Управление производственным процессом разработки программного обеспечения

BigData

Хранение и обработка данных

- Online Transaction Processing (OLTP)

  RDBMS (Oracle, MySQL, Postgres, etc.)
- Online Analytical Processing (OLAP)

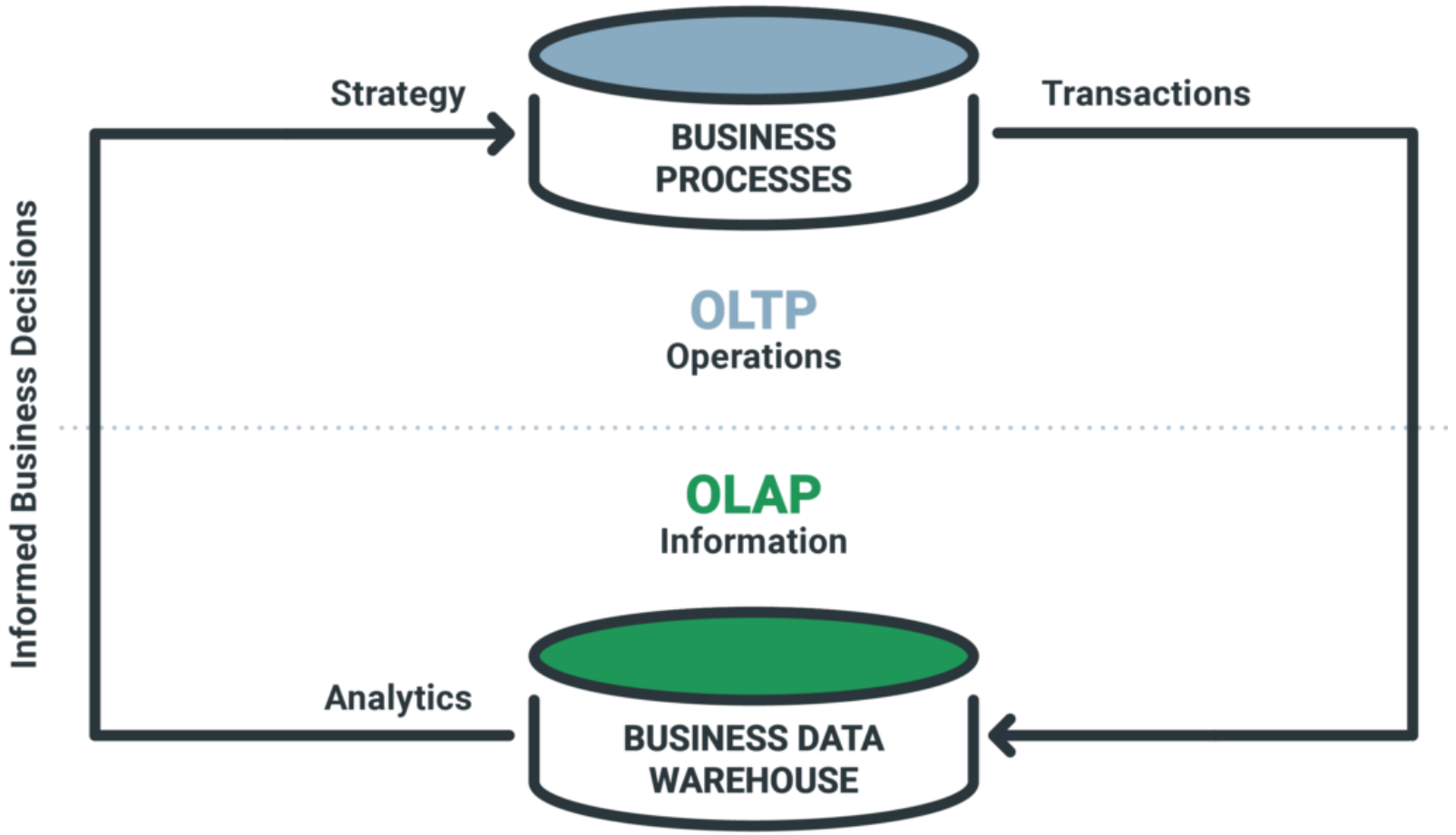
  Clickhouse, Snowflake, PrestoDB, etc.
- Key-value (NoSQL)

  Aerospike, Redis, Memcached, Hazelcast
- Persisted Queues

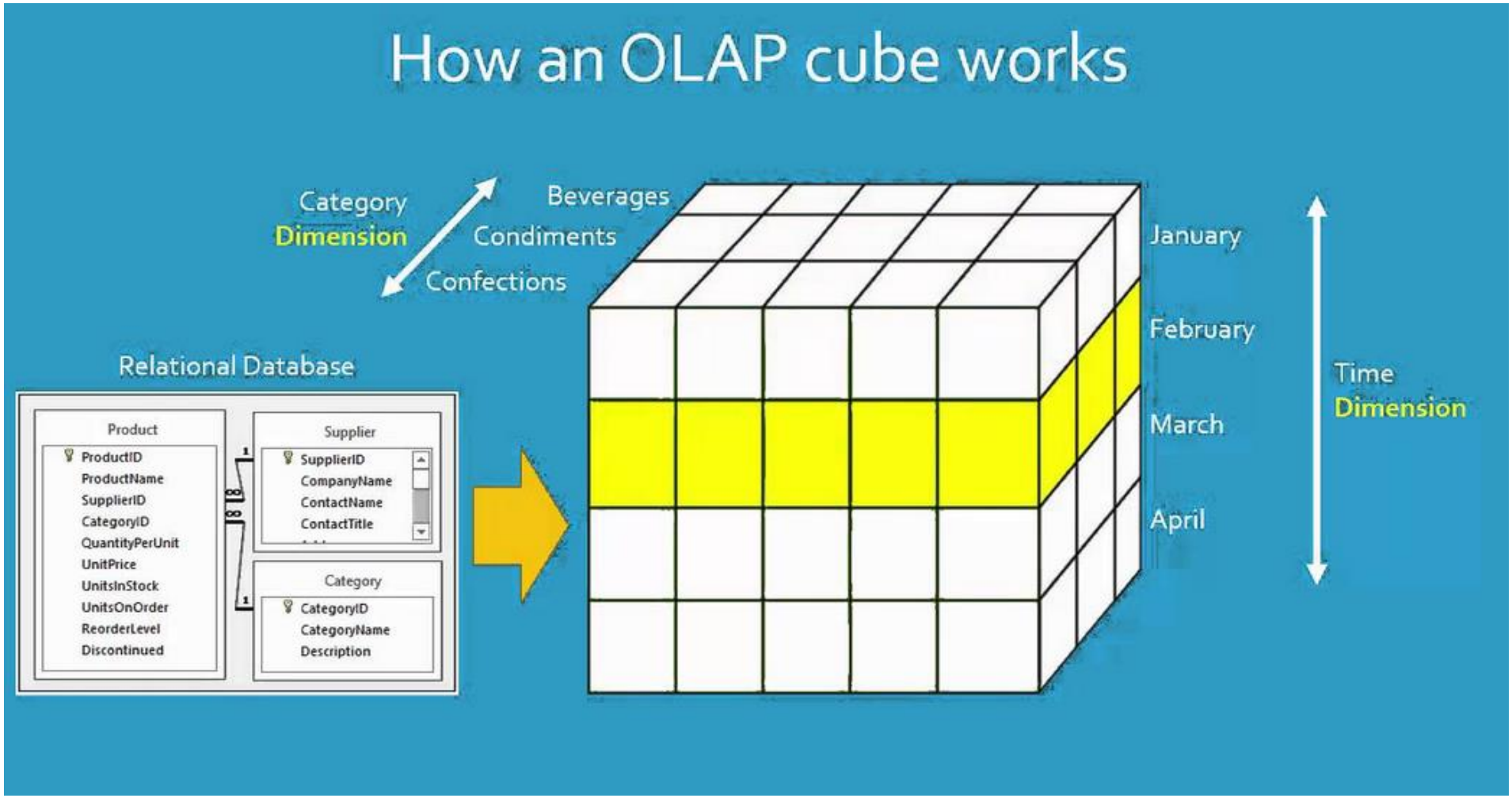
  Kafka, RabbitMQ
- Сетевое хранилище файлов/объектов

  NFS, AWS S3, MinIO, HDFS

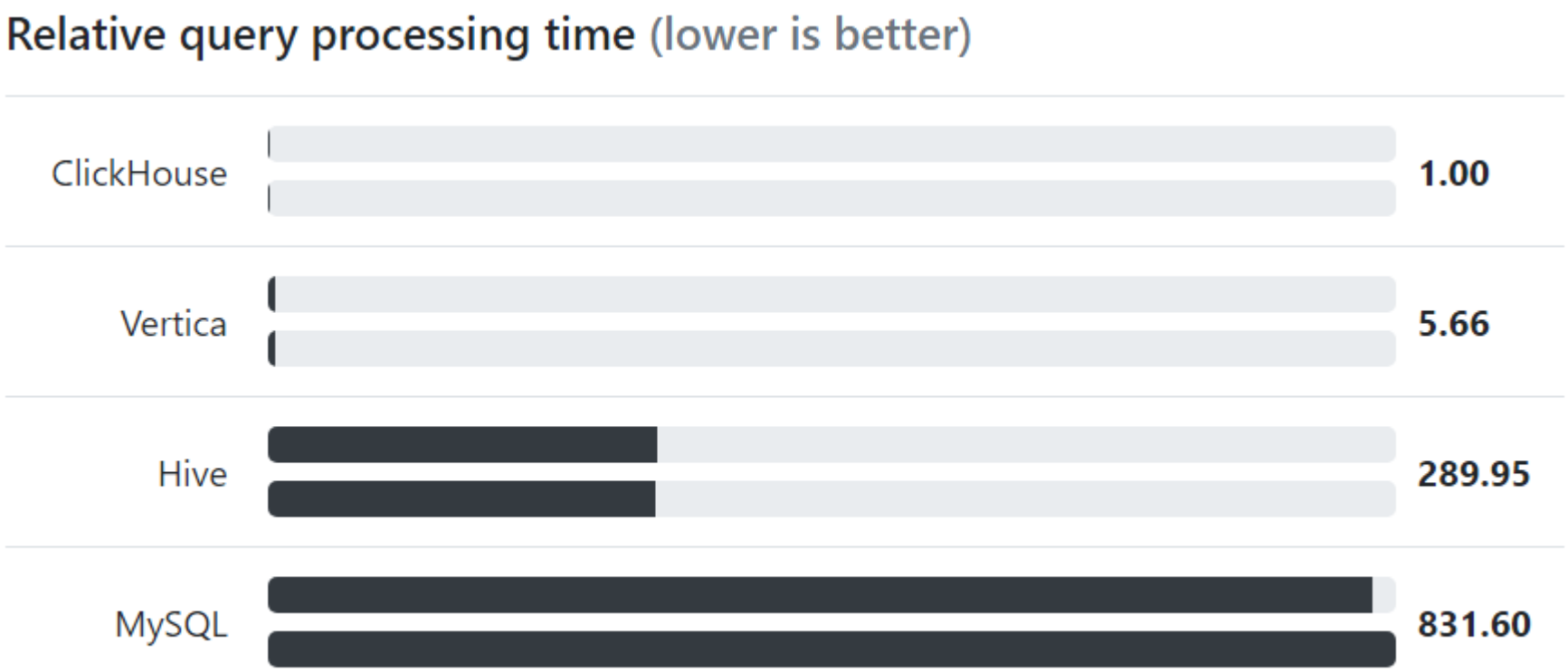
OLTP vs OLAP


Strategy
BUSINESS PROCESSES
Transactions
Informed Business Decisions
OLTP
Operations
OLAP
Information
Analytics
BUSINESS DATA WAREHOUSE

OLAP-куб


How an OLAP cube works
Category
Dimension
Beverages
Condiments
Confections
Relational Database
Product
ProductID
ProductName
SupplierID
CategoryID
QuantityPerUnit
UnitPrice
UnitsInStock
UnitsOnOrder
ReorderLevel
Discontinued
Supplier
SupplierID
CompanyName
ContactName
ContactTitle
Category
CategoryID
CategoryName
Description
January
February
March
April
Time
Dimension

OLTP vs OLAP



100,000,000 rows, various SELECT operations

Источник: https://clickhouse.tech/benchmark/dbms/

OLAP сценарий работы

- подавляющее большинство запросов - на чтение;
- данные обновляются достаточно большими пачками (> 1000 строк), а не по одной строке, или не обновляются вообще;
- данные добавляются в БД, но не изменяются;
- при чтении, вынимается достаточно большое количество строк из БД, но только небольшое подмножество столбцов;
- таблицы являются «широкими», то есть, содержат большое количество столбцов;
- запросы идут сравнительно редко (обычно не более сотни в секунду на сервер);
- при выполнении простых запросов, допустимы задержки в районе 50 мс;
- значения в столбцах достаточно мелкие - числа и небольшие
- требуется высокая пропускная способность при обработке одного запроса (до миллиардов строк в секунду на один сервер);
- транзакции отсутствуют;
- низкие требования к консистентности данных;
- в запросе одна большая таблица, все таблицы кроме одной маленькие;
- результат выполнения запроса существенно меньше исходных данных - то есть, данные фильтруются или агрегируются; результат выполнения помещается в оперативку на одном сервере.

OLAP решения

- **Надстройка над RDBMS**
  Microsoft SQL Server Analysis Services, Essbase, Oracle Database OLAP Option, Mondrian

- **Column-Oriented Databases**
  Yandex Clickhouse, Amazon Redshift, Vertica, Apache Druid, Snowflake

- **Distributed SQL query engine**
  Facebook PrestoDB, Apache Impala, Apache Hive

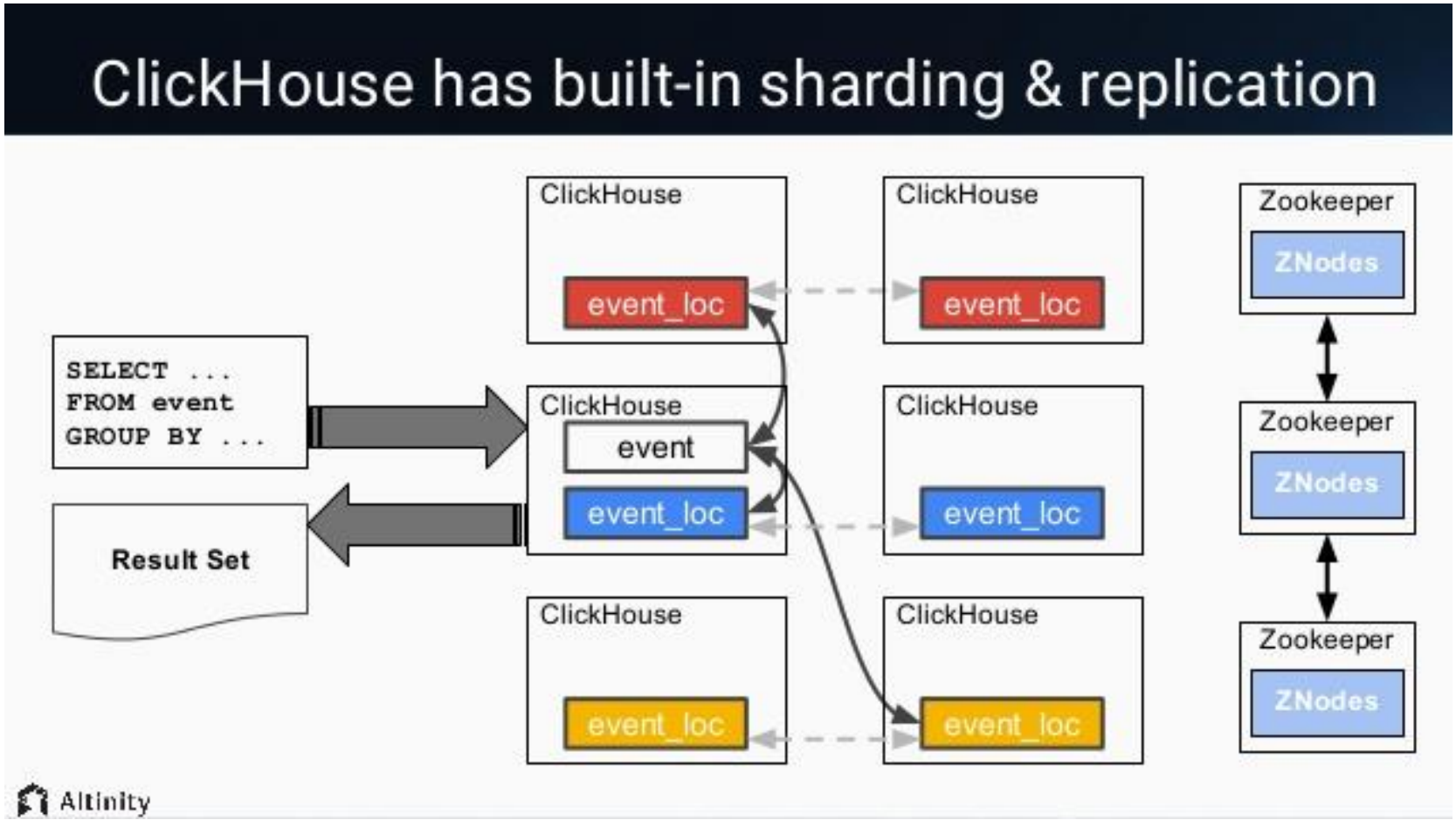
Column-Oriented Database

Column-Oriented vs Row-Oriented

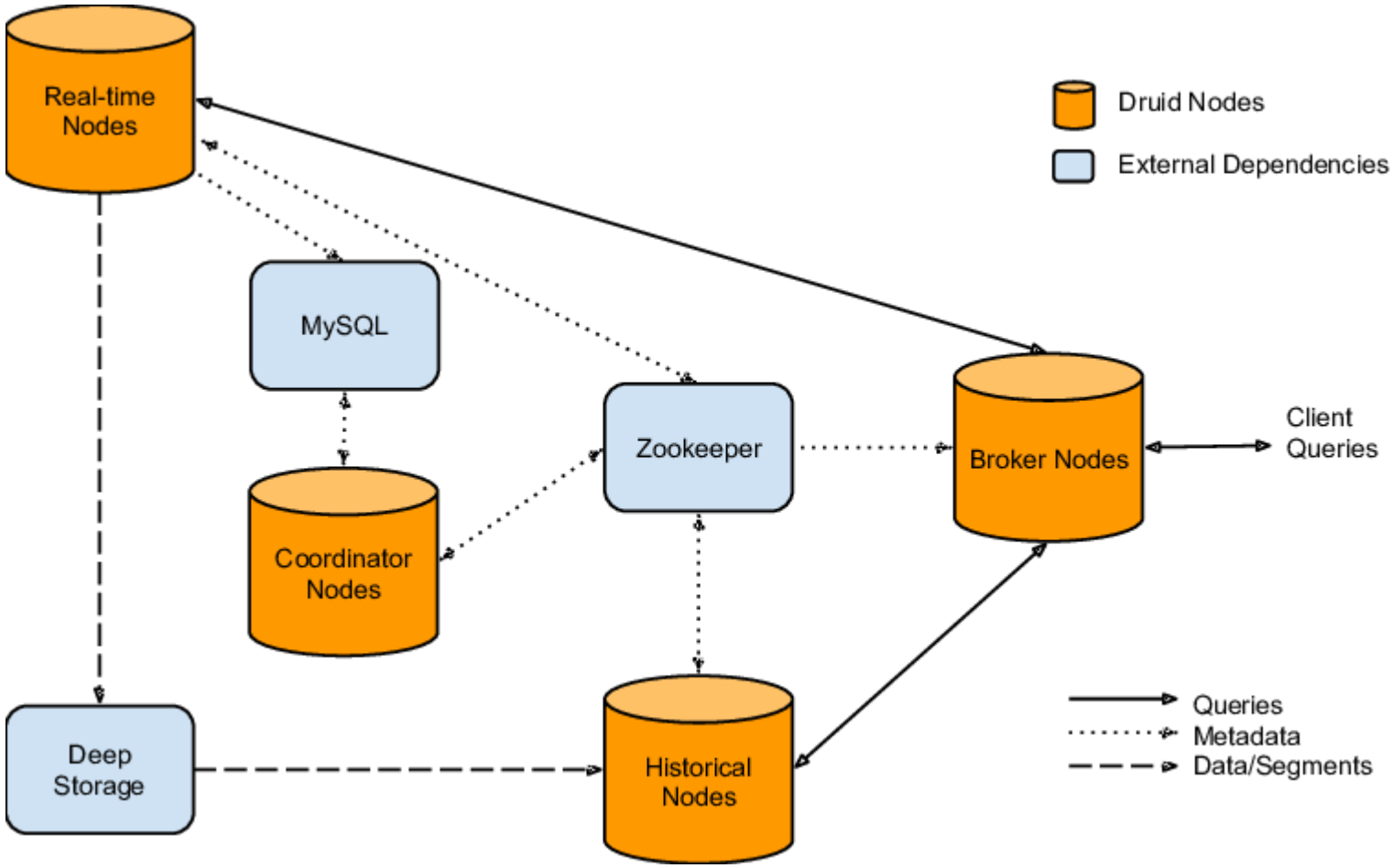
Yandex Clickhouse

- Используется для Yandex Metrics
- Open Source
- 100,000,000 – 1,000,000,000 строк в секунду на 1-й ноде
- 2Tb данных в секунду на кластере из 400 нод
- Простота развертки и настройки
- Поддерживает очень сложные SQL
- Гибкий импорт данных из разных источников: CSV files, Parquet files, RDBMS, Kafka и т.д.
- Поддержка модификации данных

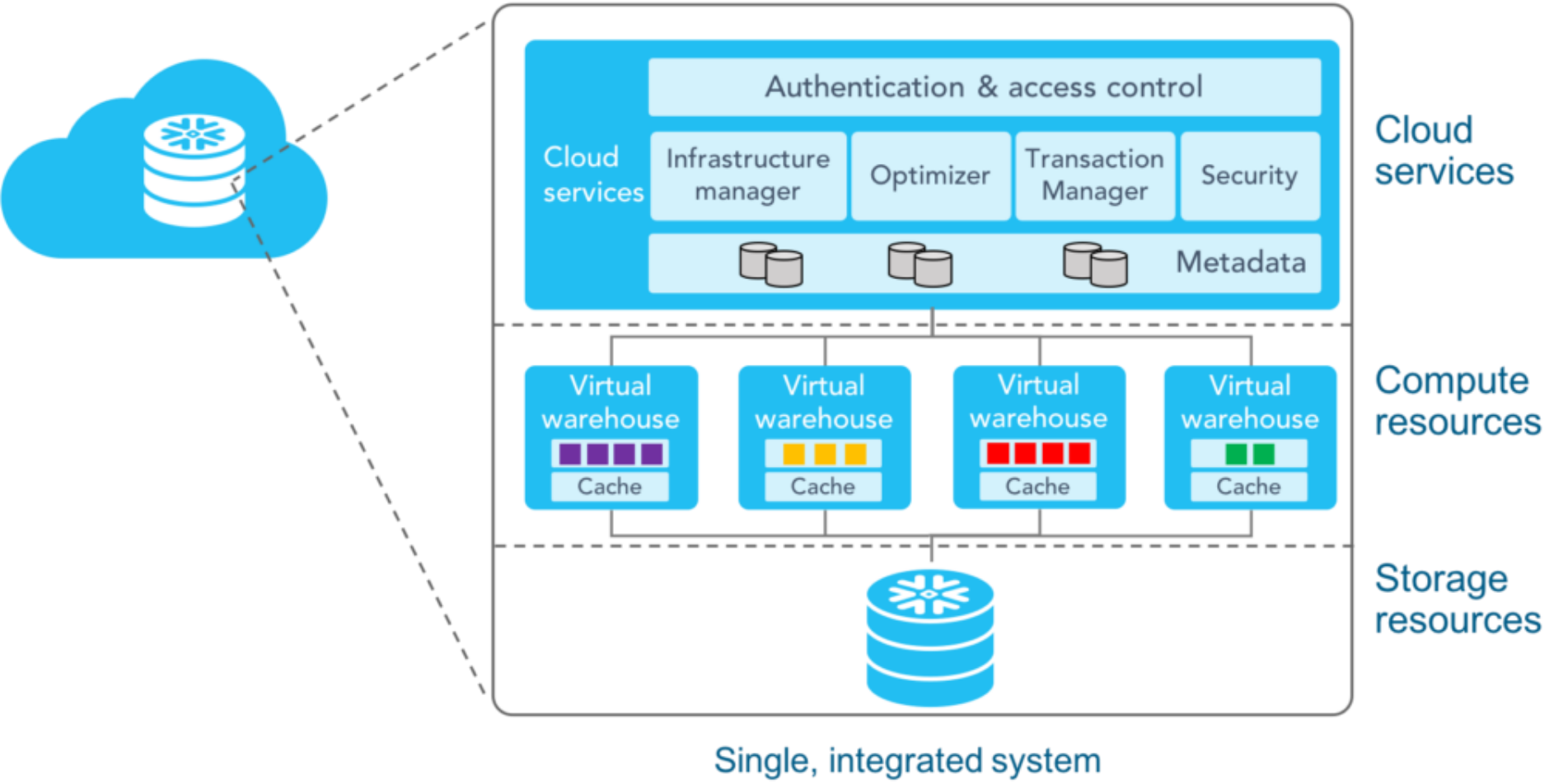
Yandex Clickhouse



Apache Druid

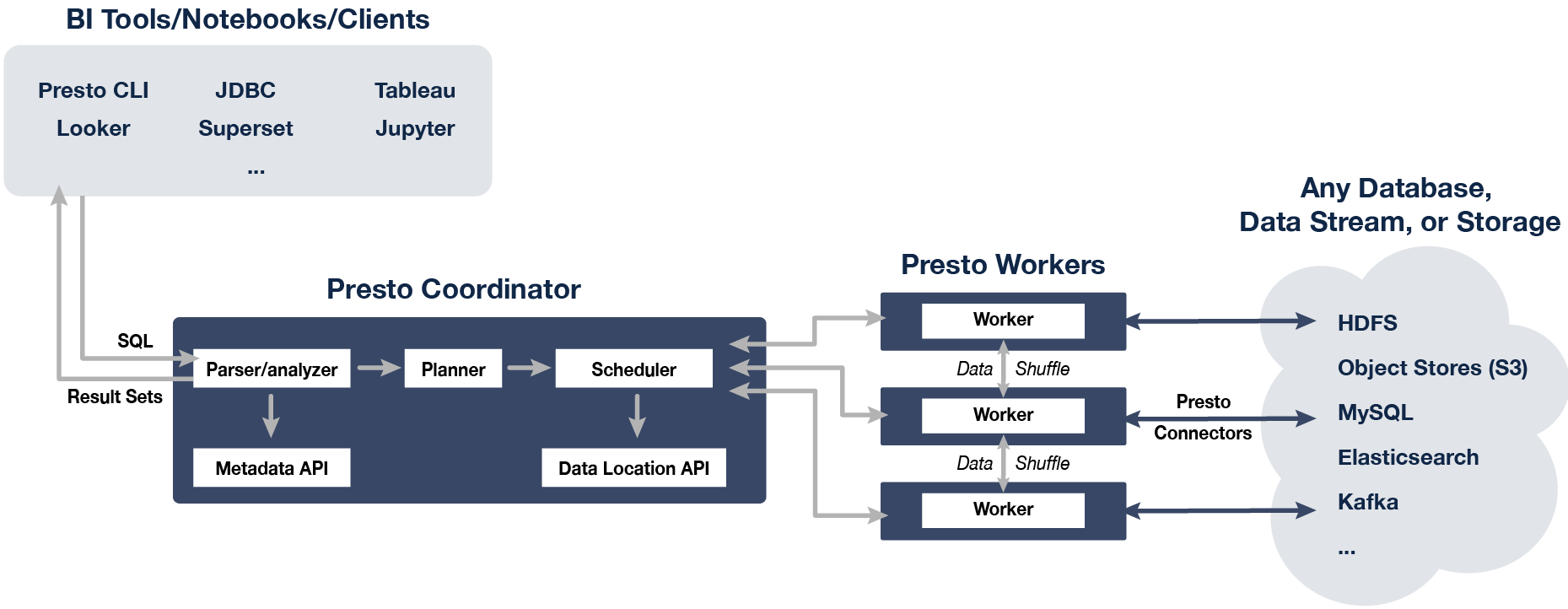


Snowflake
Multi-Cluster Shared Data Architecture



Single, integrated system

Facebook PrestoDB



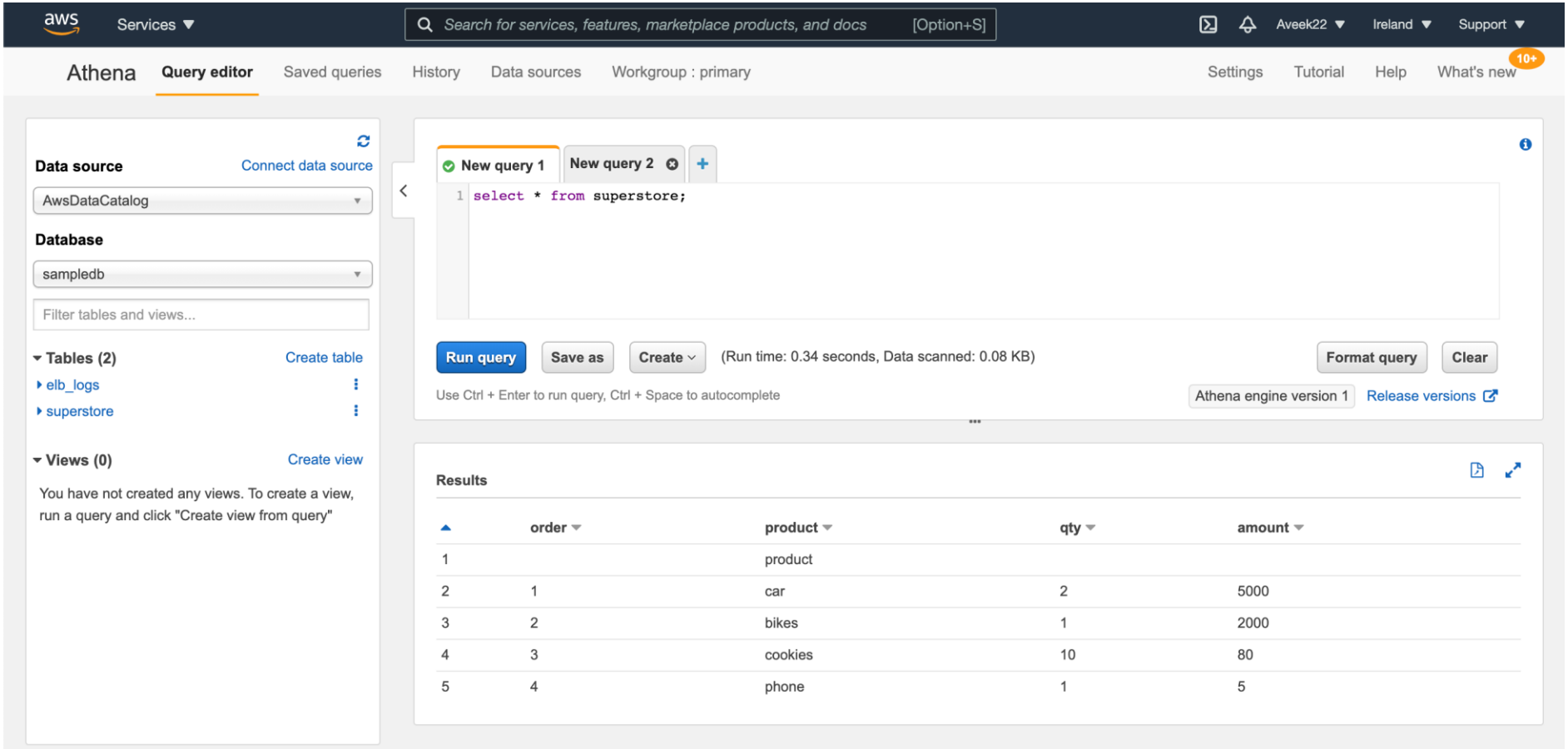
AWS Athena

```
CREATE EXTERNAL TABLE `nginx_202006`(
  `remote_addr` string COMMENT '',
  `remote_user` string COMMENT '',
  `day` string COMMENT '',
  `month` string COMMENT '',
  `year` string COMMENT '',
  `time` string COMMENT '',
  `zone` string COMMENT '',
  `method` string COMMENT '',
  `type` string COMMENT '',
  `path1` string COMMENT '',
  `trackingid` string COMMENT '',
  `path2` string COMMENT '',
  `http_protocol` string COMMENT '',
  `status_code` smallint COMMENT '',
  `other` string COMMENT '')
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  'input.regex'='(\\S+) - (\\S+) \\[(\\S{2})\\/(\\S{3})\\/(\\S{4})\\:(\\S+) (\\S+)\\] \\\"(\\S+)
\\/rtb\\/([^?/]+)([^=]+)=([^ &]+)(\\S*) (\\S+)\\\" (\\d{3}) (.+)')
LOCATION
  's3://mytest-bidder-logs/nginx_202006'
TBLPROPERTIES (
  'compressionType'='gz’
)
```

AWS Athena

```
CREATE EXTERNAL TABLE `request`(
  `time` bigint, `timebid` bigint, `type` string, `cid` string, `crid` string, `user_id` string, `sspid` string,
  `appid` string,
`skadnversion` string)
PARTITIONED BY (
  `year` string,
  `month` string,
  `day` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
LOCATION
  's3://mytest-prod/request'
TBLPROPERTIES (
  'parquet.compress'='GZIP',
  'projection.day.digits'='2',
  'projection.day.range'='1,31',
  'projection.day.type'='integer',
  'projection.enabled'='true’,
  'projection.month.digits'='2',
  'projection.month.range'='1,12',
  'projection.month.type'='integer',
  'projection.year.range'='2018,2025',
  'projection.year.type'='integer’
)
```

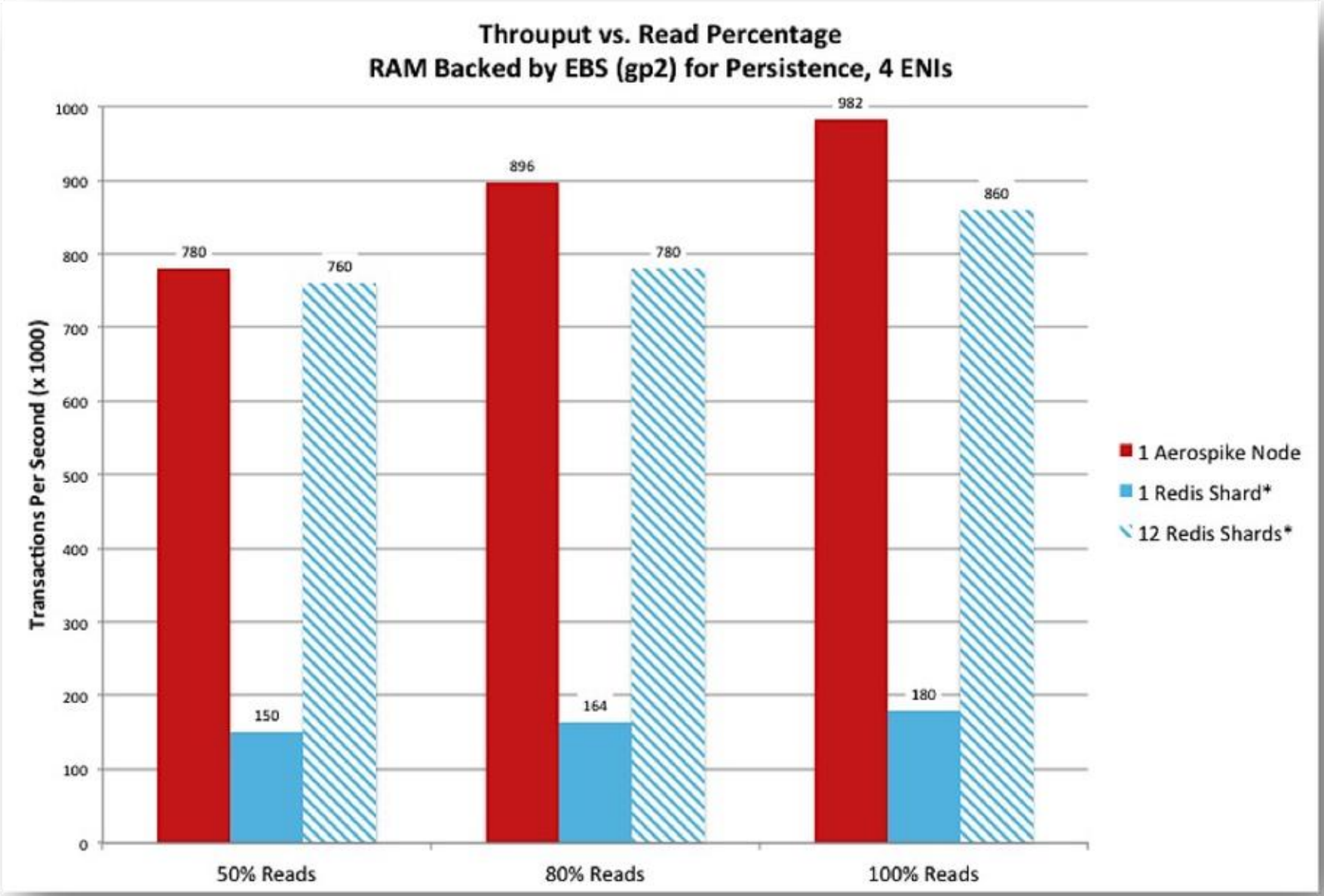
AWS Athena


aws
Services
Search for services, features, marketplace products, and docs
[Option+S]
Aveek22
Ireland
Support
Athena
Query editor
Saved queries
History
Data sources
Workgroup : primary
Settings
Tutorial
Help
What's new
10+
Data source
Connect data source
AwsDataCatalog
Database
sampledb
Filter tables and views...
Tables (2)
Create table
elb_logs
superstore
Views (0)
Create view
You have not created any views. To create a view, run a query and click "Create view from query"
New query 1
New query 2
1 select * from superstore;
Run query
Save as
Create
(Run time: 0.34 seconds, Data scanned: 0.08 KB)
Format query
Clear
Use Ctrl + Enter to run query, Ctrl + Space to autocomplete
Athena engine version 1
Release versions
Results
order
product
qty
amount
1 product
2 1 car 2 5000
3 2 bikes 1 2000
4 3 cookies 10 80
5 4 phone 1 5

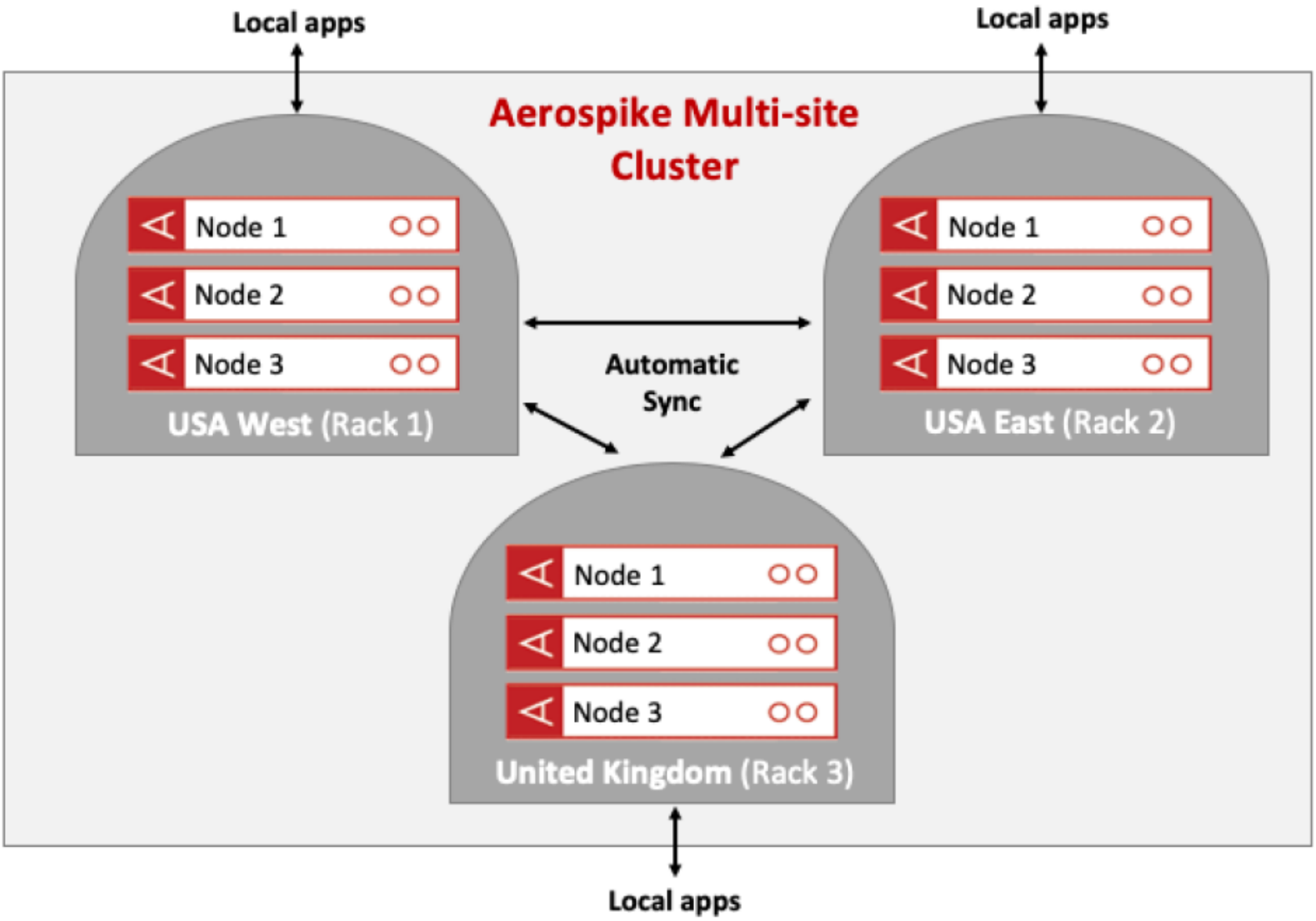
Key-Value NoSQL DB

- Fast single/multi put/get by primary key(s)
- Atomic increment/decrement/etc. operations
- Store complex structures
- Secondary indexes
- Persistence
- Clustering: sharding, replication factor
- Multi-clustering, async replication
- SSD instead of RAM (Aerospike)

Aerospike


Throuput vs. Read Percentage
RAM Backed by EBS (gp2) for Persistence, 4 ENIs
Transactions Per Second (x 1000)
1000
900
800
700
600
500
400
300
200
100
0
780
150
760
896
164
780
982
180
860
50% Reads
80% Reads
100% Reads
1 Aerospike Node
1 Redis Shard*
12 Redis Shards*

Aerospike


Local apps
Local apps
Aerospike Multi-site Cluster
Node 1
Node 2
Node 3
USA West (Rack 1)
Automatic Sync
Node 1
Node 2
Node 3
USA East (Rack 2)
Node 1
Node 2
Node 3
United Kingdom (Rack 3)
Local apps