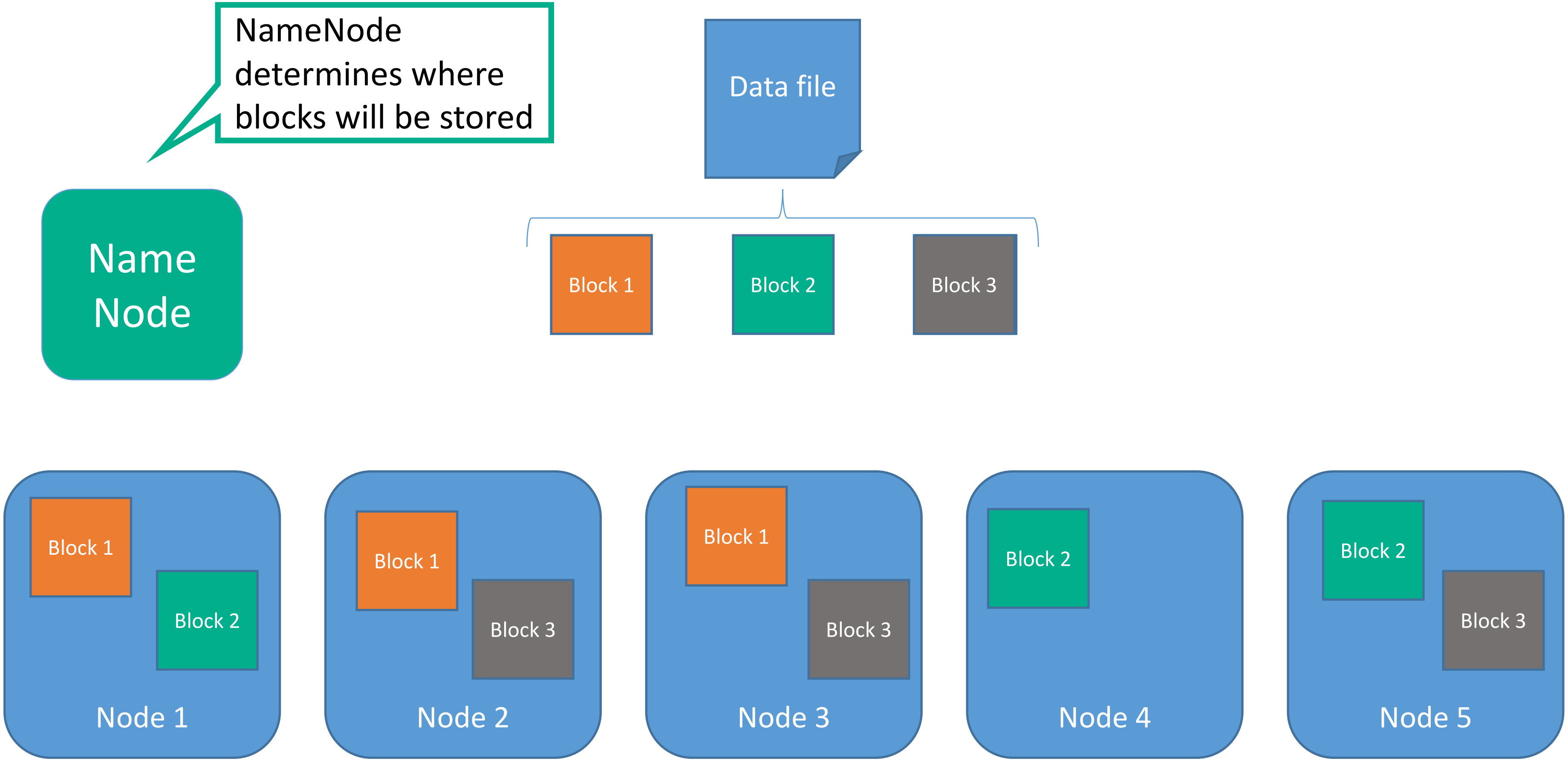
gluent.

How data storage works in HDFS

NameNode determines where blocks will be stored

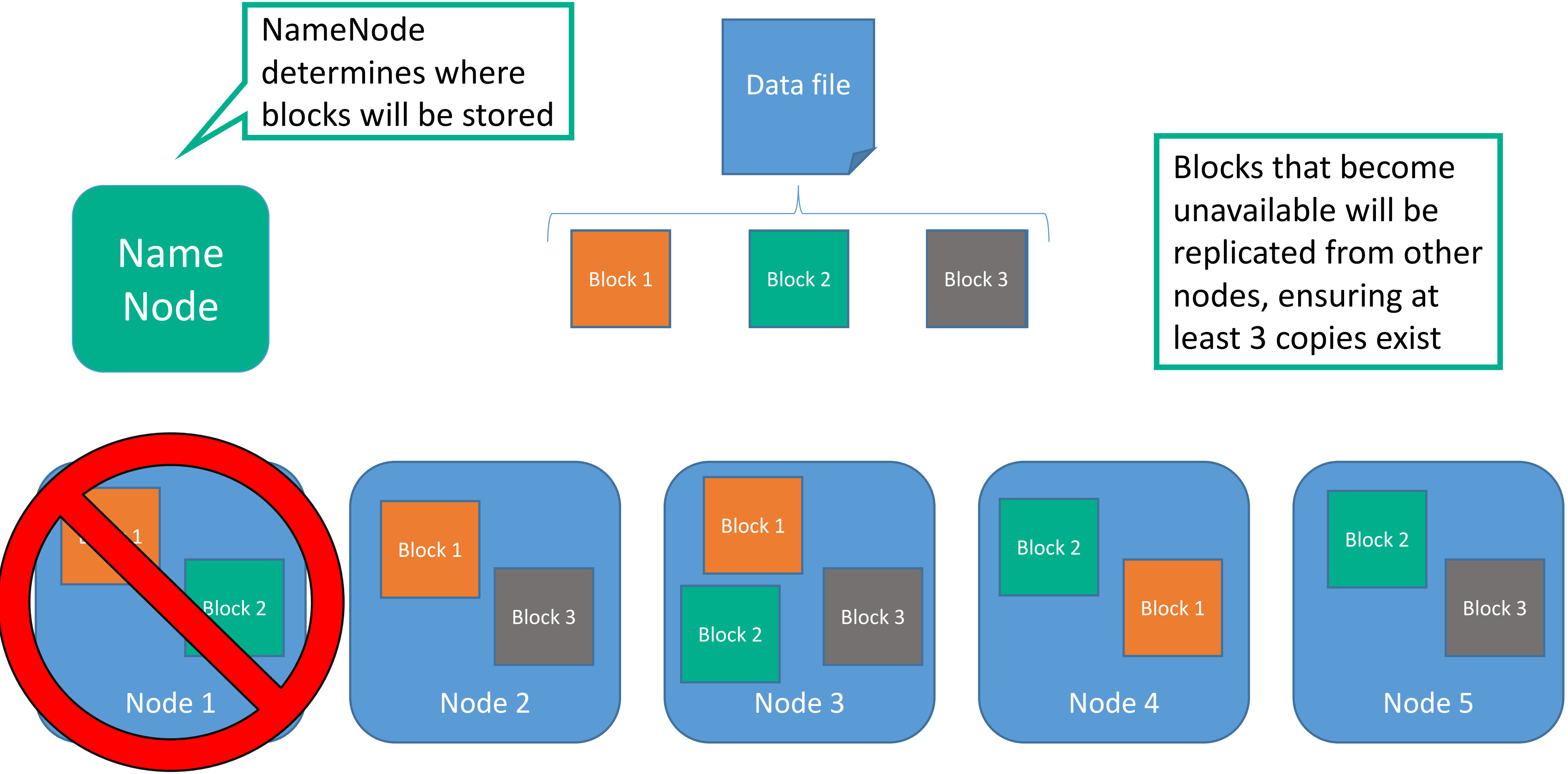


How data storage works in HDFS



How data storage works in HDFS

NameNode determines where blocks will be stored



Blocks that become unavailable will be replicated from other nodes, ensuring at least 3 copies exist

HDFS considerations

- HDFS is designed for "write once, read many" use cases
- Large sequential streaming reads (scans) of files
  - Not optimal for random seeks and small IOs
- "Inserts" are appended to the end of the file
- No updates (by default)
- HDFS performs the replication ("mirroring") and is designed anticipating frequent DataNode failures

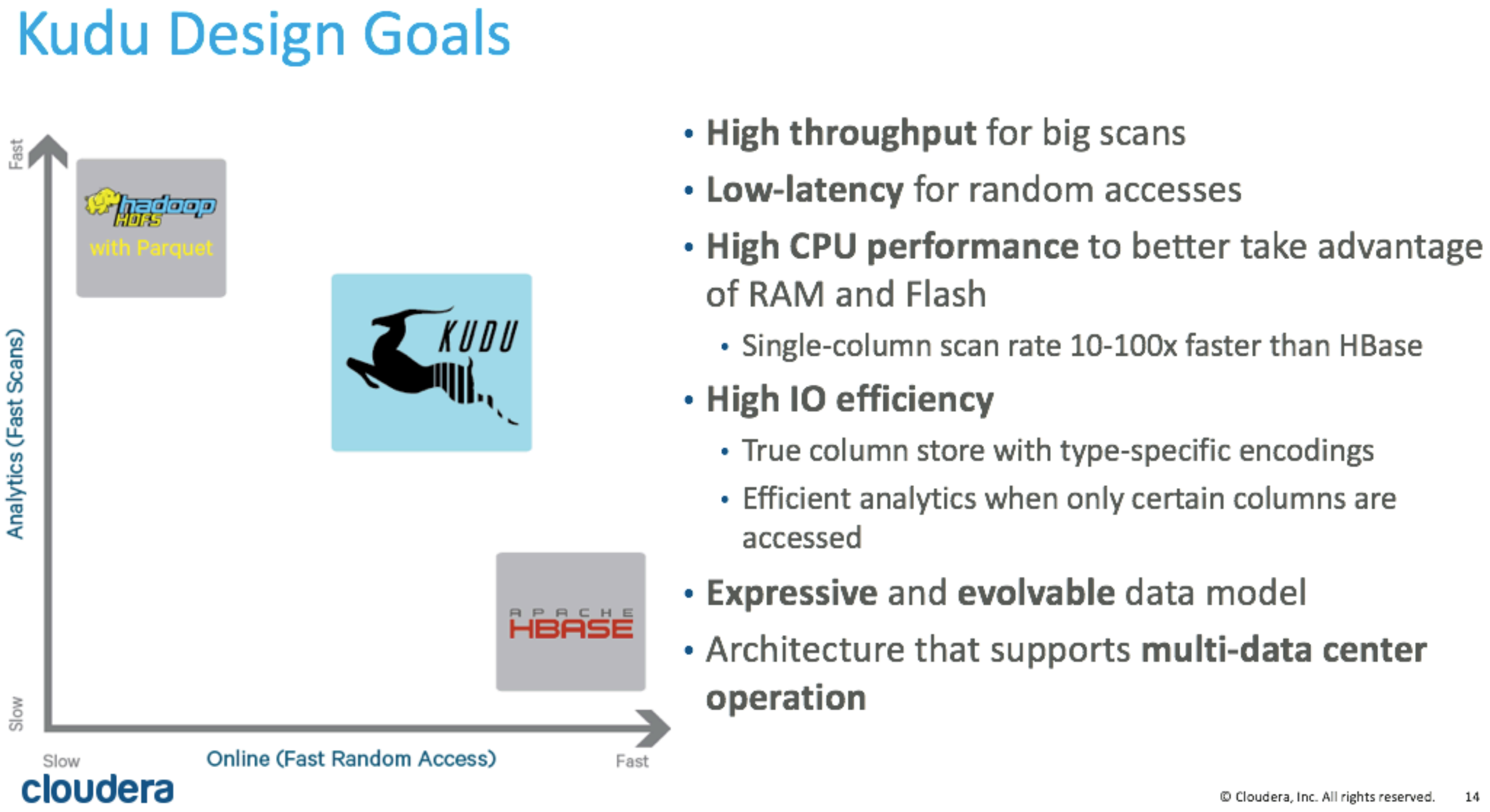
Various other applications (Hbase, Kudu, etc) that allow for updates, random lookups, and resolve some of the other “tradeoffs”, and should be considered

Data formats

Data formats supported by Hadoop

- Standard file formats
  - Plain text (CSV, tab-delimited)
  - Structured text (XML, JSON)
  - Binary (images)
- Hadoop-specific file formats
  - Sequence files
- Serialization formats
  - Avro
- Columnar formats
  - Parquet
  - ORC

The latest in storage for Hadoop – Apache Kudu



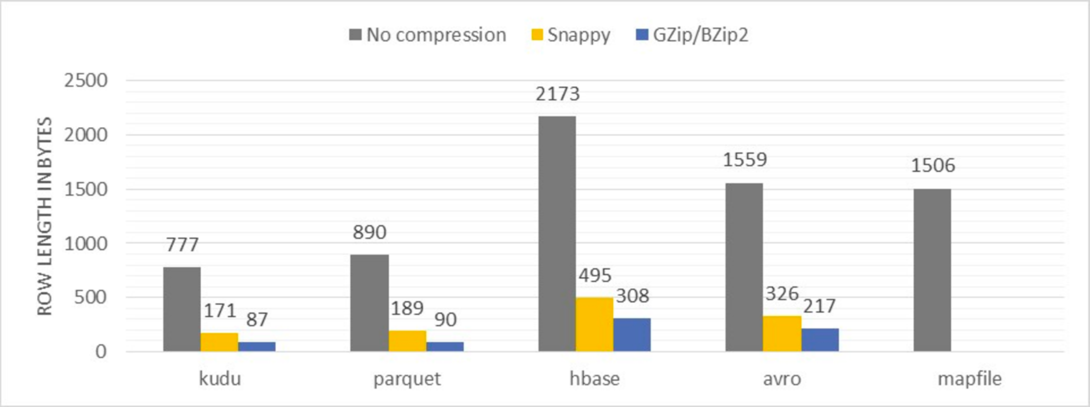
Kudu Design Goals

- **High throughput** for big scans
- **Low-latency** for random accesses
- **High CPU performance** to better take advantage of RAM and Flash
  - Single-column scan rate 10-100x faster than HBase
- **High IO efficiency**
  - True column store with type-specific encodings
  - Efficient analytics when only certain columns are accessed
- **Expressive** and **evolvable** data model
- Architecture that supports **multi-data center operation**

© Cloudera, Inc. All rights reserved. 14

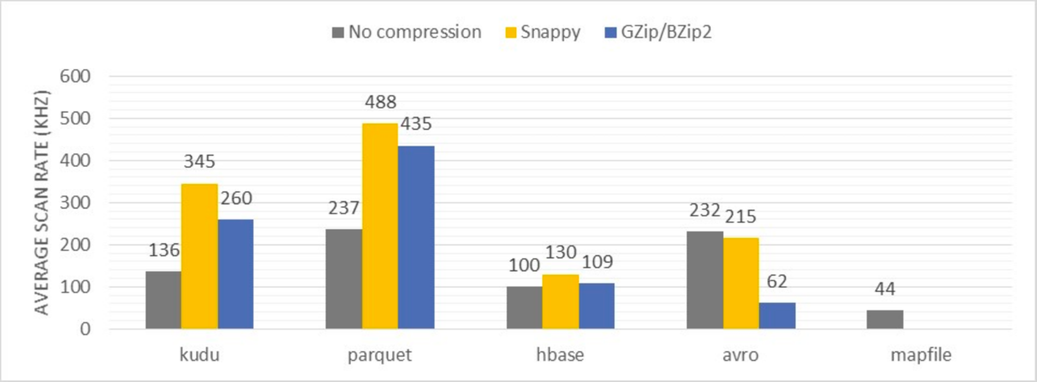
Source: http://www.slideshare.net/jhols1/introduction-to-apache-kudu

Storage format performance comparison



Data space utilization by storage format

Data scan rate by storage format



Source and much more info:
https://db-blog.web.cern.ch/blog/zbigniew-baranowski/2017-01-performance-comparison-different-file-formats-and-storage-engines

Getting data into Hadoop

Data ingestion into Hadoop

Hadoop data load concepts are similar to RDBMS, but the toolset is quite different.

	Oracle	SQL Server	MySQL	Hadoop
Data file load	SQL Loader	BCP Utility	LOAD DATA INFILE	Copy file to HDFS
Bulk data load	Data Pump	BULK Insert	LOAD DATA INFILE	Sqoop
Real-time data load	Various data replication tools	Various data replication tools	Various data replication tools	Flume, Kafka, Storm, etc

HDFS file copy utilities

- Hadoop Common Libraries provide two command line interface approaches to copying files into HDFS

```
# hdfs dfs -put prod_sales_2015.csv /user/hadoop/sales/prod_sales_2015.csv
```

- “copyFromLocal” is similar to “put”, but requires a local file

```
# hdfs dfs –copyFromLocal prod_sales_2016.csv /user/hadoop/sales/prod_sales_2016.csv
```

Bulk data load with Sqoop

- Command line client used to bulk copy data from a relational database to HDFS over JDBC connection
  - Also works in reverse, HDFS to RDBMS
  - Sqoop generates MapReduce jobs for the work

```
sqoop import --connect jdbc:oracle:thin:@myserver:1521/MYDB1
   --username myuser
   --null-string ''
   --null-non-string ''
   --target-dir=/user/hive/warehouse/ssh/sales
   --append -m1 --fetch-size=5000
   --fields-terminated-by '','' --lines-terminated-by ''\\n''
   --optionally-enclosed-by ''\"'' --escaped-by ''\"''
   --split-by TIME_ID
   --query "\"SELECT * FROM SSH.SALES WHERE TIME_ID < DATE'1998-01-01'\""
```

Bulk data load with Sqoop

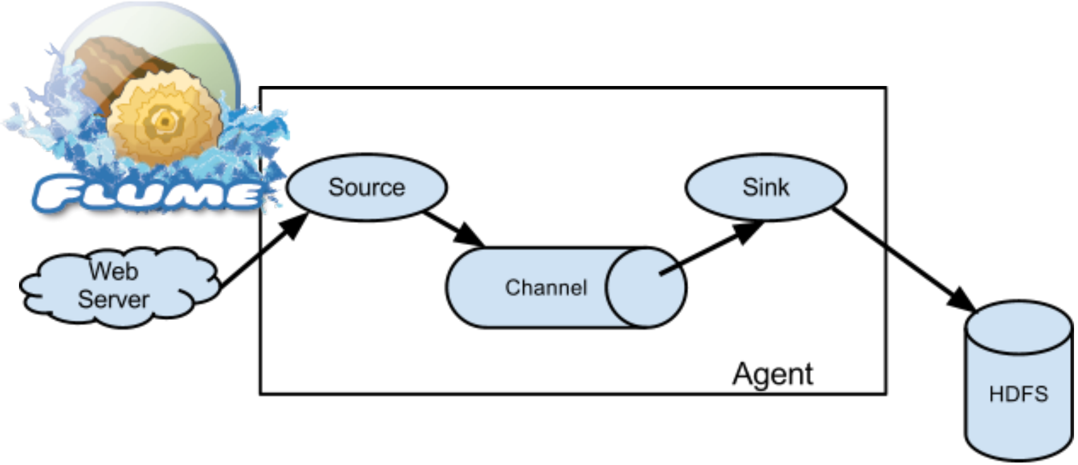
- Command line client used to bulk copy data from a relational database to HDFS over JDBC connection
  - Also works in reverse, HDFS to RDBMS
  - Sqoop generates MapReduce jobs for the work

```
sqoop import \
    --connect jdbc:mysql://127.0.0.1:3306/retail_db \
    --username=myuser \
    --password=mypwd \
    --table products \
    --target-dir=/user/hive/demo \
    --as-parquetfile -m 1 \
    --append
```

Streaming data

- Data streams from external sources (RDBMS, logs, etc), through Flume / Kafka / StreamSets / etc, and into HDFS

- Each technology has its own approach to data movement from various sources to various targets



kafka

Recap - Hadoop Storage and Data Ingestion

- HDFS is the standard distributed file system for Hadoop
  - It's really just a bunch of jar files, config files, and other files in a file system!
  - NameNode tracks metadata and where the data (file) is located
  - DataNode stores the actual data (file blocks) and provides access to the data

- Various data formats are available for data storage in Hadoop, depending on your use case
  - Parquet provides both high compression and high scan efficiency

- Other storage options, such as Amazon S3 in the cloud or Apache Kudu, are available
  - Kudu fills a gap for storage and data access
  - Kudu works best when you need a simultaneous combination of sequential *and* random reads and writes

- Getting data into Hadoop is similar to RDBMS, but with different tools
  - File copy, bulk load, or real-time streaming

Q&A

Break (30 min)

SQL Processing in Hadoop

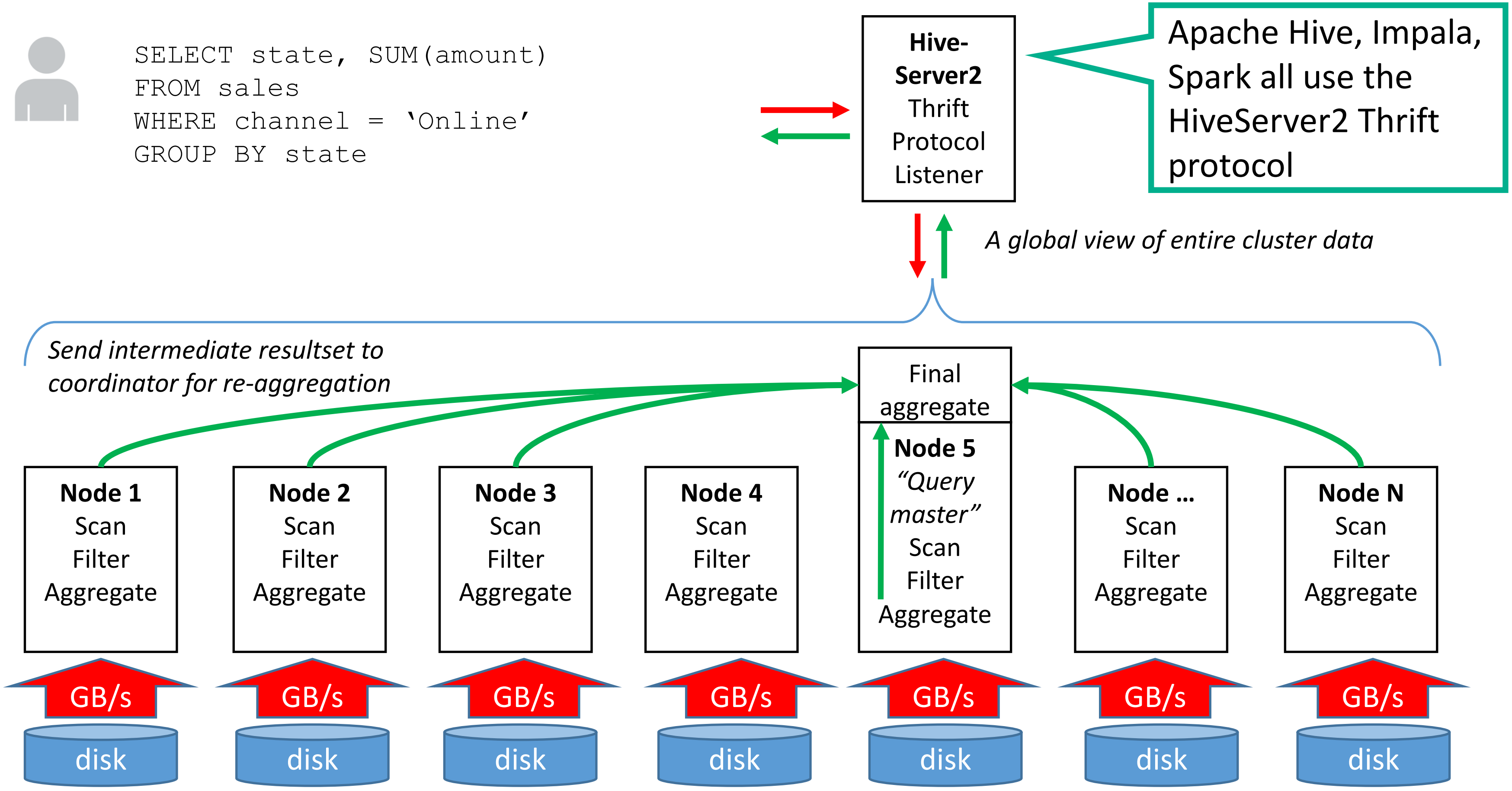
Example syntax: Impala SQL Language

```
[WITH name AS (select_expression) [, ...] ]
SELECT
   [ALL | DISTINCT]
   [STRAIGHT_JOIN]
   expression [, expression ...]
FROM table_reference [, table_reference ...]
[[FULL | [LEFT | RIGHT] INNER | [LEFT | RIGHT] OUTER | [LEFT | RIGHT]
   SEMI | [LEFT | RIGHT] ANTI | CROSS]
   JOIN table_reference [ON join_equality_clauses | USING (col1[, col2
   ...]] ...
WHERE conditions
GROUP BY { column | expression [ASC | DESC] [NULLS FIRST | NULLS LAST]
   [, ...] }
HAVING conditions
GROUP BY { column | expression [ASC | DESC] [, ...] }
LIMIT expression [OFFSET expression]
[UNION [ALL] select_statement] ...]
```

SQL-like syntax, very familiar to relational database developers!

SQL Execution on Hadoop: MPP Distributed processing (ideal)

```
SELECT state, SUM(amount)
FROM sales
WHERE channel = 'Online'
GROUP BY state
```



Apache Hive, Impala, Spark all use the HiveServer2 Thrift protocol

A global view of entire cluster data

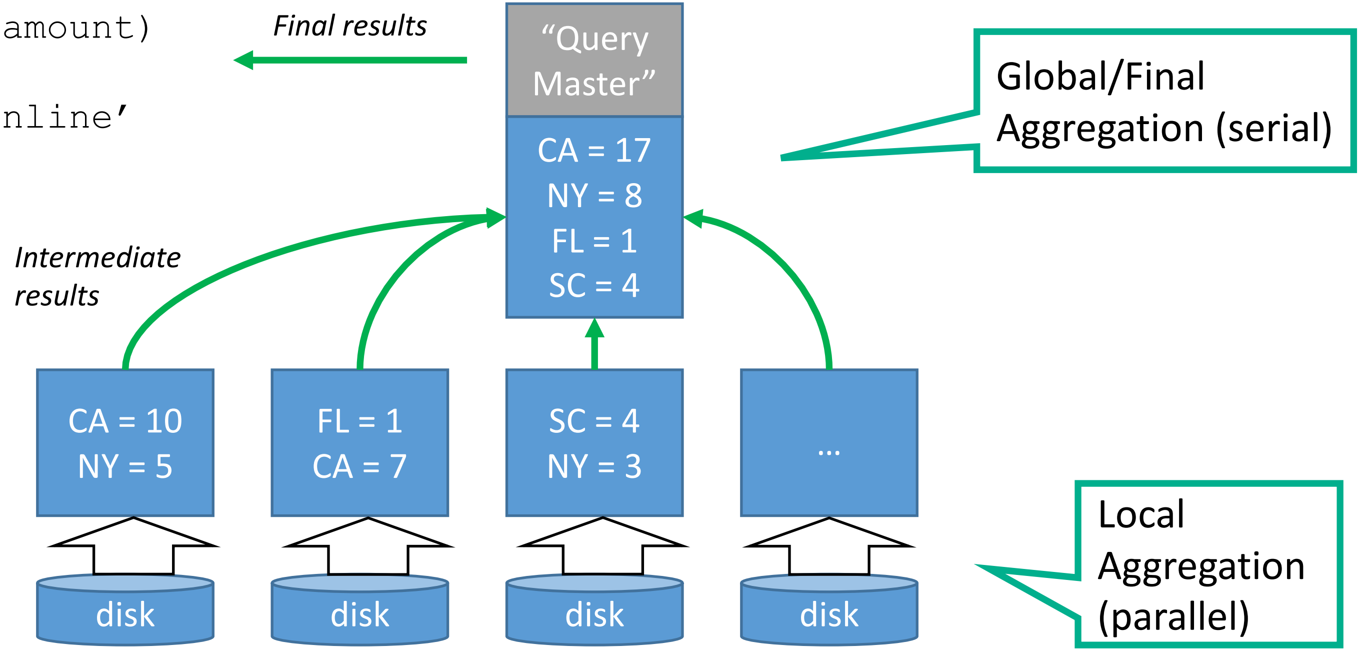
Send intermediate resultset to coordinator for re-aggregation

Hive-Server2 Thrift Protocol Listener
Final aggregate
Node 1 Scan Filter Aggregate
Node 2 Scan Filter Aggregate
Node 3 Scan Filter Aggregate
Node 4 Scan Filter Aggregate
Node 5 *"Query master"* Scan Filter Aggregate
Node … Scan Filter Aggregate
Node N Scan Filter Aggregate
GB/s
disk

Scalability – Additive Aggregations

- SUM, COUNT, AVG, MIN, MAX
  - Each cluster node can compute a SUM on their local data
  - Coordinator performs a final global SUM (by summing together local SUMs)

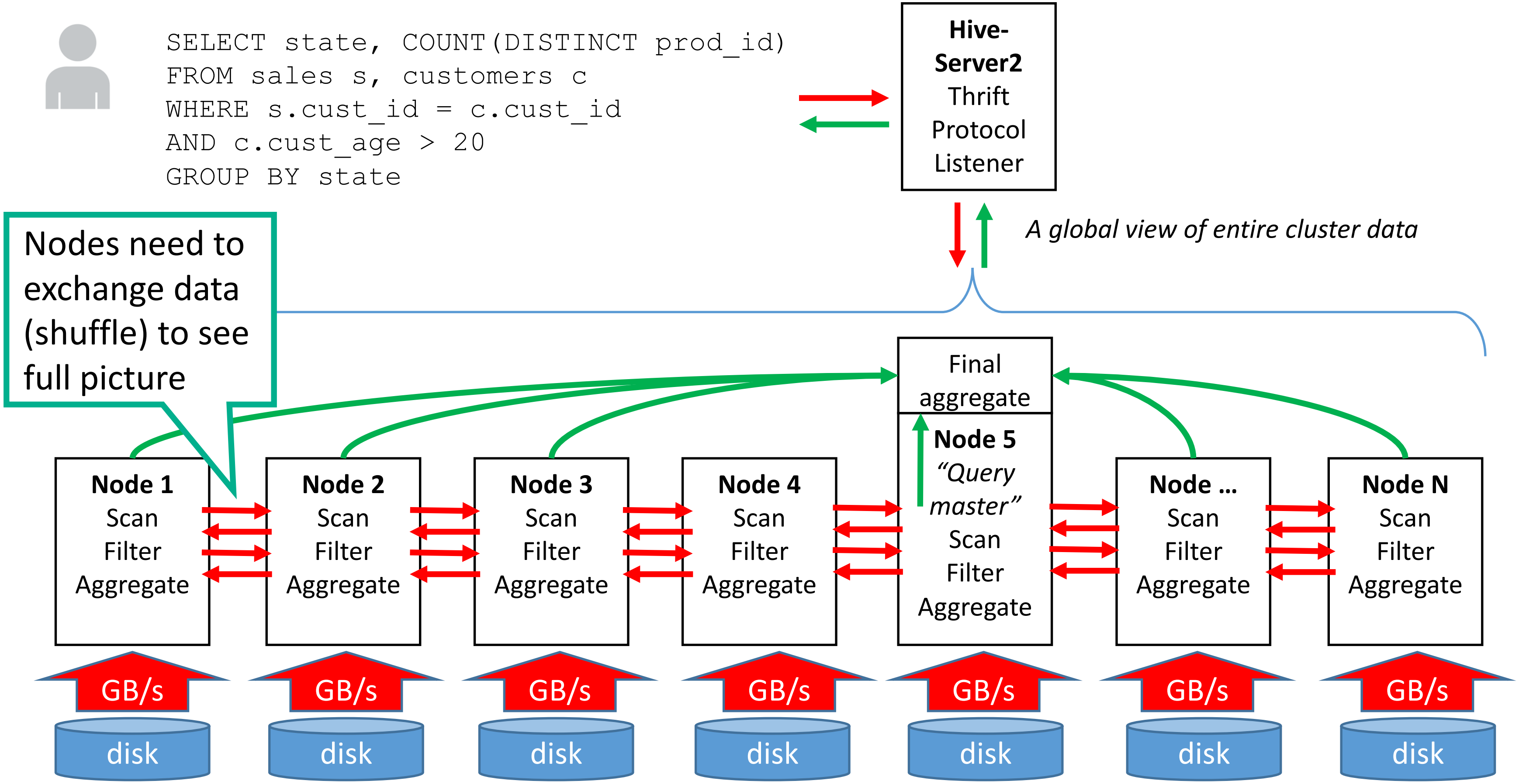
```
SELECT state, SUM(amount)
FROM sales
WHERE channel = ‘Online’
GROUP BY state
```



gluent.

SQL Execution on Hadoop: MPP Distributed processing (big queries)

```
SELECT state, COUNT(DISTINCT prod_id)
FROM sales s, customers c
WHERE s.cust_id = c.cust_id
AND c.cust_age > 20
GROUP BY state
```



Hive-Server2 Thrift Protocol Listener

A global view of entire cluster data

Nodes need to exchange data (shuffle) to see full picture

Final aggregate

Node 1 Scan Filter Aggregate

Node 2 Scan Filter Aggregate

Node 3 Scan Filter Aggregate

Node 4 Scan Filter Aggregate

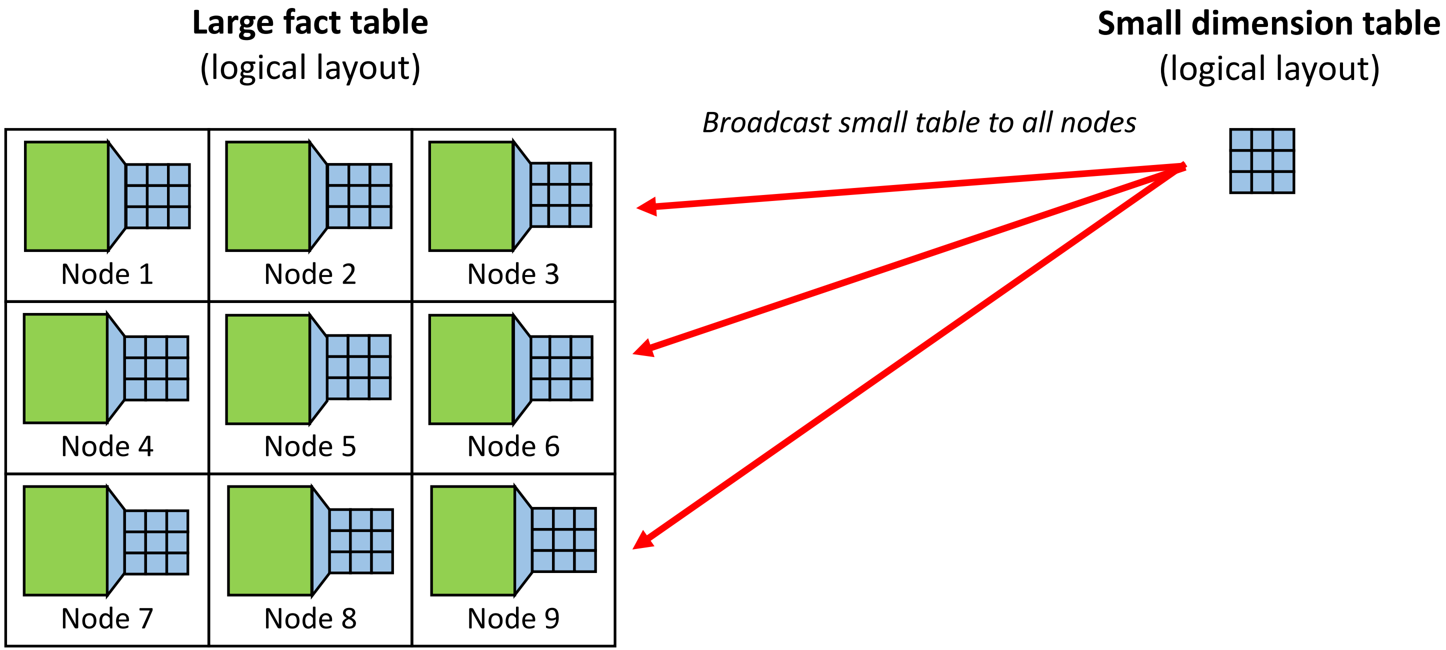
Node 5 *"Query master"* Scan Filter Aggregate

Node … Scan Filter Aggregate

Node N Scan Filter Aggregate

Scalability – Broadcast Joins

- In a distributed system with “random” physical placement of data in shards a join must be able to compare all data from both its sides (after any direct filters)
- If one dataset is small enough, it can be *broadcasted* and kept in memory on all nodes for node-local joining (broadcast join or replicated join)

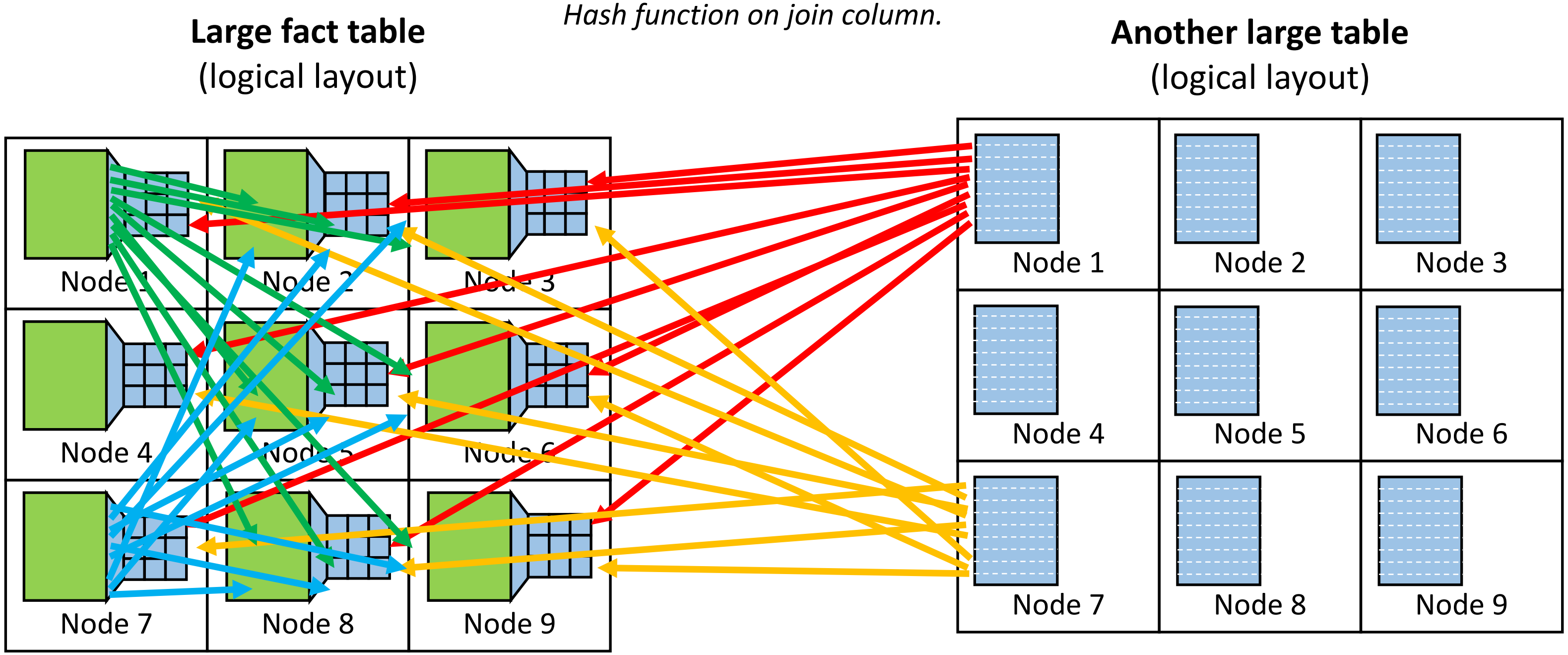


Scalability – Large Fact-to-Fact Joins

- If both sides of the join are too big for broadcasting & keeping in each node's memory

Hash & Shuffle non-overlapping subsets of data to the other join side.

Hash function on join column.



How to improve/combat/fix these issues

- Pre-join large fact tables (that are frequently joined)
  - Denormalization!
  - Hadoop storage is cheap – it's affordable to keep multiple copies
  - Columnar data formats
    - Queries access & scan only narrow slices of the wide table
    - De-normalized duplicate values compressed well thanks to columnar compression

- Approximate Distinct operations
  - HyperLogLog – COUNT(distinct) vs NDV()

- Partition-wise joins help with memory usage
  - Hive calls it bucketed map join (SMB map join) – shuffling still will happen
  - On Impala, currently the SQL query must join individual partitions (union all)

Apache Impala

Apache Impala

- Provides high-performance, low-latency SQL queries
  - Enables interactive exploration and fine-tuning of analytic queries
- Multiple open data formats (Parquet, Avro, CSV, etc) and storage formats (HDFS Kudu, S3, etc) can be accessed via Impala
- Uses the **Hive metastore** – the “data dictionary” for Hadoop
  - Stores table structure metadata (columns, datatypes) and file location in HDFS
  - Serves two purposes: data abstraction and data discovery
- Impala uses the same metadata (Hive metastore), SQL syntax, drivers, and UI (Hue) as Hive

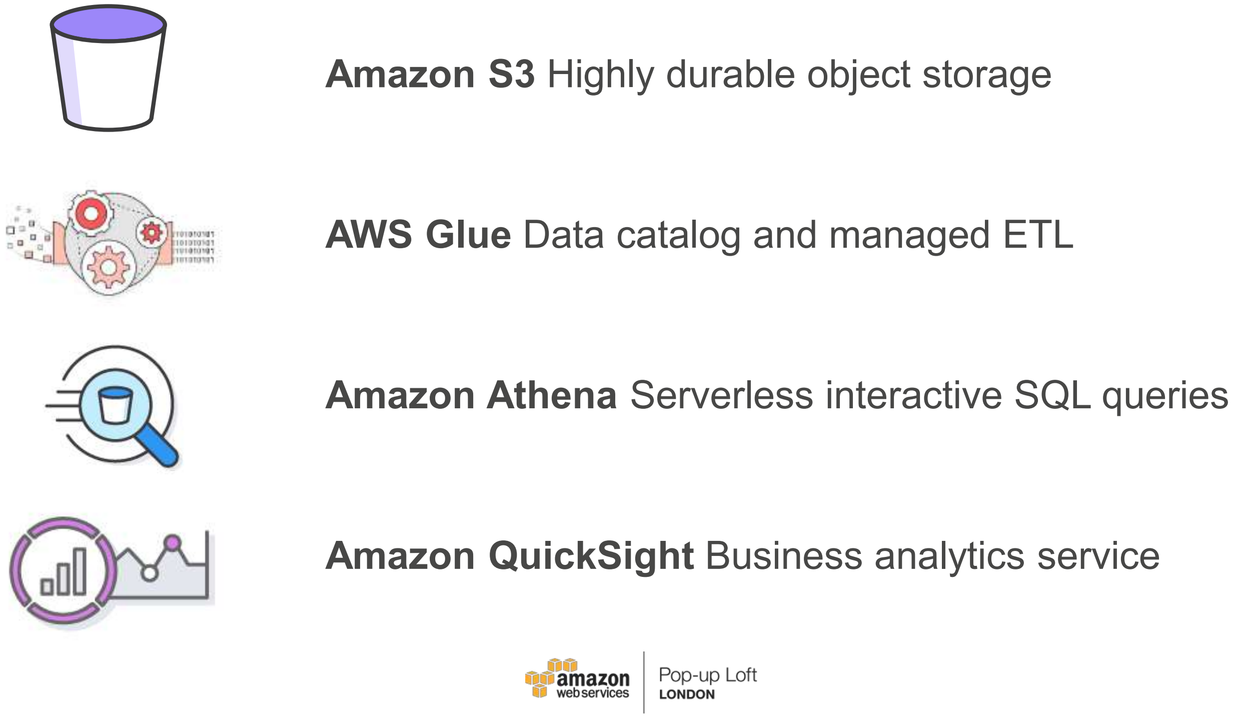
SQL on Hadoop processing engines compared

	Impala	Hive	Drill	Presto
Data sources	HDFS, Hbase, Kudu	HDFS, cloud storage, Hbase	HDFS, NoSQL, cloud storage	HDFS, NoSQL, RDBMS
Data model	Relational	Relational	Schema free JSON	Relational
Metadata	Hive Metastore	Hive Metastore	Hive Metastore (Optional)	Hive Metastore (remote)
Deployment model	Collocated with Hadoop	Collocated with Hadoop	Standalone or collocated with Hadoop	Collocated with Hadoop
SQL	HiveQL syntax	HiveQL syntax	ANSI SQL	ANSI SQL
Use case	Fast, interactive analytics. High concurrency. Large datasets.	Data warehouse / reporting analytics, data intensive, complex queries	Self-service, SQL based analytics, heterogeneous sources	Interactive analytics on large datasets

Going serverless

Serverless Analytics

- No servers to provision or manage
- Scales with usage
- Pay only when in use
- HA / fault tolerance built-in



Amazon S3 Highly durable object storage

AWS Glue Data catalog and managed ETL

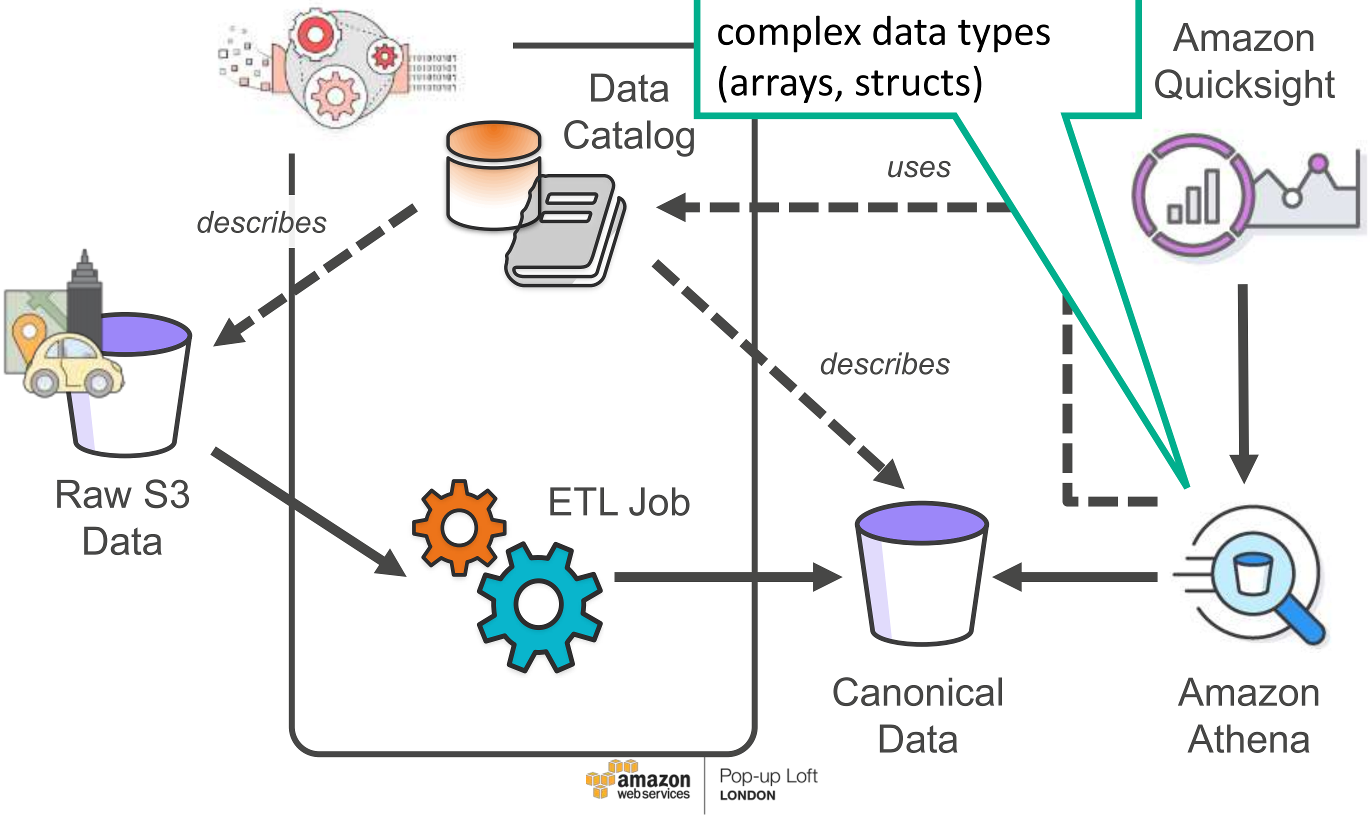
Amazon Athena Serverless interactive SQL queries

Amazon QuickSight Business analytics service

Source: https://www.slideshare.net/AmazonWebServices/serverlesss-big-data-analytics-with-amazon-athena-and-quicksight

Going serverless

Support for complex joins, nested queries & window functions and complex data types (arrays, structs)



Source: https://www.slideshare.net/AmazonWebServices/serverlesss-big-data-analytics-with-amazon-athena-and-quicksight

Where SQL on Hadoop can improve

- Large “fact to fact” table joins
- DML & Updates
- SQL engine sophistication & syntax richness

We RDBMS developers have been spoiled with very sophisticated SQL engines for years :)

```
hive> SELECT SUM(duration)
    > FROM call_detail_records
    > WHERE
    >      type = 'INTERNET‘
    > OR   phone_number IN ( SELECT phone_number
    >                        FROM customer_details
    >                        WHERE region = 'R04' );

FAILED: SemanticException [Error 10249]: Line 5:17
    Unsupported SubQuery Expression 'phone_number':
    Only SubQuery expressions that are top level
    conjuncts are allowed
```

Remember, Hadoop evolves very fast!

Is Hadoop going to replace my database?

- Transactional systems (OLTP)
  - Simple - it’s possible
  - Complex (ERP) - No (not anytime soon!)

- Data warehouse / Reporting / Analytics
  - Traditional DW - maybe, but it will take time
  - Big Data - Yes

- ETL and data integration
  - Probably!
  - Most ETL tools already support integration with Big Data transformation technologies

gluent.

Maintaining Hadoop

Backups

- RDBMS
  - Create full backup of database / log files (store to offsite location or tape)
  - Incremental / differential backups
  - Data replication to disaster recovery server

- Hadoop
  - Forget it and send data to multiple clusters
  - Backup the old way, use an agent to move data to tape
  - Copy / export HBase table (or HDFS files)
  - Backup to the cloud (like S3/Glacier). Use the cloud backup as data source for Hadoop

- HDFS / HBase snapshots
  - Create a snapshot of the metadata only
  - Snapshot points to file locations for recovery to a point in time
  - Export snapshots copy data and metadata for backup to another cluster

Keep Hadoop secure

- Yes, Hadoop is secure!
- Authentication
  - Authenticate via Kerberos, tied to your organizations LDAP
- Authorization
  - Role-based access control and policy management for files, folders, databases, tables, even columns
  - Apache Ranger or Apache Sentry (depends on your Hadoop distribution)
- Encryption
  - Data at-rest or in-transit
  - Transparent encryption seamlessly integrates with current applications

Myth #4: Hadoop is unsecure

- Authentication
  - Kerberos, LDAP (for SQL engines)
- Authorization
  - SQL engines: Role-Based Access Control (Sentry, Ranger)
  - HDFS: File System ACLs
- Encryption
  - Data at rest: HDFS Encryption (every directory can have its own keys)
  - Data in-flight: SSL/TLS

BUSTED!

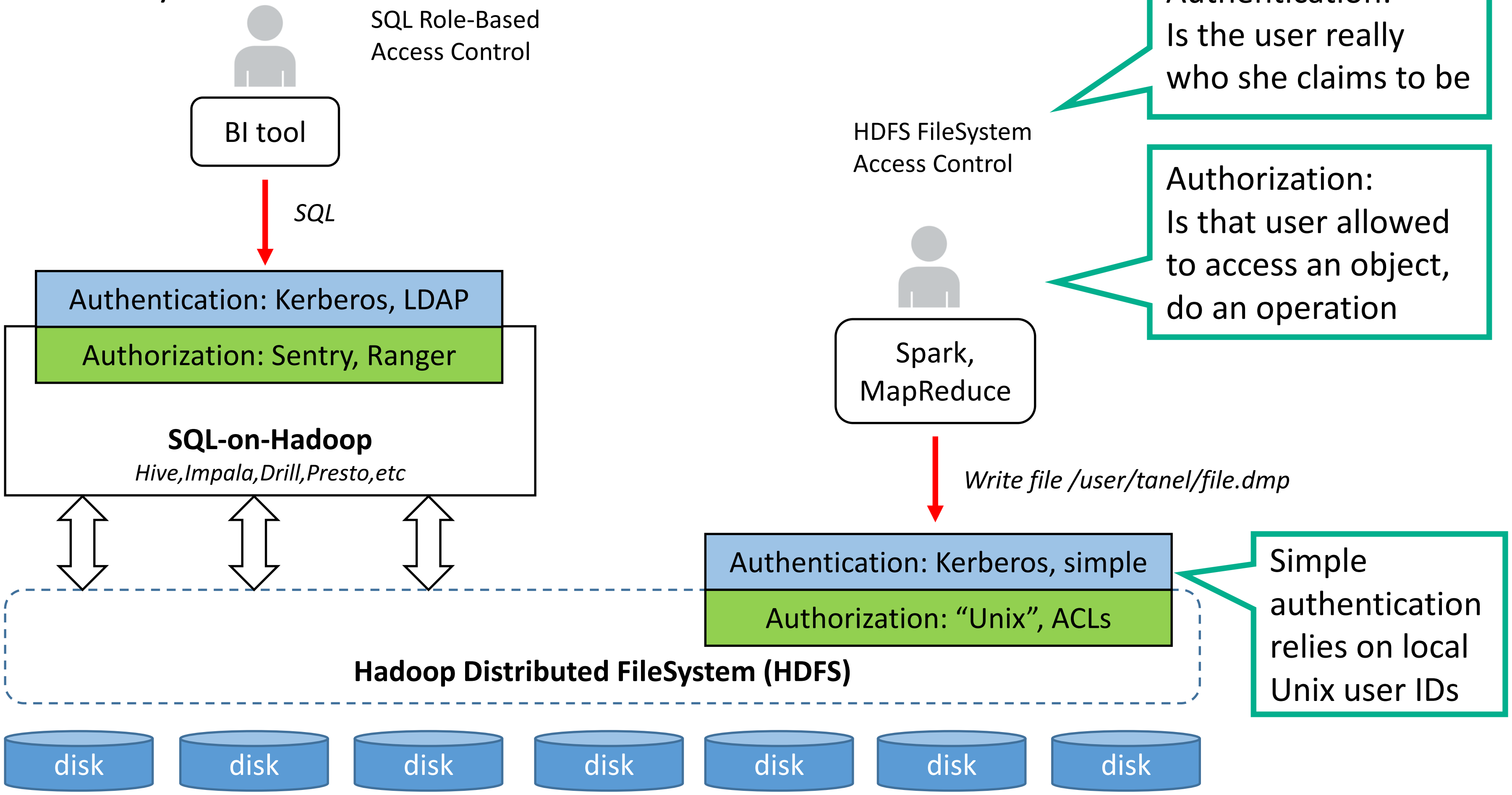
APACHE KNOX
Sentry
Apache Ranger

Hadoop security is enterprise ready

gluent.

26

Security


SQL Role-Based
Access Control
BI tool
SQL
Authentication: Kerberos, LDAP
Authorization: Sentry, Ranger
SQL-on-Hadoop
Hive,Impala,Drill,Presto,etc
HDFS FileSystem
Access Control
Spark,
MapReduce
Write file /user/tanel/file.dmp
Authentication: Kerberos, simple
Authorization: "Unix", ACLs
Hadoop Distributed FileSystem (HDFS)
disk
disk
disk
disk
disk
disk
disk

Authentication:
Is the user really
who she claims to be

Authorization:
Is that user allowed
to access an object,
do an operation

Simple authentication relies on local Unix user IDs

Security – HDFS Access Control

Unix-like file access permissions

```
$ hdfs dfs -ls /user
Found 9 items
drwxrwxrwx   - cloudera cloudera            0 2017-01-29 22:18 /user/cloudera
drwxr-xr-x   - cloudera supergroup          0 2016-08-29 16:19 /user/gluent
drwxr-xr-x   - mapred   hadoop              0 2015-11-18 13:03 /user/history
drwxrwxrwx   - hive     supergroup          0 2016-11-14 23:24 /user/hive
```

HDFS Access Control Lists (ACLs)

```
$ hdfs dfs -setfacl -m group:execs:r-- /sales-data

$ hdfs dfs -getfacl /sales-data
# file: /sales-data
# owner: bruce
# group: sales
user::rw-
group::r--
group:execs:r--
mask::r--
other::---

$ hdfs dfs -ls /sales-data
Found 1 items
-rw-r-----+  3 bruce sales          0 2014-03-04 16:31 /sales-data
```

Security – SQL Access Control

Familiar GRANT x ON y TO z commands

```
hive> grant select on table secured_table to role my_role;
No rows affected (0.046 seconds)

hive> revoke update, select on table secured_table from role my_role;
No rows affected (0.028 seconds)
```

Show granted privileges

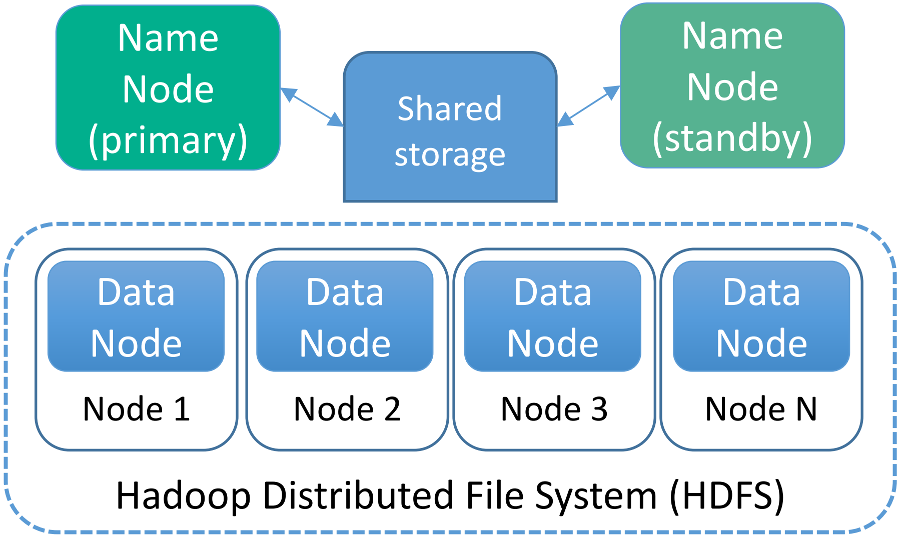
```
hive> show grant on table hivejiratable;
+-----------+----------------+---------+-----------------+-----------------+-----------+
| database  |     table      | column  | principal_name  | principal_type  | privilege |
+-----------+----------------+---------+-----------------+-----------------+-----------+
| default   | hivejiratable  |         | ashutosh        | USER            | DELETE    |
| default   | hivejiratable  |         | ashutosh        | USER            | SELECT    |
| default   | hivejiratable  |         | navis           | USER            | INSERT    |
| default   | hivejiratable  |         | navis           | USER            | SELECT    |
| default   | hivejiratable  |         | public          | ROLE            | SELECT    |
| default   | hivejiratable  |         | thejas          | USER            | DELETE    |
| default   | hivejiratable  |         | thejas          | USER            | INSERT    |
| default   | hivejiratable  |         | thejas          | USER            | SELECT    |
| default   | hivejiratable  |         | thejas                                              |
+-----------+----------------+---------+--------------                                 --+
```

Slightly different syntax possible in different SQL engines, but concepts are the same

gluent.

High availability

- HDFS DataNodes have built-in high availability (unlike most RDBMS)
  - Replication of each block to at least 3 nodes
- NameNode, on the other hand, is on a single server
  - Setup additional server as NameNode for active-passive HA
  - Use shared storage (NFS directory) or Journal Manager to keep edit logs in-sync
- Optional software for health monitoring and automatic failover



Recap - SQL Processing on Hadoop

- SQL processing on Hadoop has very familiar syntax - SQL!
  - Open formats and storage decoupled from compute allows for flexible, future proof data systems
  - Data security extends to SQL engines on Hadoop
- SQL engines on Hadoop cannot replace all RDBMS functionality (but they're getting closer)!
  - Tuning your query joins within Hadoop can help improve performance
- Maintenance and operations
  - Several options exist for backup and recovery
  - Data high availability is built-in, though still requires some additional HA setup
  - Each Hadoop distribution has a monitoring interface

Q&A

SQL Processing in Hadoop

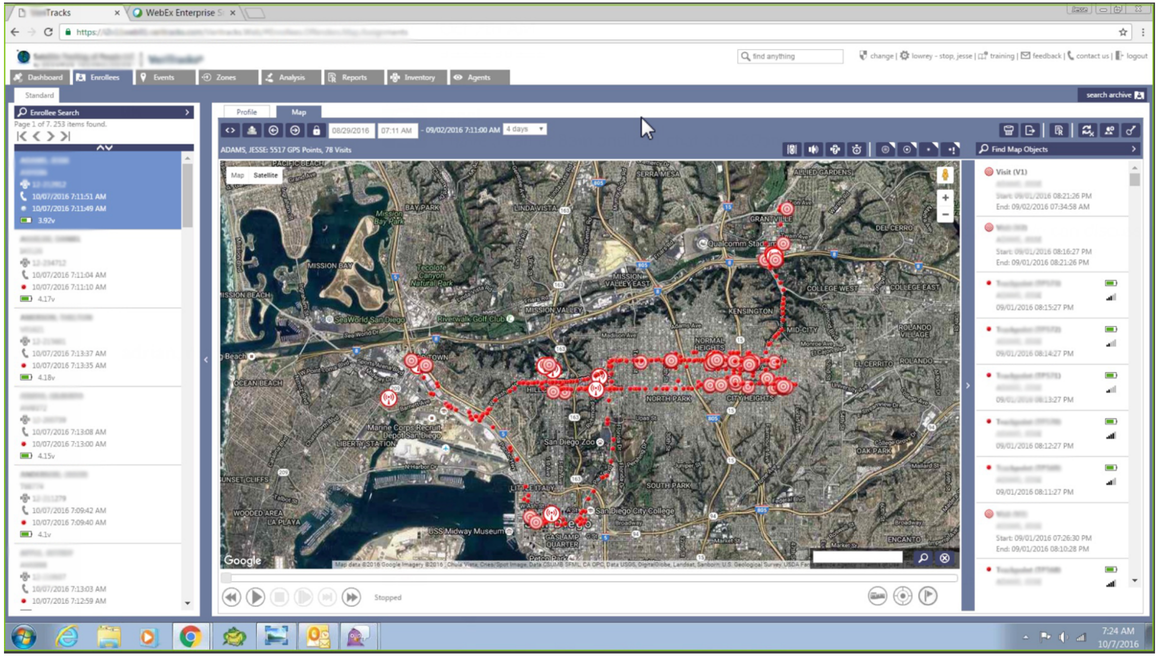
Hadoop in Action

Demo

The hybrid world

Securus – Enhance existing application capabilities with Hadoop

- Geospatial data (Google Maps API), used to track ankle bracelets
- Pseudo Real Time (few seconds delay)
- 150 TB with several years of historical data
- Database growing, too big and too slow
- New analytics & capabilities needed
- Rewriting the entire application out of question



Page 1 of 7. 253 items found.

08/29/2016 07:11 AM - 09/02/2016 7:11:00 AM 4 days

ADAMS, JESSE: 5517 GPS Points, 78 Visits

10/07/2016 7:11:51 AM
10/07/2016 7:11:49 AM
3.92v

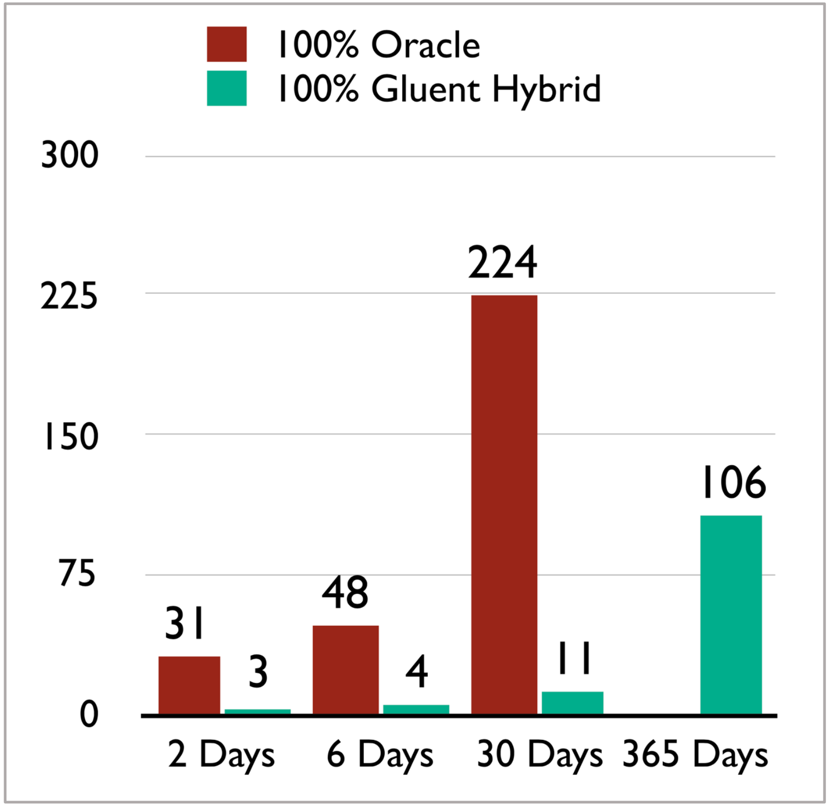
10/07/2016 7:11:04 AM
10/07/2016 7:11:10 AM
4.17v

Visit (V1)
End: 09/02/2016 07:34:58 AM

Source: https://gluent.com/products/case-studies/

Securus – Offload impact with Gluent Data Platform

- Max query of 2 days on previous architecture
- With hybrid architecture, multiple years possible
- Gluent data enrichment enables all applications to access all enterprise data at any time
  - Creates whole new investigative possibilities within the organization



Query response time comparison (in seconds) between datasets situated 100% in Oracle vs. 100% in Hadoop.

Note: 365 Day query in Oracle did not complete within designated time window.

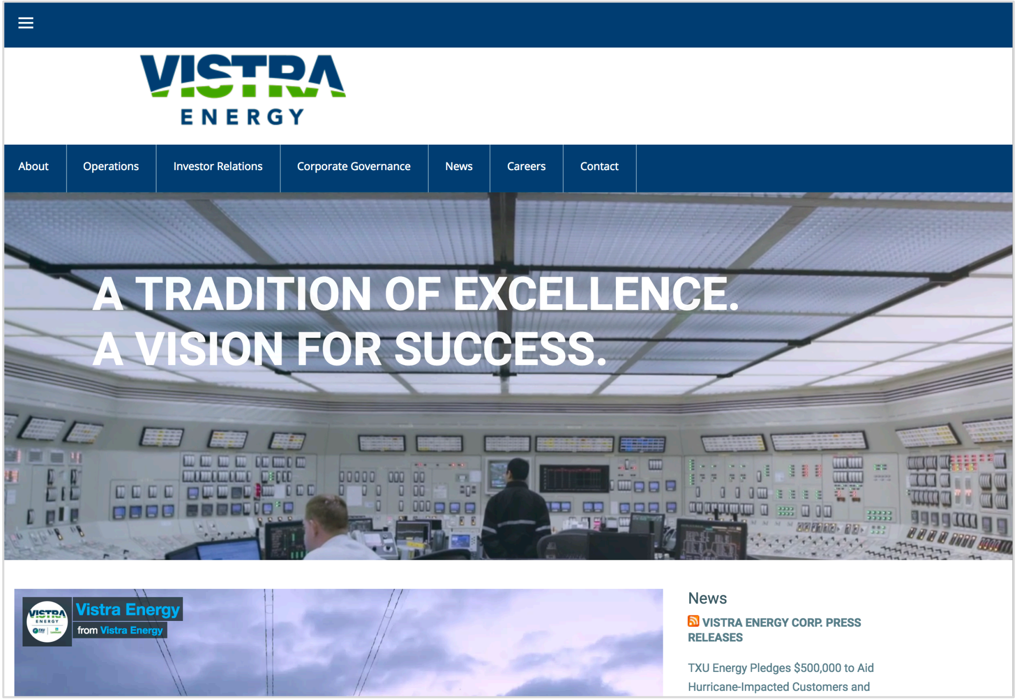
gluent.

Source: https://gluent.com/products/case-studies/

Vistra Energy - Data sharing enabled with Gluent

- Vistra Energy is the largest electricity generator and distributor in Texas

- Building out Data Lake as a primary source for analytics throughout organization
  - Many data sources, both RDBMS and Hadoop
  - Consolidation of data and data enrichment 2 key drivers



gluent.

Source: https://gluent.com/products/case-studies/

Benefits of Gluent

- Reduced DW storage costs (est. $220K savings)
  - Ability to control future growth
- Shift CPU resource usage from Oracle to Hadoop
- Expose large datasets in DW
  - Provide new insights about customer behavior (Customer 360º view)
  - Enable self-service analytics and visualizations using Tableau
- Offload historical DW data to Hadoop
  - Create custom analytical datasets using DW data and Hadoop data
  - Support development, execution, and retraining of predictive models

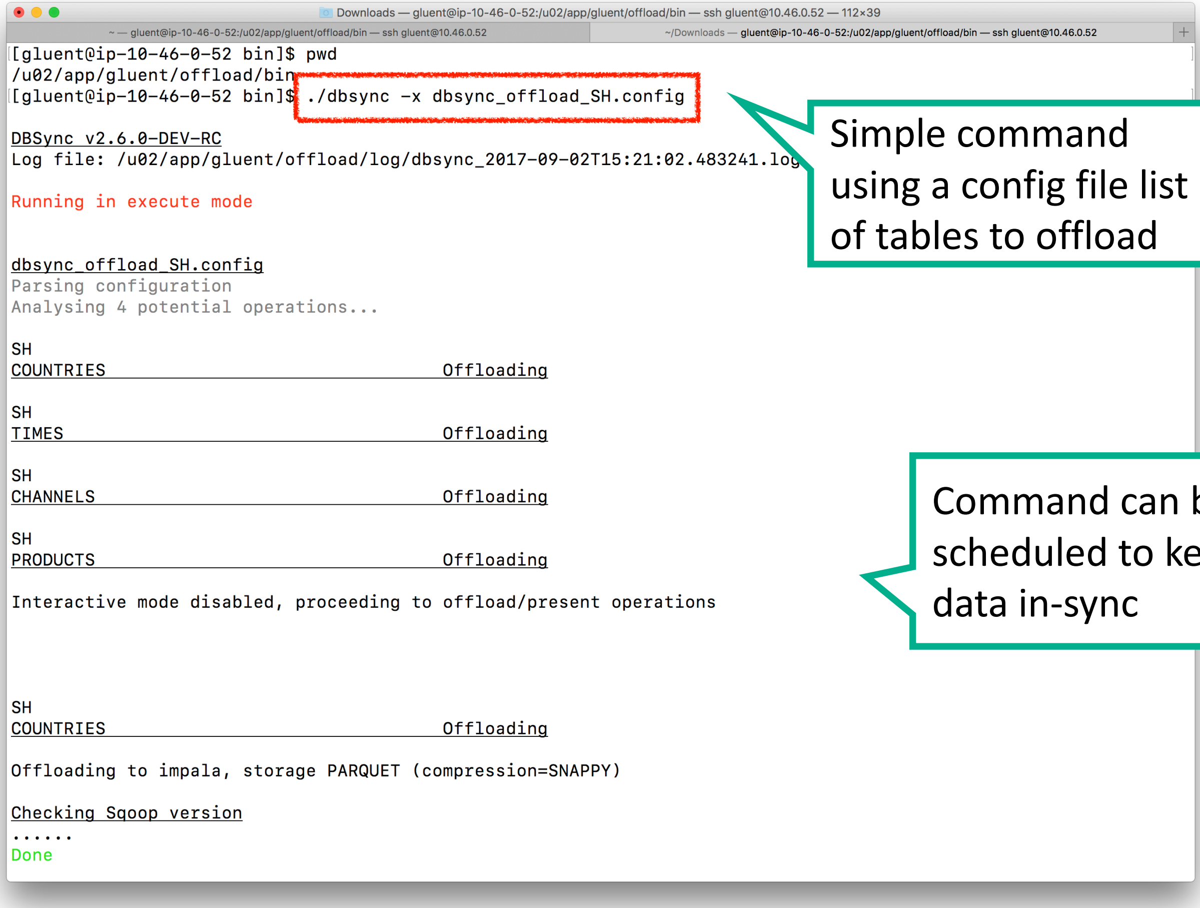
Source: https://gluent.com/products/case-studies/

Gluent in Action - Demo

Gluent in action - demo

- Enterprise data sharing using Gluent Present
- Enable advanced analytics with Gluent Present https://gluent.com/using-gluent-and-tableau-to-access-data-in-both-oracle-and-hadoop/
- Offload configuration via the Gluent web UI
- Offload multiple tables via command line
- Transparent query in a hybrid environment (video)

Gluent offload multiple tables


Downloads — gluent@ip-10-46-0-52:/u02/app/gluent/offload/bin — ssh gluent@10.46.0.52 — 112×39
~ — gluent@ip-10-46-0-52:/u02/app/gluent/offload/bin — ssh gluent@10.46.0.52
~/Downloads — gluent@ip-10-46-0-52:/u02/app/gluent/offload/bin — ssh gluent@10.46.0.52

```
[gluent@ip-10-46-0-52 bin]$ pwd
/u02/app/gluent/offload/bin
[gluent@ip-10-46-0-52 bin]$ ./dbsync -x dbsync_offload_SH.config

DBSync v2.6.0-DEV-RC
Log file: /u02/app/gluent/offload/log/dbsync_2017-09-02T15:21:02.483241.log

Running in execute mode


dbsync_offload_SH.config
Parsing configuration
Analysing 4 potential operations...

SH
COUNTRIES                                 Offloading

SH
TIMES                                     Offloading

SH
CHANNELS                                  Offloading

SH
PRODUCTS                                  Offloading

Interactive mode disabled, proceeding to offload/present operations




SH
COUNTRIES                                 Offloading

Offloading to impala, storage PARQUET (compression=SNAPPY)

Checking Sqoop version
......
Done
```

Simple command using a config file list of tables to offload

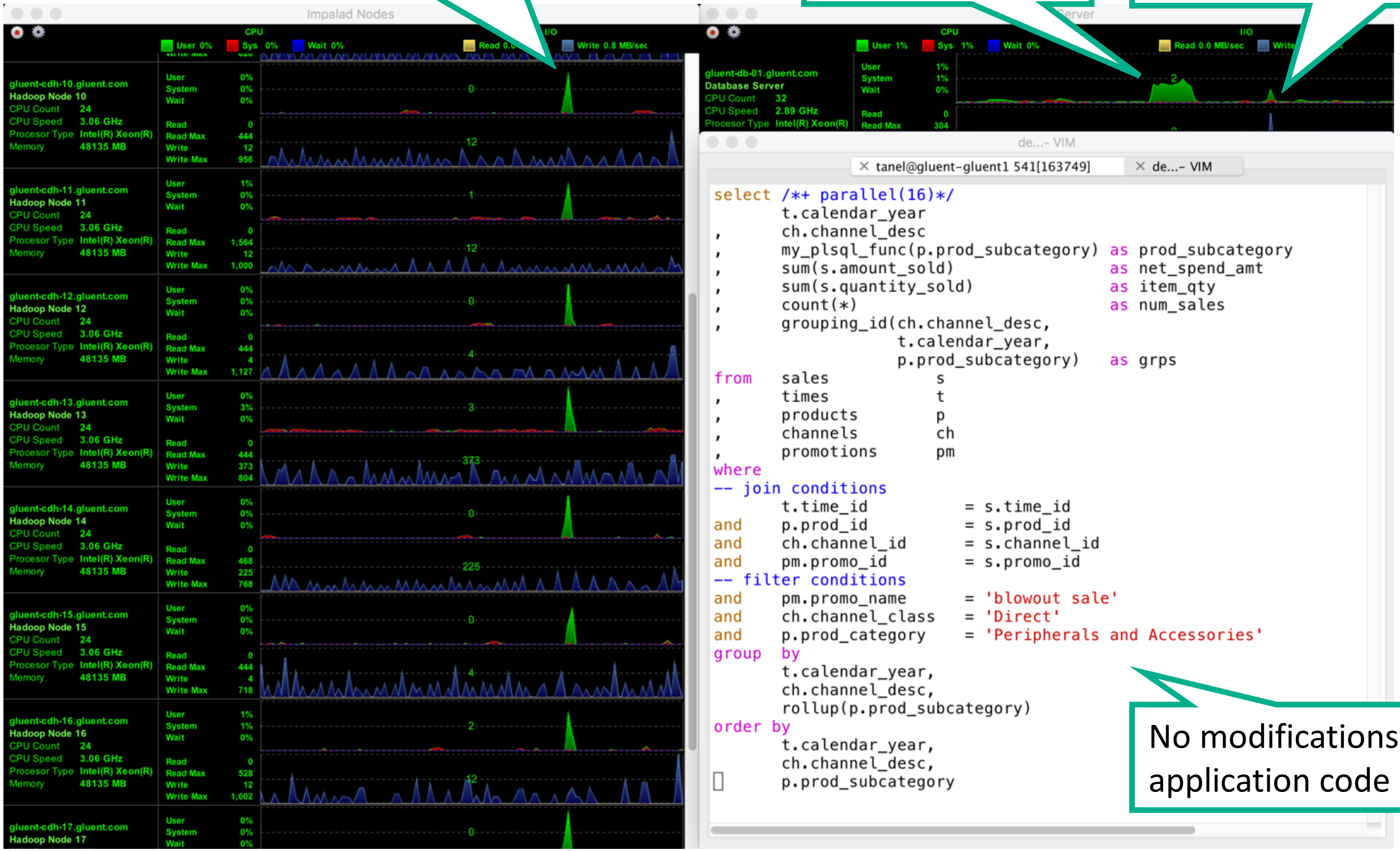
Command can be scheduled to keep data in-sync

Technical Demo

Gluent pushes SQL heavy-lifting to Hadoop

SQL query Oracle only - 10s lots of CPU

Same query with Gluent - 2s much less CPU



```
gluent-cdh-10.gluent.com      User      0%
Hadoop Node 10                System    0%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  444
                              Write     12
                              Write Max 956

gluent-cdh-11.gluent.com      User      1%
Hadoop Node 11                System    0%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  1,564
                              Write     12
                              Write Max 1,000

gluent-cdh-12.gluent.com      User      0%
Hadoop Node 12                System    0%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  444
                              Write     4
                              Write Max 1,127

gluent-cdh-13.gluent.com      User      0%
Hadoop Node 13                System    3%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  444
                              Write     373
                              Write Max 804

gluent-cdh-14.gluent.com      User      0%
Hadoop Node 14                System    0%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  468
                              Write     225
                              Write Max 768

gluent-cdh-15.gluent.com      User      0%
Hadoop Node 15                System    0%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  444
                              Write     4
                              Write Max 718

gluent-cdh-16.gluent.com      User      1%
Hadoop Node 16                System    1%
CPU Count      24             Wait      0%
CPU Speed      3.06 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
Memory         48135 MB       Read Max  528
                              Write     12
                              Write Max 1,002

gluent-cdh-17.gluent.com      User      0%
Hadoop Node 17                System    0%
                              Wait      0%
```

```
gluent-db-01.gluent.com       User      1%
Database Server               System    1%
CPU Count      32             Wait      0%
CPU Speed      2.89 GHz
Procesor Type  Intel(R) Xeon(R)  Read      0
                              Read Max  304
```

```sql
select /*+ parallel(16)*/
       t.calendar_year
,      ch.channel_desc
,      my_plsql_func(p.prod_subcategory) as prod_subcategory
,      sum(s.amount_sold)                as net_spend_amt
,      sum(s.quantity_sold)              as item_qty
,      count(*)                          as num_sales
,      grouping_id(ch.channel_desc,
                   t.calendar_year,
                   p.prod_subcategory)    as grps
from   sales             s
,      times             t
,      products          p
,      channels          ch
,      promotions        pm
where
-- join conditions
       t.time_id             = s.time_id
and    p.prod_id             = s.prod_id
and    ch.channel_id         = s.channel_id
and    pm.promo_id           = s.promo_id
-- filter conditions
and    pm.promo_name         = 'blowout sale'
and    ch.channel_class      = 'Direct'
and    p.prod_category       = 'Peripherals and Accessories'
group  by
       t.calendar_year,
       ch.channel_desc,
       rollup(p.prod_subcategory)
order  by
       t.calendar_year,
       ch.channel_desc,
       p.prod_subcategory
```

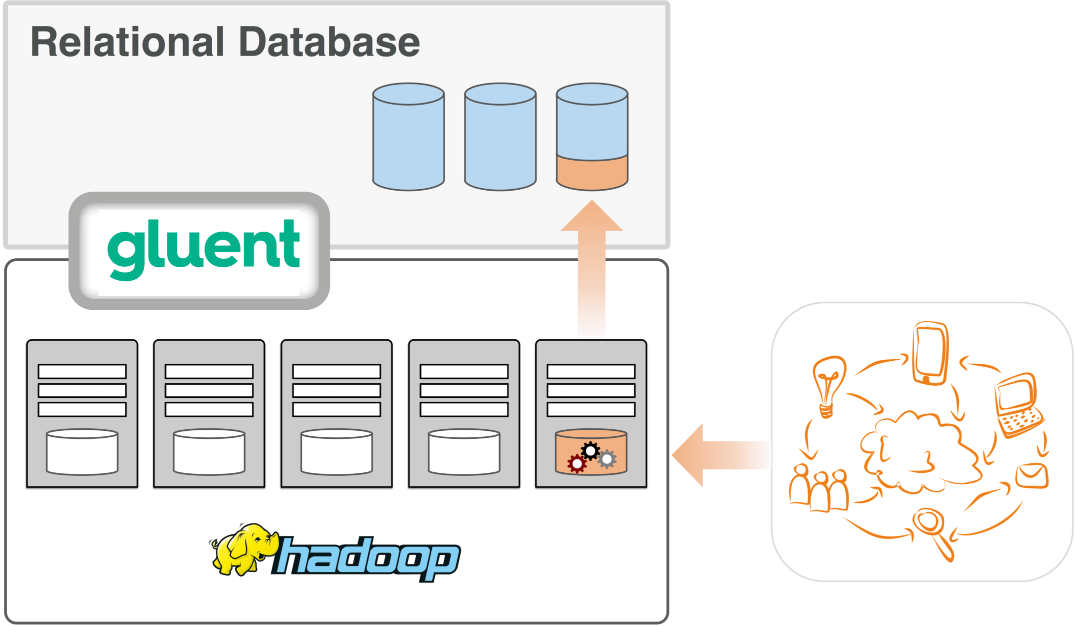
No modifications to application code

Gluent use cases

Gluent IoT Data Enrichment

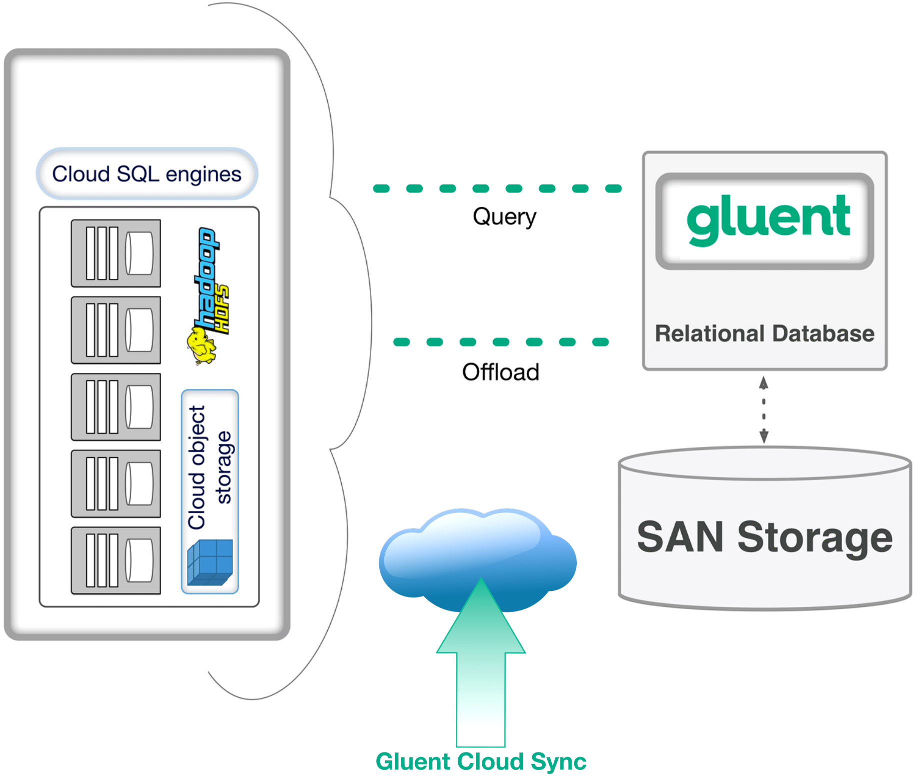
- Enhance relational data with tables from modern storage systems
- Gluent Present can share Hadoop native tables with the RDBMS
- No ETL necessary, just a simple command
- Data is available in the RBMS in real-time

All enterprise data is just a query away!

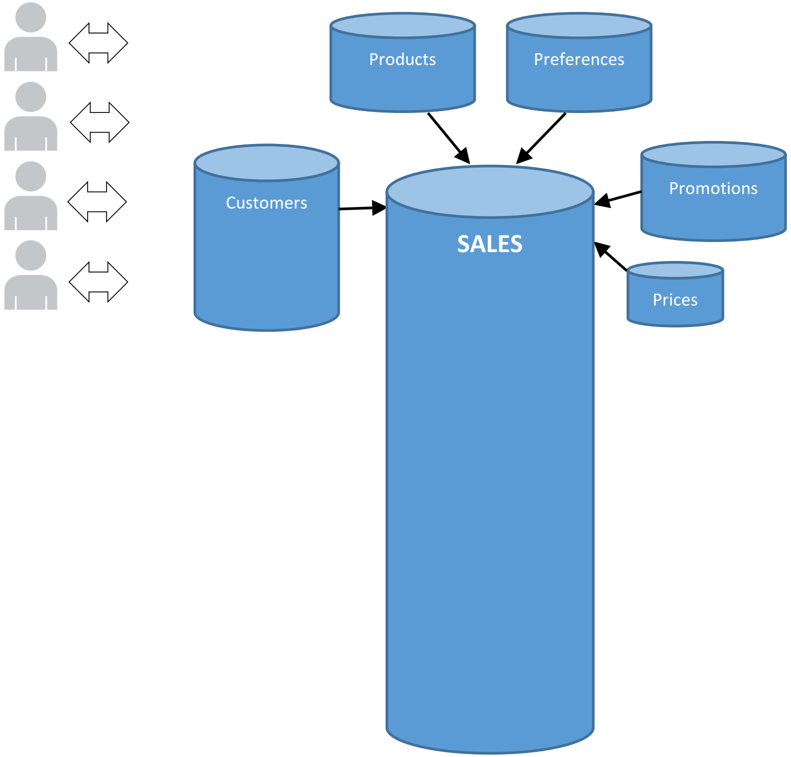


Gluent Cloud Migration and Data Sync

- Migrate relational databases to cloud storage
  - Incremental updates to keep data in-sync
  - No ETL or data streams
- Implement a hybrid cloud database
  - Small RDBMS implementation in the cloud
  - Hadoop backend does heavy lifting
- Sync Hadoop native tables to the cloud with Gluent Cloud Sync
  - Backup & restore, archive, or disaster recovery



Traditional relational data warehouse

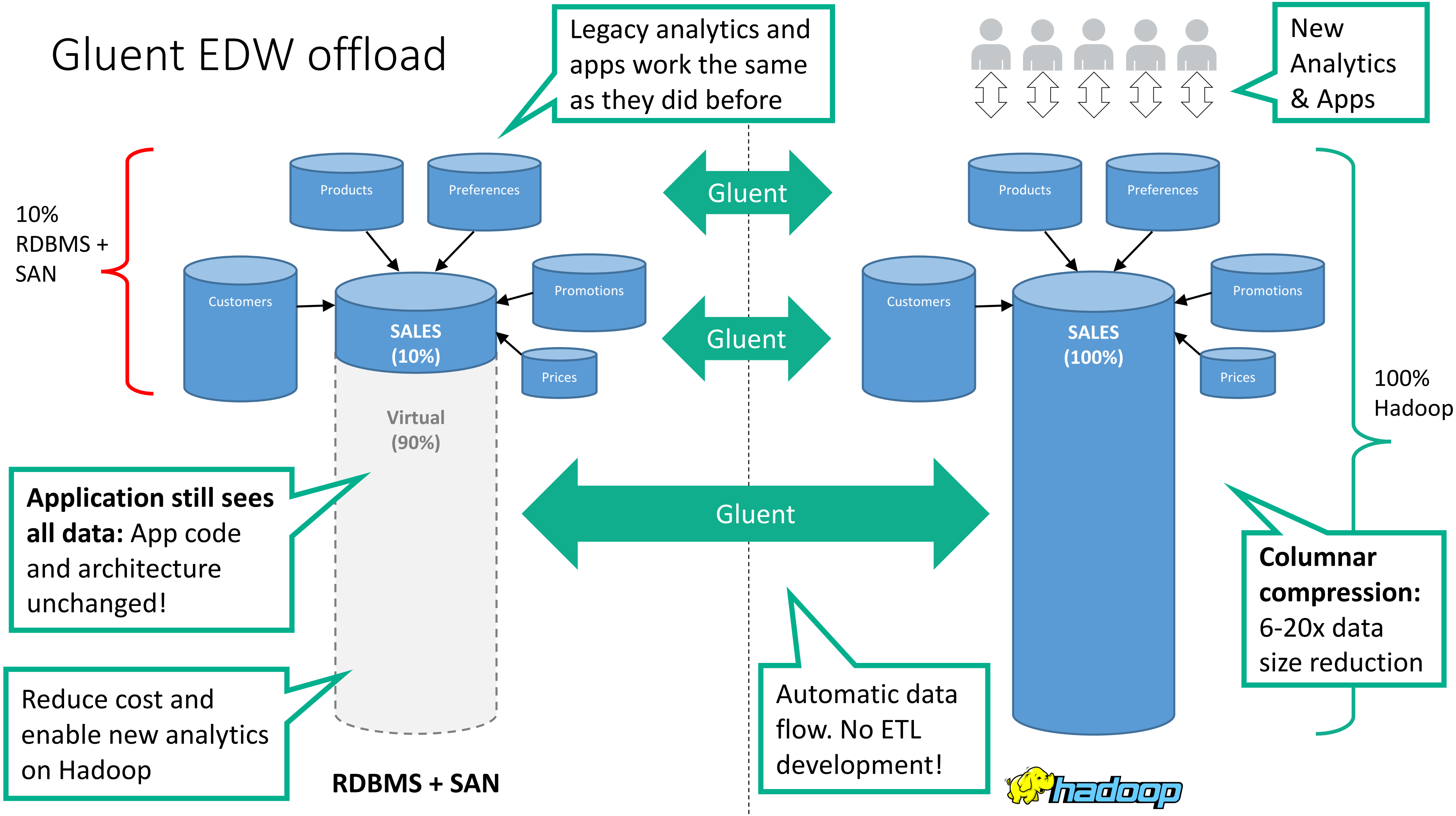


RDBMS + SAN

- ✓ A complex business application running on a RDBMS
- ✓ Years of application development & improvement
- ✓ Upstream & downstream dependencies
- ✓ Terabytes of historical data (usually years of history)
- ✗ Big queries run for too long or never complete (or never tried)
- ✗ Does not scale with modern demand
- ✗ Way too expensive
- ✗ **Application rewrite very costly & risky or virtually impossible**

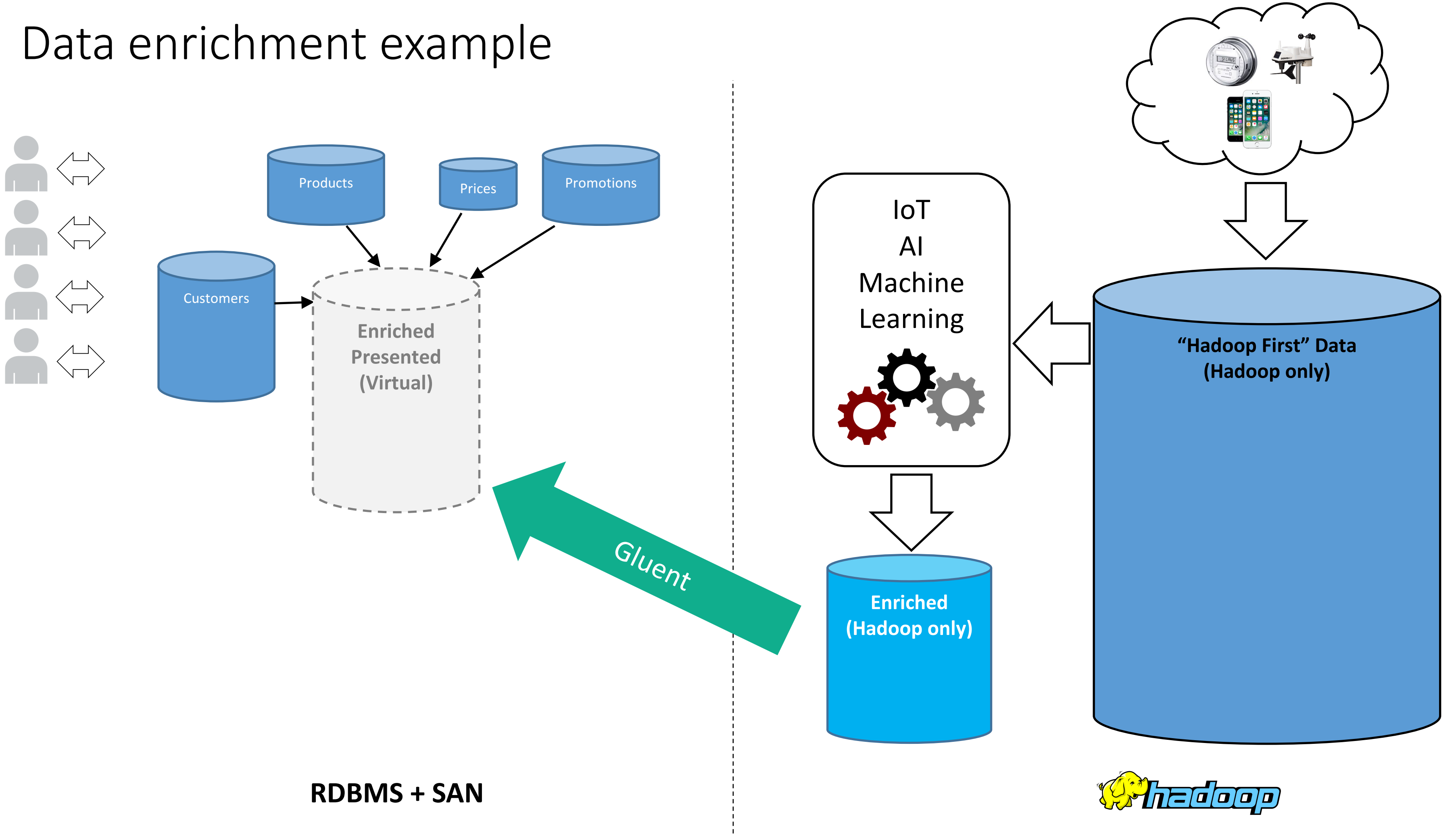
gluent.

Gluent EDW offload


Legacy analytics and apps work the same as they did before
New Analytics & Apps
10% RDBMS + SAN
Products
Preferences
Customers
Promotions
SALES (10%)
Prices
Virtual (90%)
Gluent
Gluent
Gluent
Products
Preferences
Customers
Promotions
SALES (100%)
Prices
100% Hadoop
Application still sees all data: App code and architecture unchanged!
Reduce cost and enable new analytics on Hadoop
RDBMS + SAN
Automatic data flow. No ETL development!
Columnar compression: 6-20x data size reduction
hadoop

gluent.

Data enrichment example



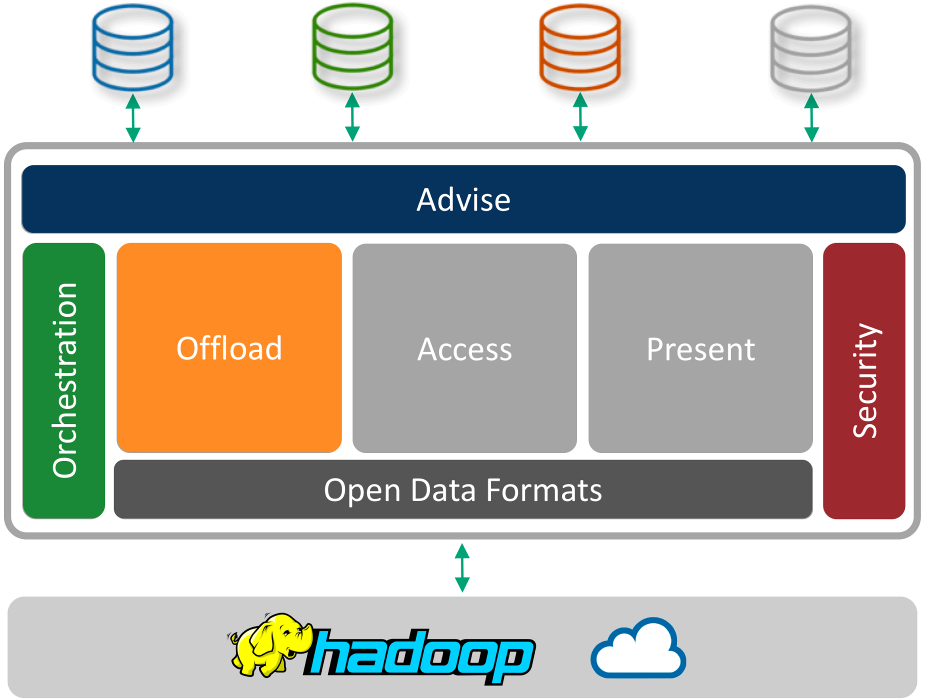
gluent.

Announcing Gluent Offload Engine!

- Announced as a standalone product this week!

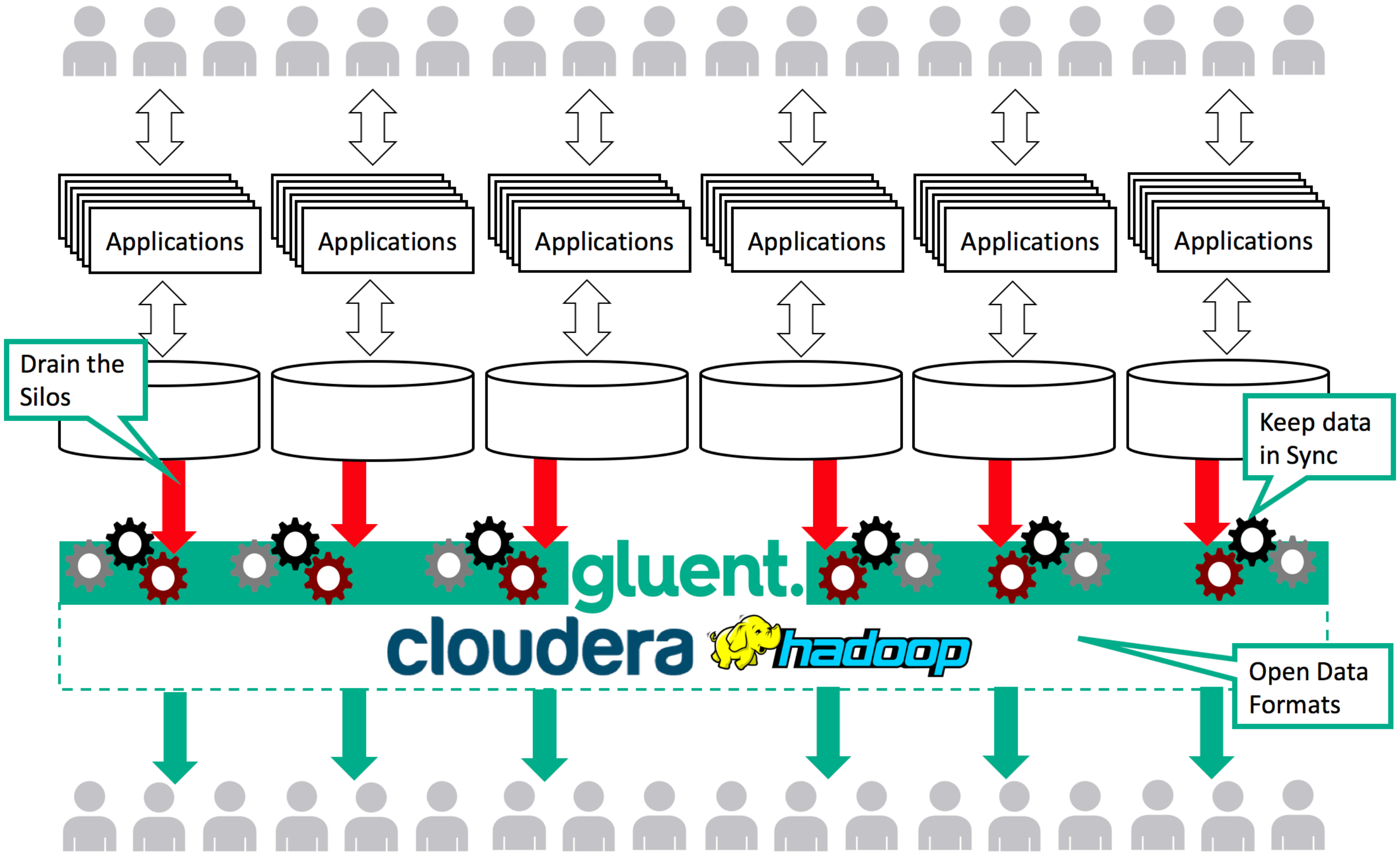
- GOE synchronizes tables from enterprise relational databases to modern data storage platforms, both on-premises and in the cloud

- Rapidly fill your data lake!
- Archive RDBMS tables to a more scalable, highly available (and less expensive) data store!



gluent.

Rapid data lake population



Drain the Silos

Keep data in Sync

Open Data Formats

Recap - Hadoop in Action

- Hadoop has been implemented for a variety of scenarios
  - Both for Big Data and traditional data

- Common requirements across the use cases
  - Large datasets
  - Fast analytics
  - Improved performance
  - Cost effectiveness
  - Innovation and discovering the previously undiscoverable

- Getting started and more information
  - http://gluent.com/hadoop

Q&A

gluent.

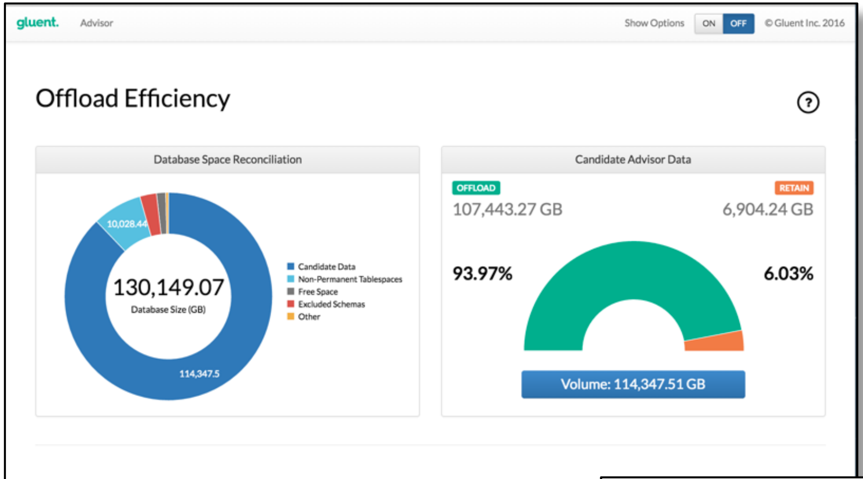
Hadoop for Database Professionals

Thank you for attending!

Gluent technical details

- Whitepapers: http://gluent.com/whitepapers
- 10 Minute Demo: https://vimeo.com/215254150
- Gluent Offload Engine Demos: https://gluent.com/products/gluent-data-platform/demo-offload/
- Getting started with Hadoop and more information
  - http://gluent.com/hadoop
- Email us! info@gluent.com

Gluent Advisor Challenge!



Offload Efficiency

Database Space Reconciliation

Candidate Advisor Data

Table Classification Summary

Partial Offload Candidates	Full Offload Candidates	Unable to Offload	Not Supported
6	132	6	30
OFFLOAD 7,753.33 GB RETAIN 6,287.91 GB	OFFLOAD 99,689.95 GB	RETAIN 310.67 GB	RETAIN 305.67

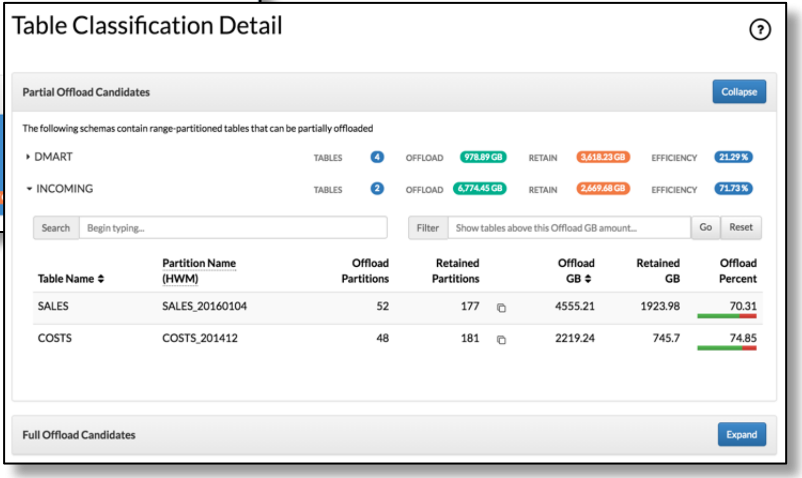


Table Classification Detail

Partial Offload Candidates Collapse

The following schemas contain range-partitioned tables that can be partially offloaded

▸ DMART	TABLES	4	OFFLOAD	978.89 GB	RETAIN	3,618.23 GB	EFFICIENCY	21.29 %
▾ INCOMING	TABLES	2	OFFLOAD	6,774.45 GB	RETAIN	2,669.68 GB	EFFICIENCY	71.73 %

Search: Begin typing... Filter: Show tables above this Offload GB amount... Go Reset

Table Name ⇕	Partition Name (HWM)	Offload Partitions	Retained Partitions	Offload GB ⇕	Retained GB	Offload Percent
SALES	SALES_20160104	52	177	4555.21	1923.98	70.31
COSTS	COSTS_201412	48	181	2219.24	745.7	74.85

Full Offload Candidates Expand

Download Gluent Advisor for free to assess potential savings and opportunities with Gluent Data Platform!

gluent.com/gluent_advisor_download