

Building a Foundation for **AI** with **ClickHouse®** and **Apache Iceberg**

Robert Hodges - Altinity CEO

OSA Con Meetup 2026 - Open Lakehouse and AI



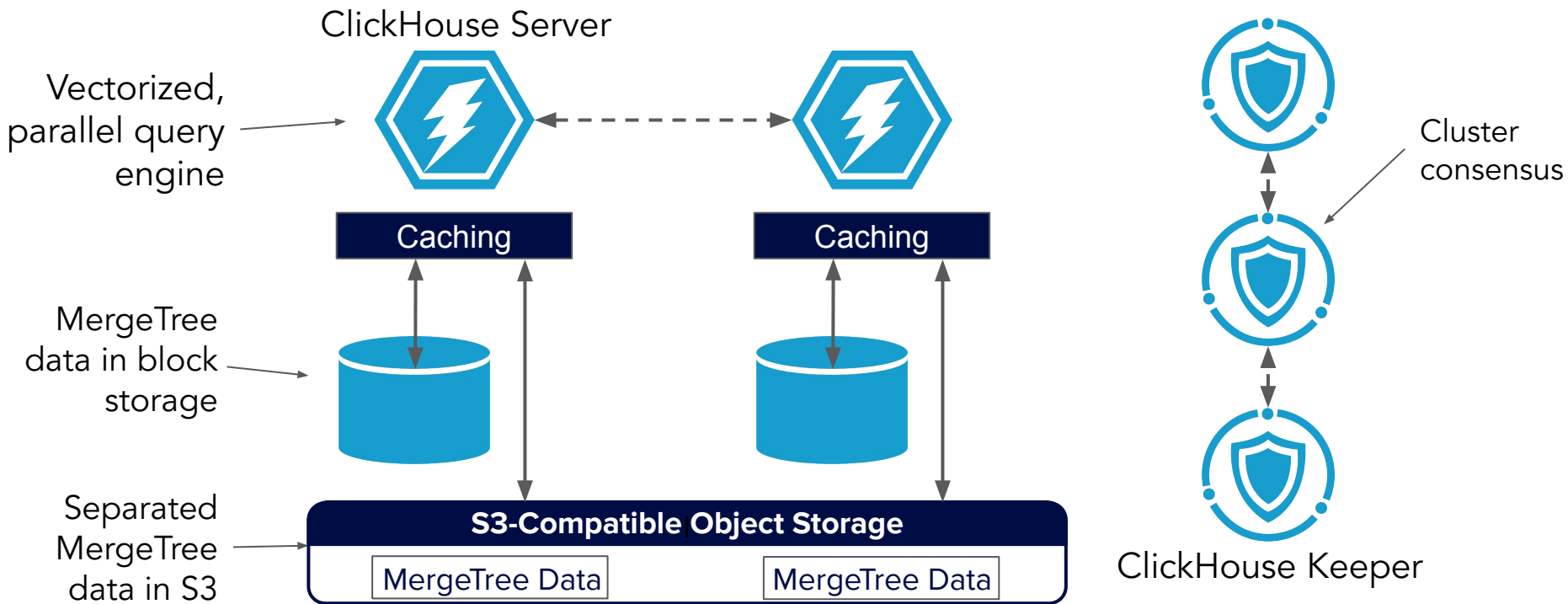
ALTINITY®

Run Open Source ClickHouse® Better

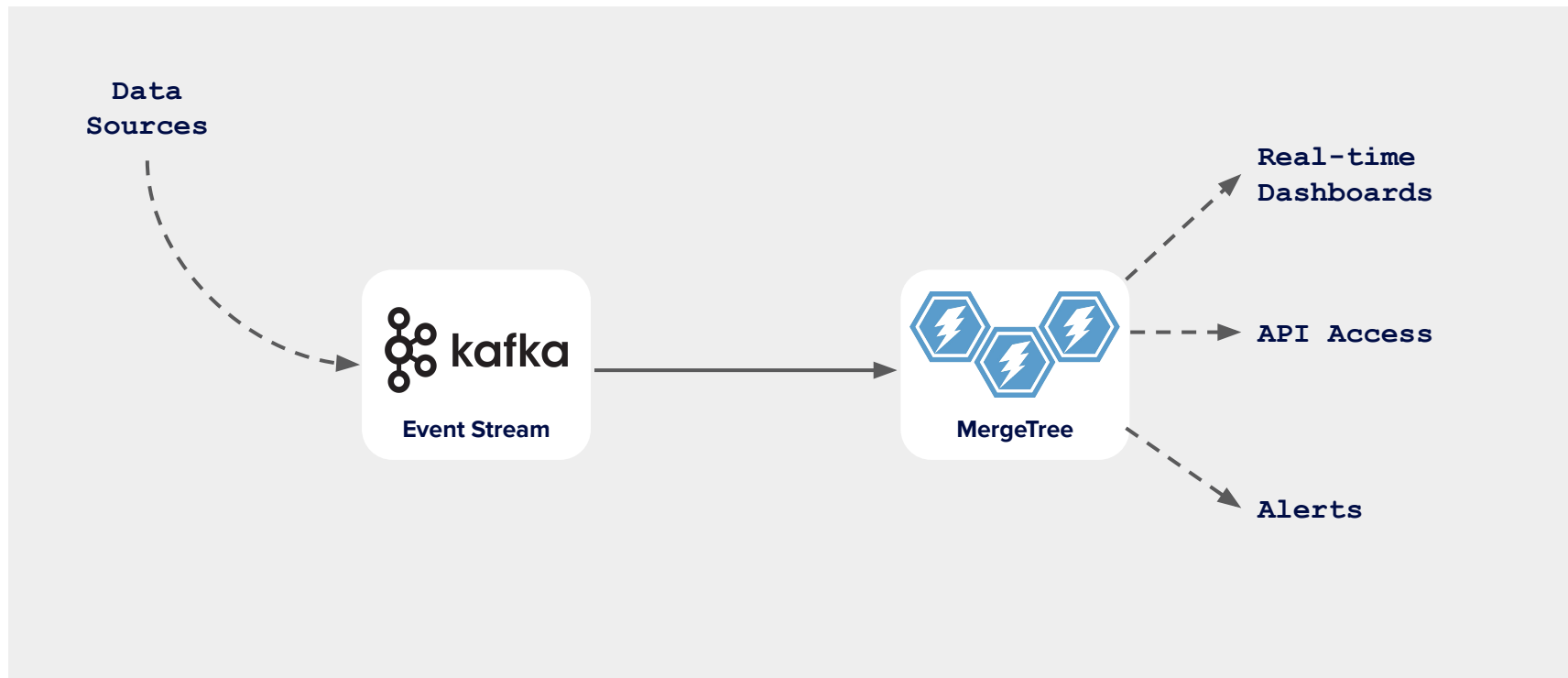
Altinity.Cloud Enterprise Support

Altinity® is a Registered Trademark of Altinity, Inc.
ClickHouse® is a registered trademark of ClickHouse, Inc.;
Altinity is not affiliated with or associated with ClickHouse, Inc.

ClickHouse shared nothing architecture



Traditional real-time event pipeline with ClickHouse



ClickHouse works great for almost any real-time analytics

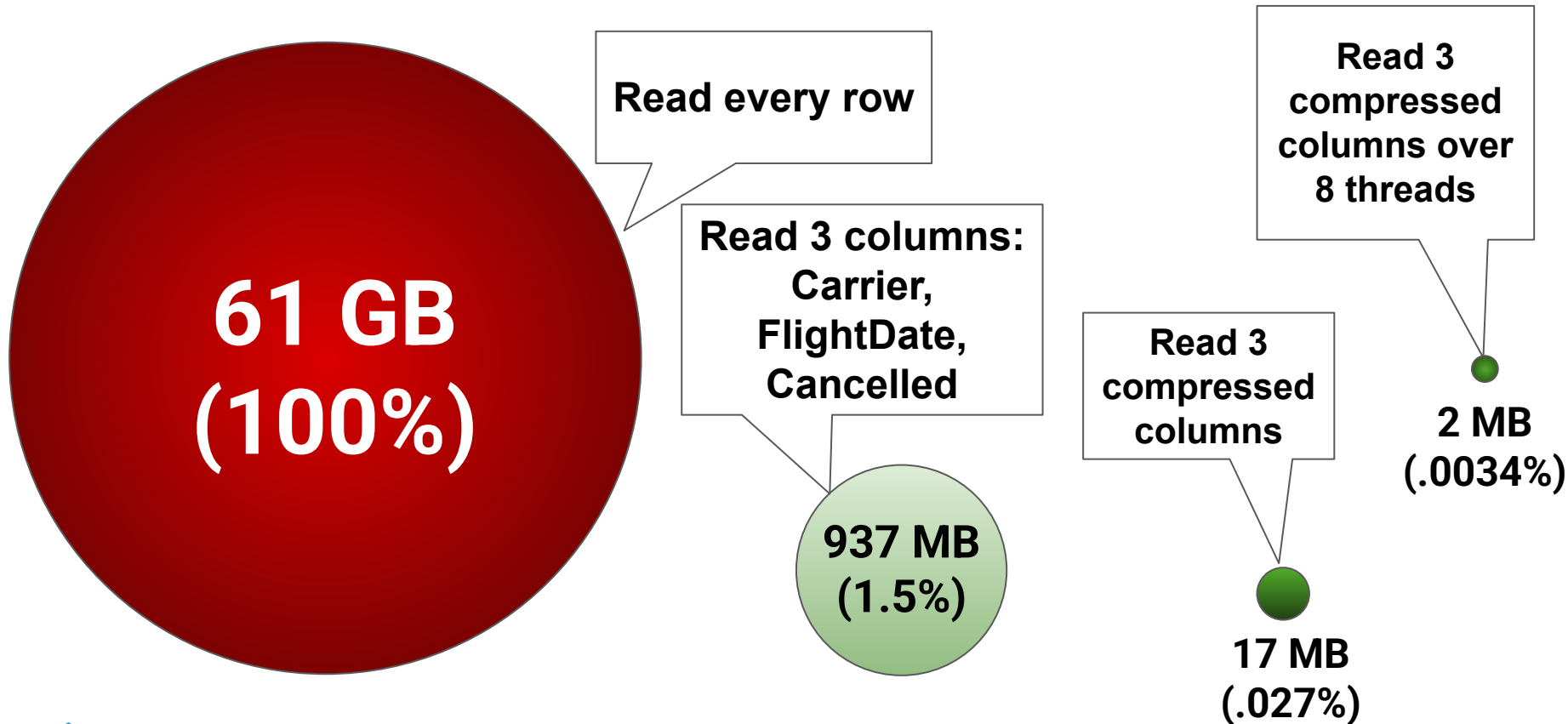
```
SELECT Carrier, toYear(FlightDate) AS Year,  
       (sum(Cancelled) / count(*)) * 100. AS cancelled_pct  
FROM default.ontime_ref  
GROUP BY Carrier, Year HAVING cancelled_pct > 1.  
ORDER BY cancelled DESC LIMIT 10
```

	Carrier	Year	cancelled_pct
1.	G4	2020	16.733186040434276
2.	EA	1989	10.321500966388536
3.	WN	2020	9.284307653599388

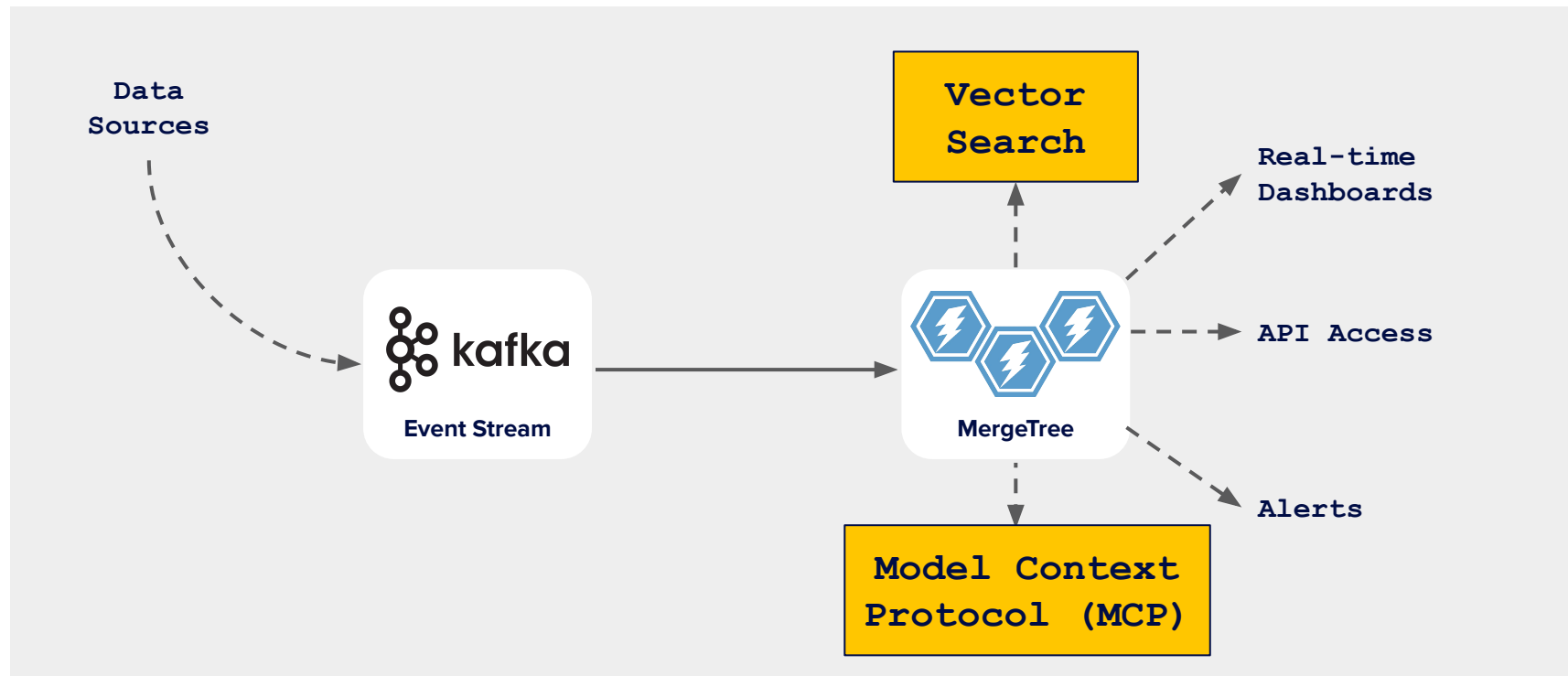
. . .

10 rows in set. Elapsed: **0.674 sec.** Processed **196.51 million rows**, 982.57 MB (**291.68 million rows/s.**, 1.46 GB/s.)10 rows

Visualizing effect of columns and compression on I/O

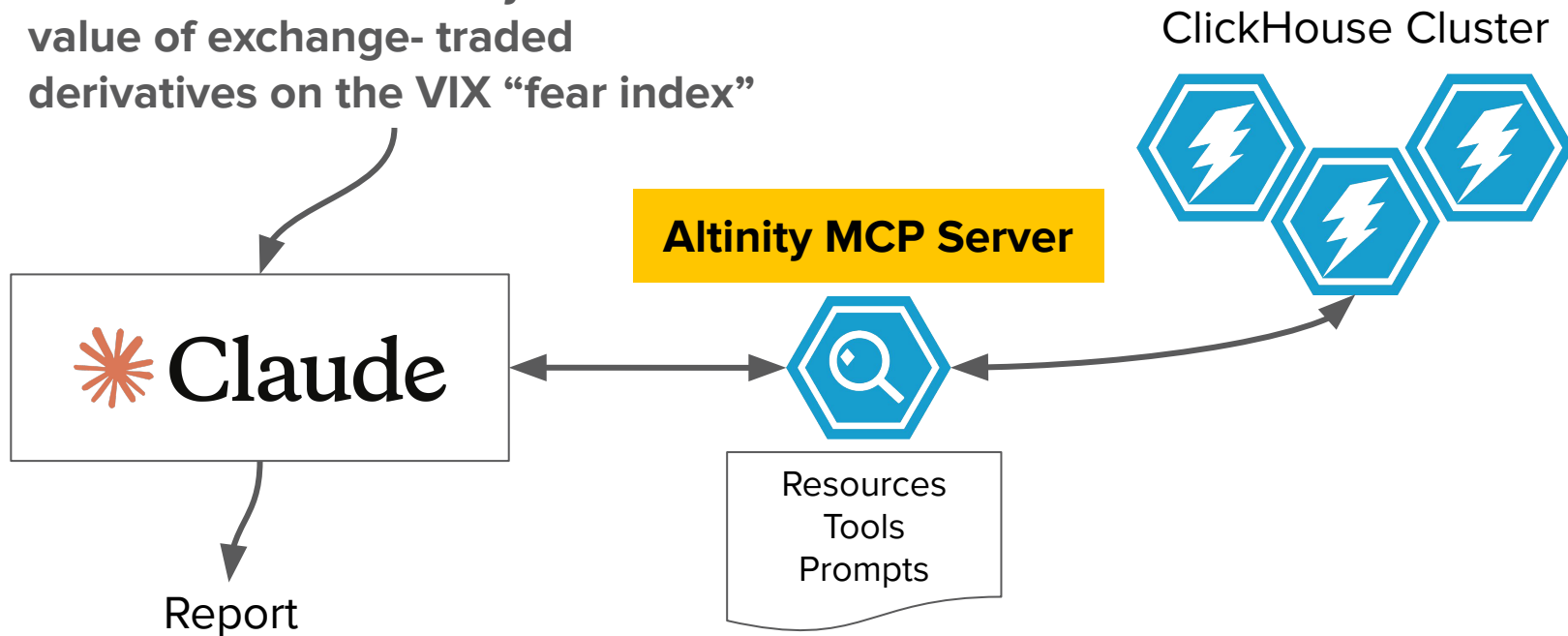


ClickHouse is adding capabilities for AI



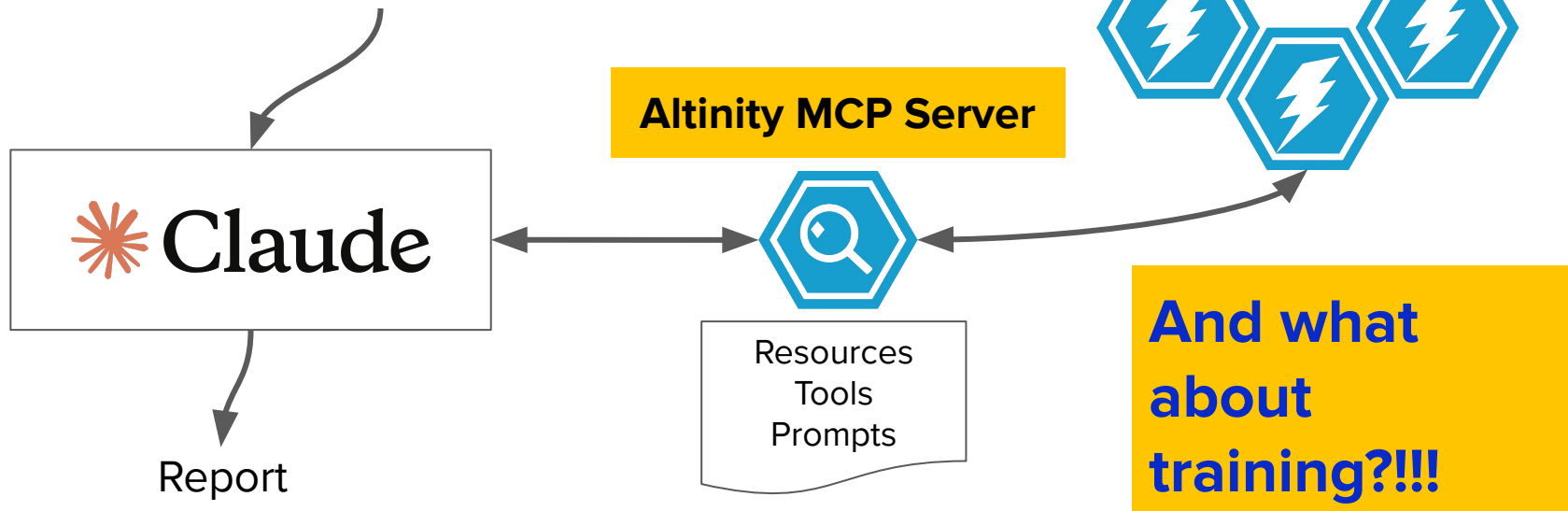
New workloads, new costs of operation

Show me the minute-by-minute
value of exchange- traded
derivatives on the VIX “fear index”



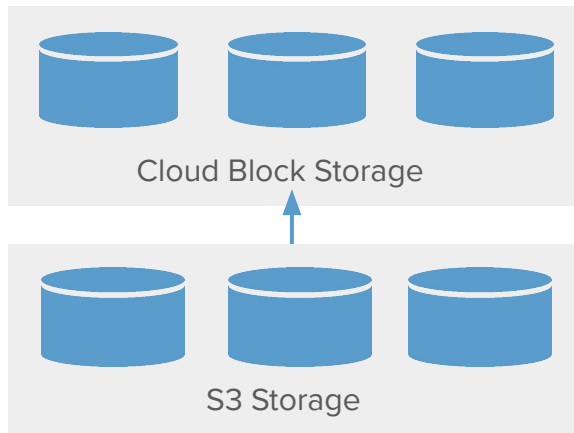
New workloads, new costs of operation

Show me the minute-by-minute
value of exchange- traded
derivatives on the VIX “fear index”

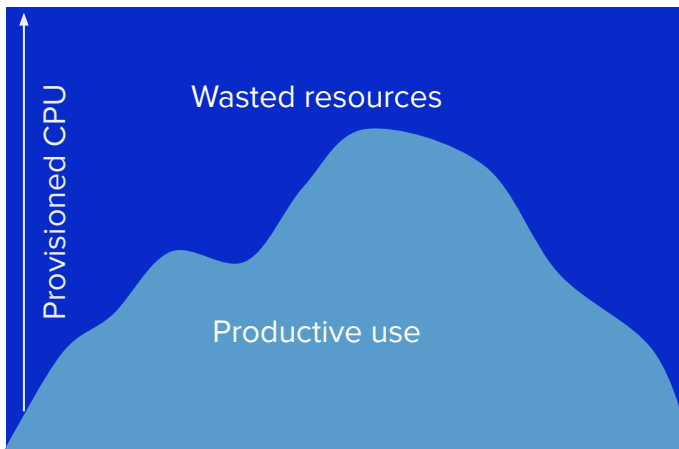


Storage and compute cost increase rapidly with data size

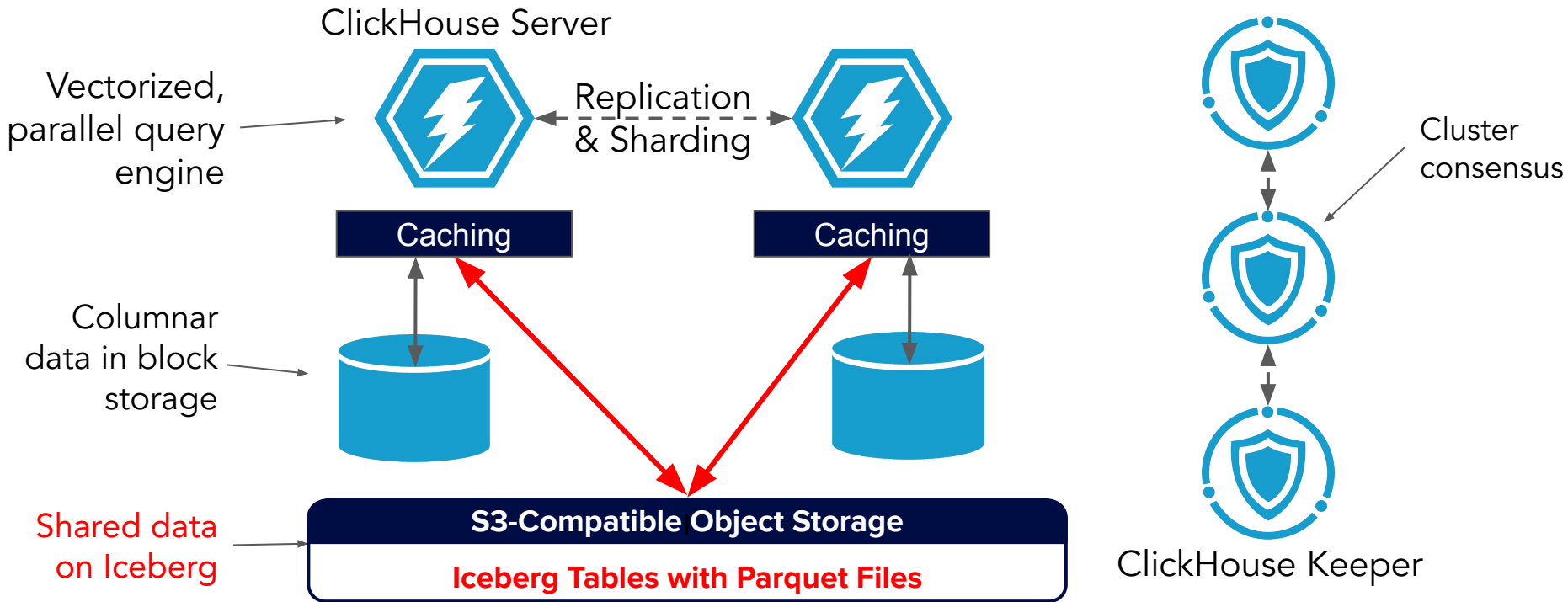
Block storage with replication is
10x more expensive



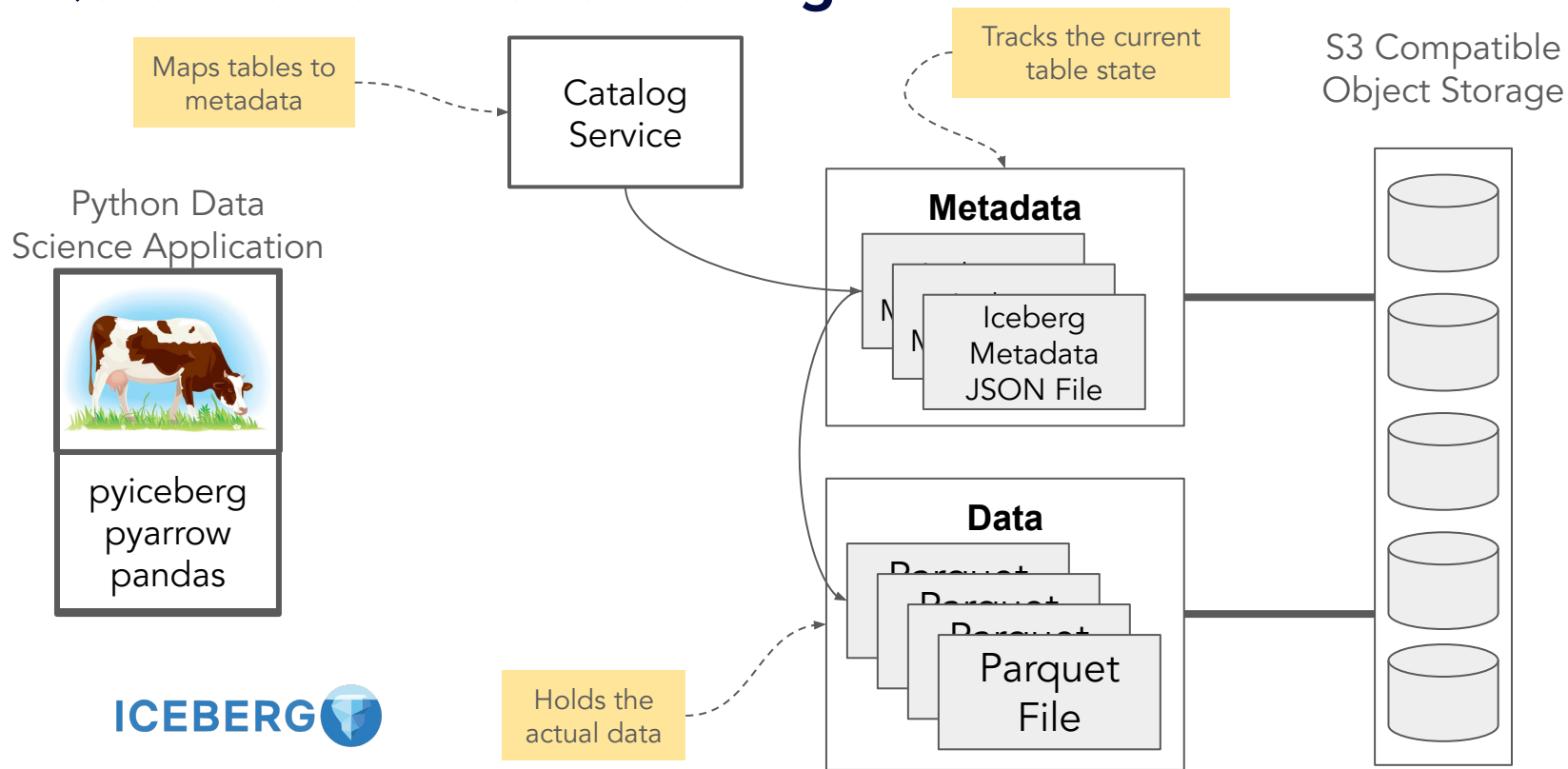
Overprovisioning wastes compute



Can we share Iceberg tables in open source Clickhouse?



Quick aside: What's Iceberg?



Project Antalya adds shared Iceberg storage to ClickHouse

Upstream Iceberg Support

CLICKHOUSE & ICEBERG INTEGRATION

Vendor catalogs, e.g., Glue
Positional & equality deletes
Column renaming
Schema evolution
Inserts to Iceberg tables
Native Parquet reader
...

Project Antalya

ICEBERG AS PRIMARY STORAGE

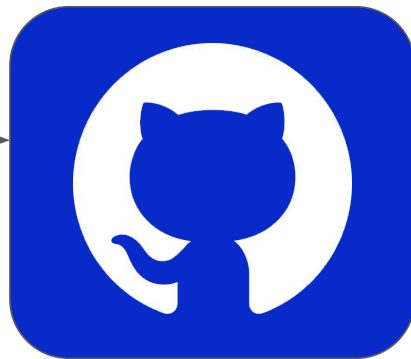
Compute/storage separation
Tiered storage
Transparent type mapping
Full query compatibility
Transactional bulk export
Built-in catalog support
Compaction
...

What's the relationship between Antalya and ClickHouse?

Upstream ClickHouse Repo



Altinity Repo



Commits
from
Altinity
Devs

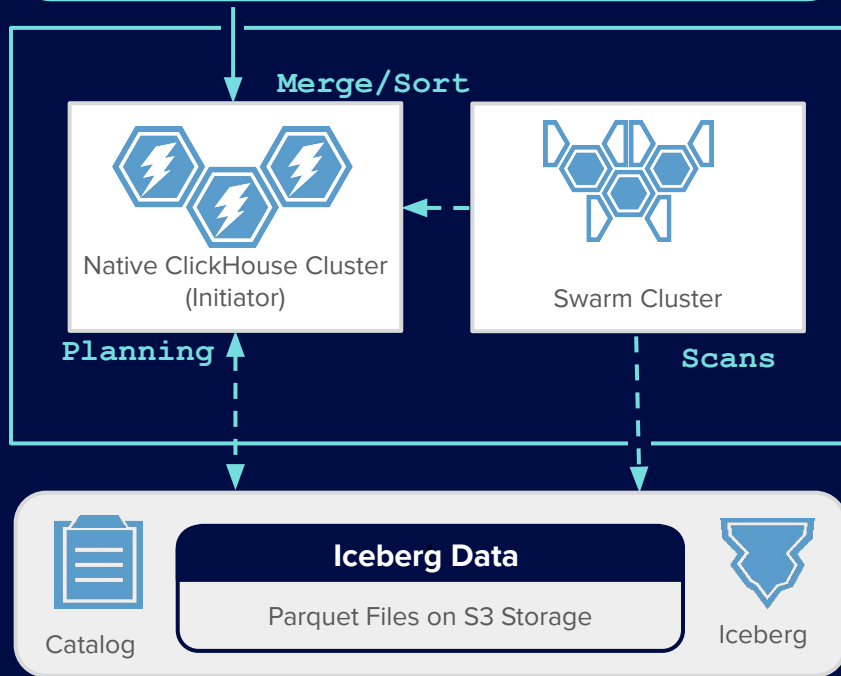
Project Antalya Builds
<https://builds.altinity.cloud>

Project Antalya Containers
<https://hub.docker.com/u/altinity>

Swarm clusters add separated compute & storage

- **Initiator** plans the query
- **Swarm Cluster nodes** scan shared files
- **Initiator** merges and sorts responses
- Lots of **new caches!**

```
SELECT data, sum(output_count)
FROM iceberg.`btc.transactions`
WHERE date >= '2024-01-01'
GROUP BY date ORDER BY date
SETTINGS
object_storage_cluster = 'swarm'
```



It's plausible to combine storage, but will it work?

MergeTree

Loads almost instantly

Compaction is built-in and fast

Low latency to first record

Expensive

Scaling is slow & limited

Limited I/O bandwidth

Access only through ClickHouse server

Iceberg

Loaded in batches, e.g., every hour

Compaction is external and (often) slow

High latency to first record

Cheap[er] storage and compute

Fast horizontal and vertical scaling

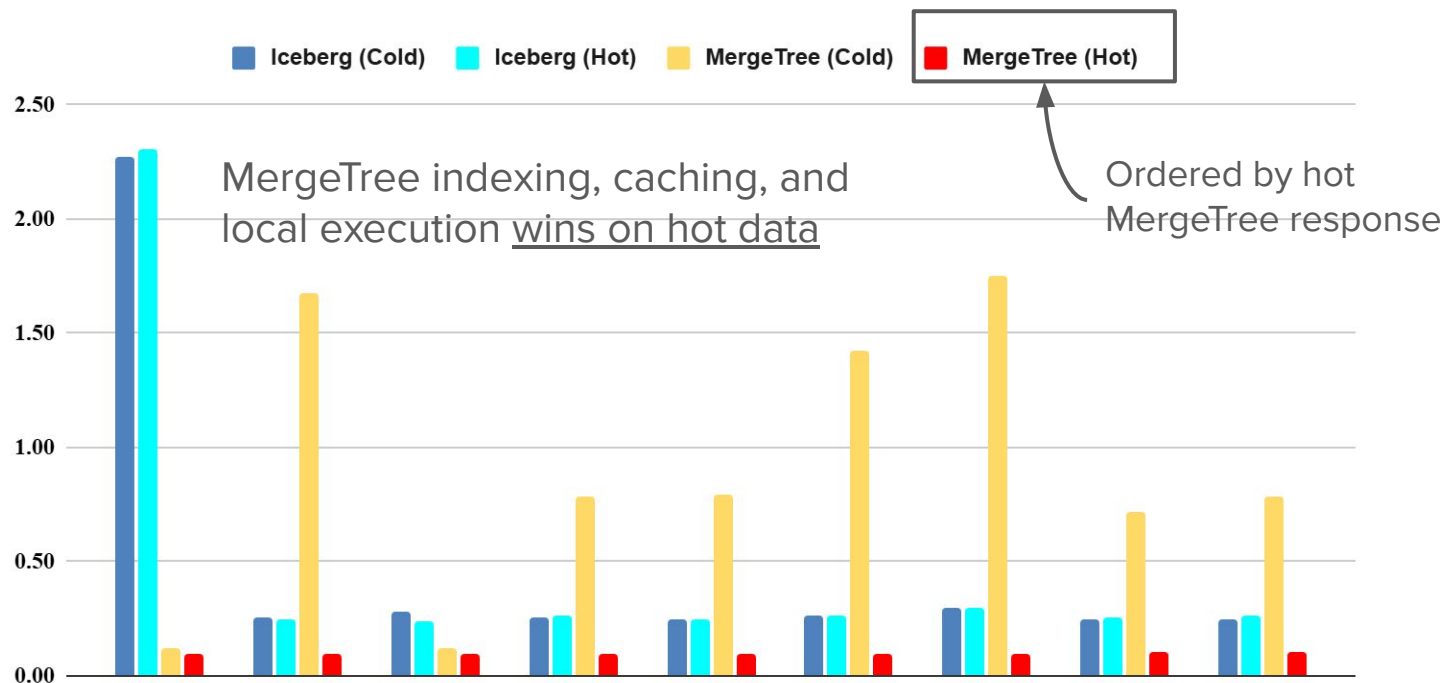
Unlimited I/O bandwidth

Shareable with other applications

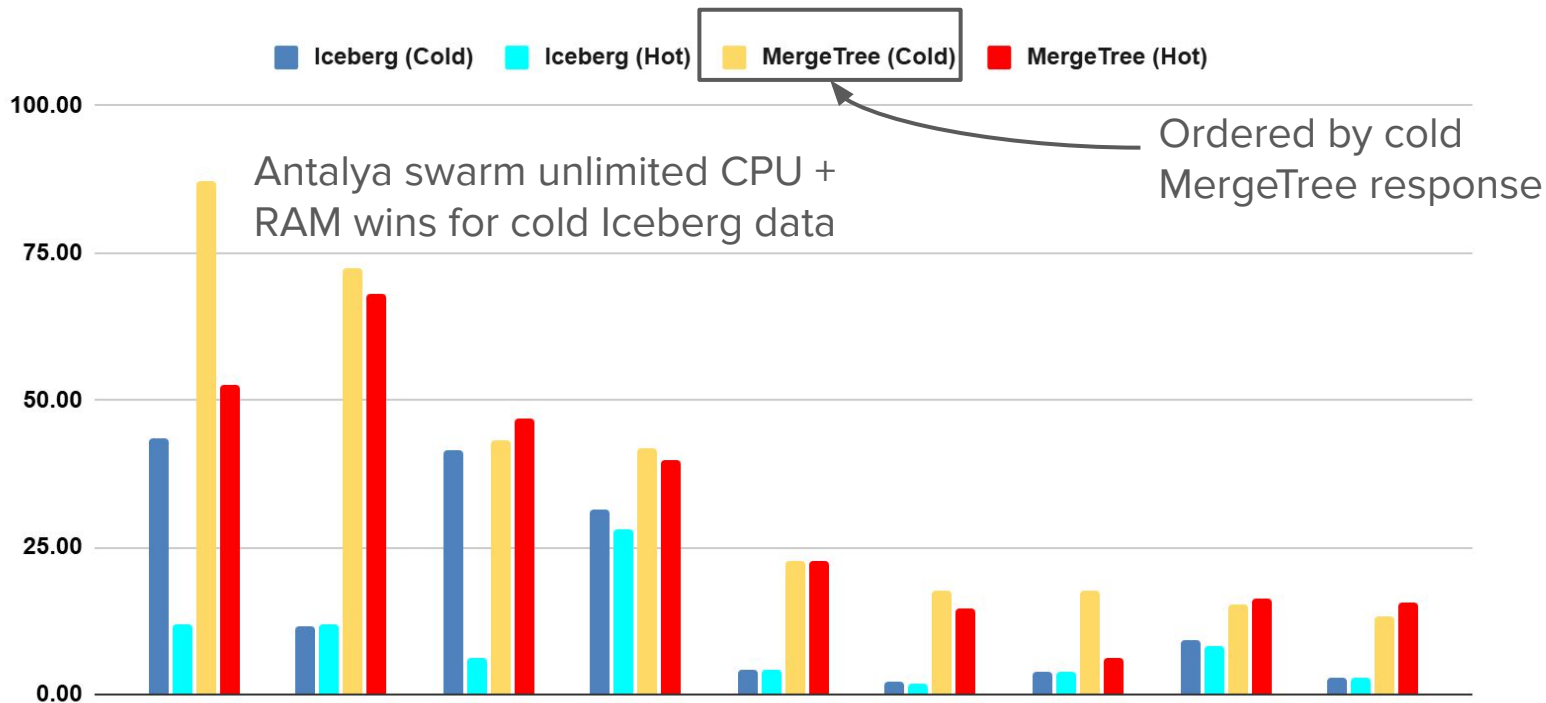
Let's compare Iceberg vs native MergeTree performance

- Simulated customer data
 - 40 most common queries
 - Compare MergeTree vs. Iceberg
 - Measure best response hot & cold
- 1 Initiator with MergeTree data
 - c8g.4xlarge (16 vCPU, 32 GiB RAM),
 - Encrypted EBS GP3 500Mb/s
 - 4 swarm nodes
 - c8g.4xlarge (16 vCPU, 32 GiB RAM)
 - Encrypted EBS GP3 500MiB/s
 - Spot instances enabled

Performance: Iceberg vs MergeTree (HOT DATA)

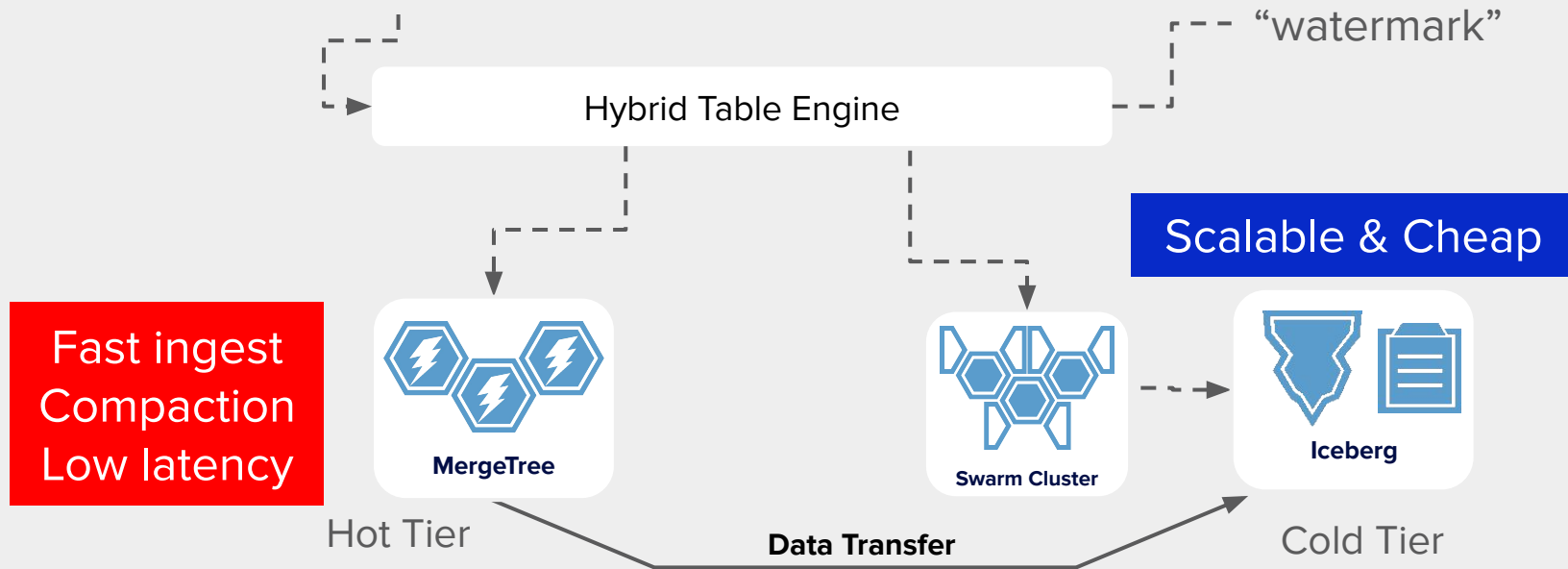


Performance: Iceberg vs MergeTree (COLD DATA)



Hybrid Engine combines MergeTree and Iceberg tables

```
SELECT sum(output_count) FROM btc.transactions
```



An example of a Hybrid table with a watermark

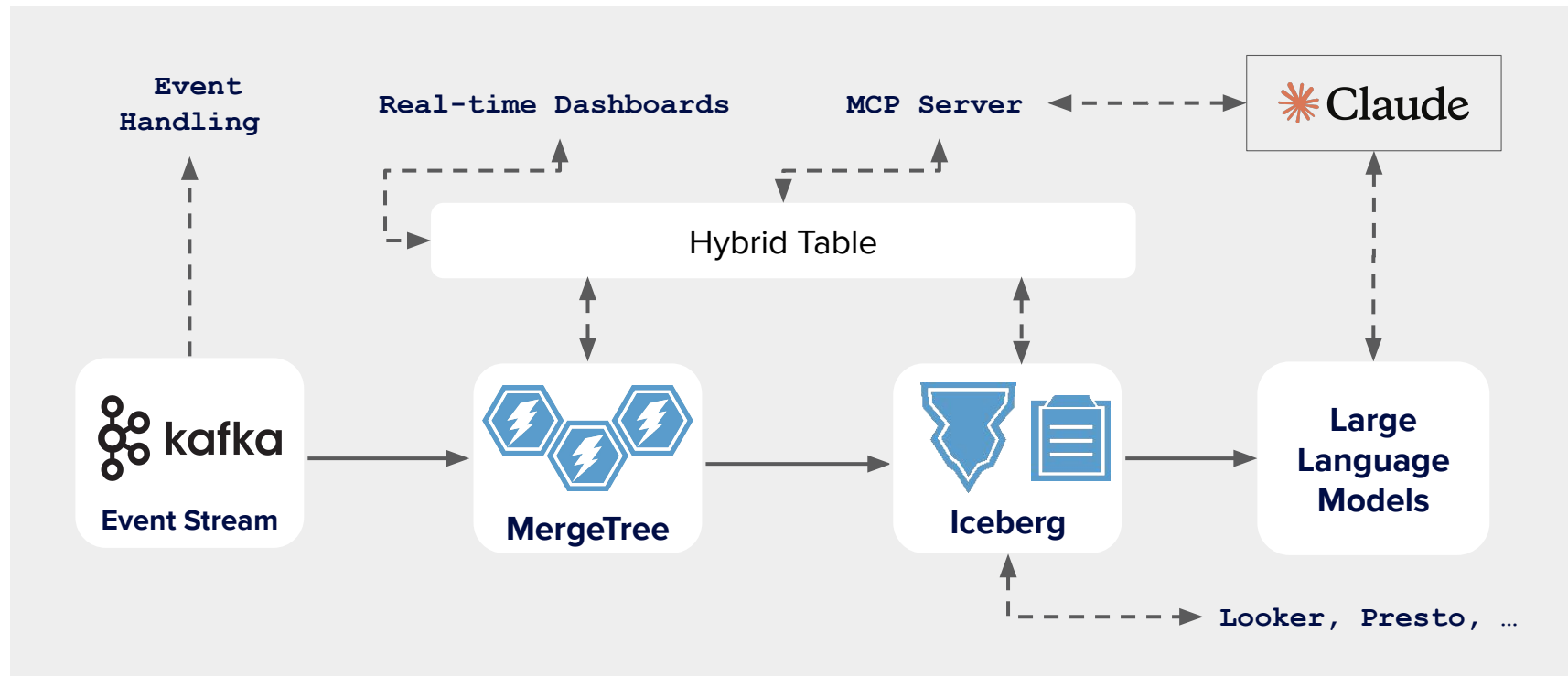
```
CREATE TABLE btc.transactions
AS mt.transactions
ENGINE = Hybrid(
  cluster('{cluster}', mt.transactions),
  eventDate >= dictGet('system.watermarks_dict',
    'watermark', 'btc.transactions'),
  -----ice.`mt.transactions`,
  eventDate < dictGet('system.watermarks_dict',
    'watermark', 'btc.transactions')
```

Hot data

Watermark

Cold data

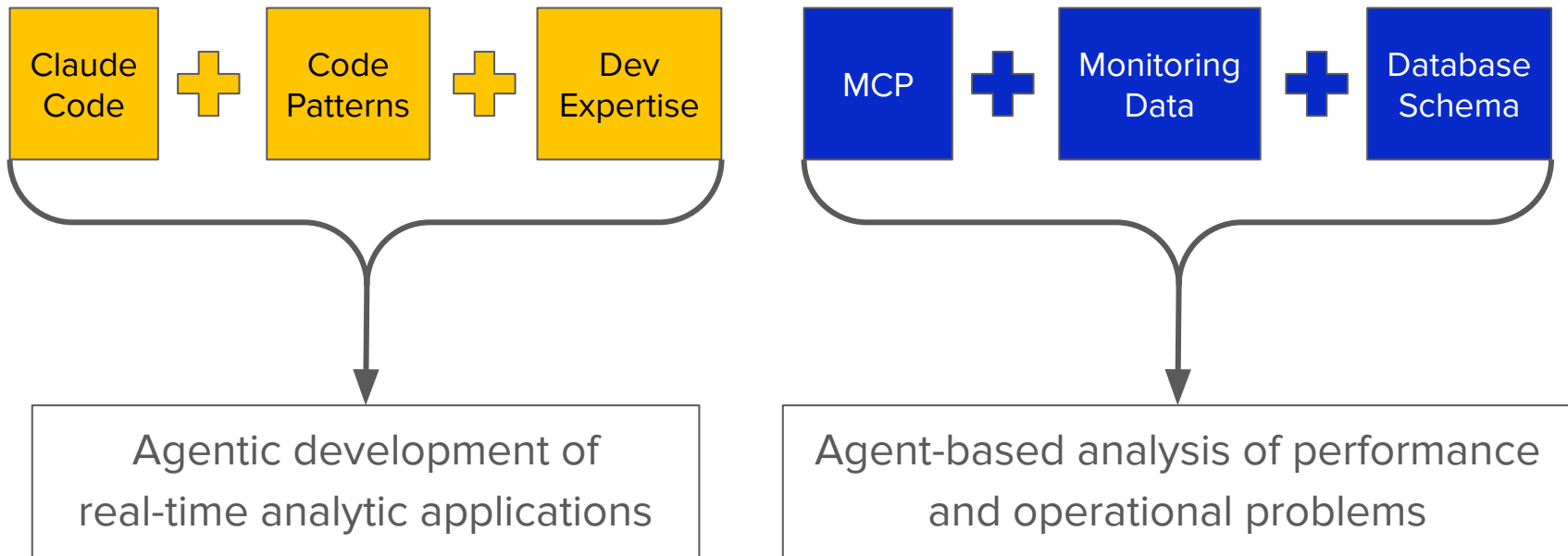
Our end goal: A single, unified pipeline for multiple apps



What have we learned?

- ClickHouse and Iceberg have complementary strengths
- Making the combination transparent is the hard problem
 - Data types, timestamps, partitioning, ordering
- Everything needs to happen in SQL – avoid external ETL
- Users want data/processing in their own environments
- Automation is crucial
 - Do you really want to run Iceberg Catalogs or ZooKeeper yourself

Bonus learnings: Where can we see AI actually working?



Try Altinity Antalya Now

Project Antalya Examples - <https://github.com/Altinity/antalya-examples>

Altinity Documentation - <https://docs.altinity.com>

Altinity ClickHouse Repo - <https://github.com/Altinity/ClickHouse>

Project Antalya Builds - <https://builds.altinity.cloud>

Altinity Slack - <https://altinity.com/slack>

Slides for Presentation



Thank you! Questions?

Robert Hodges
CEO Altinity
Contact me

<https://altinity.com>



My LinkedIn

