Cloudinary

Best Practices and Insights when migrating to Apache Iceberg for Data Engineers

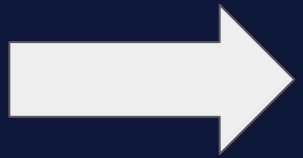
Amit Gilad
Data engineer

Agenda

- Introduction
- Ingestion
- Compaction
- Maintenance
- Monitoring
- Benchmarks

About me

Cloudinary



About Cloudinary

- 10,000 customers
- 2,000,000 Developers
- 60 Billion assets
- 30 PB monthly bandwidth

- 10-20 TB of logs daily
- 50 Billion records every day
- ~7-14 GB every minute

Why  ?

Cost

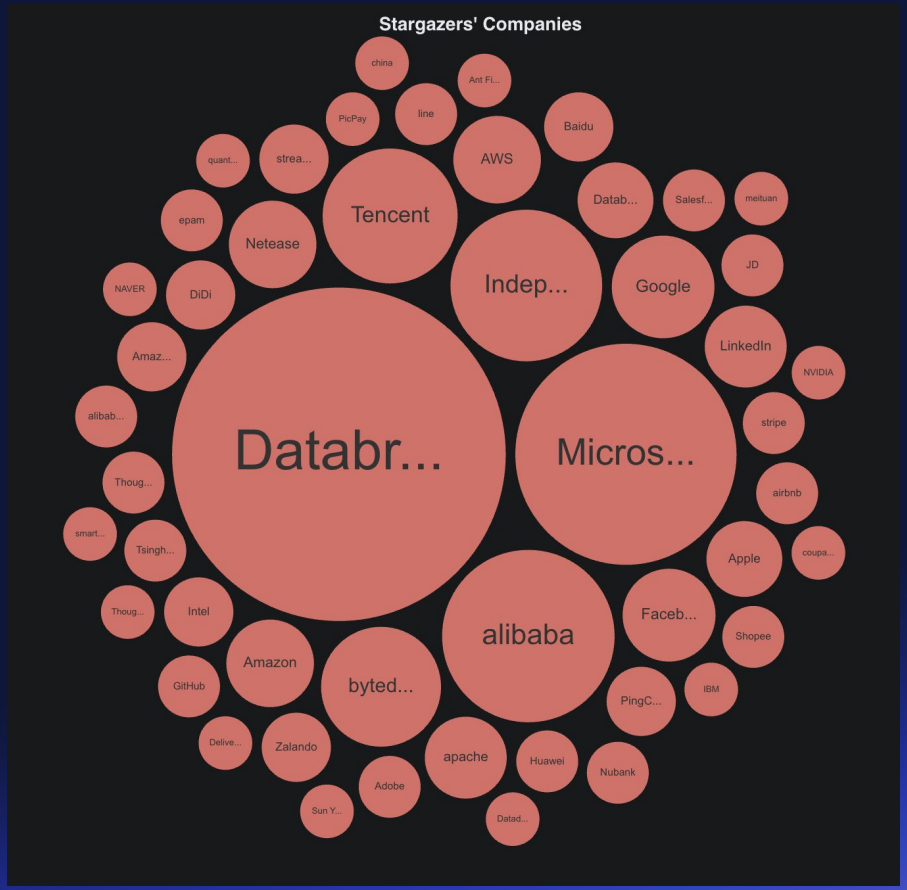
Time travel

Features

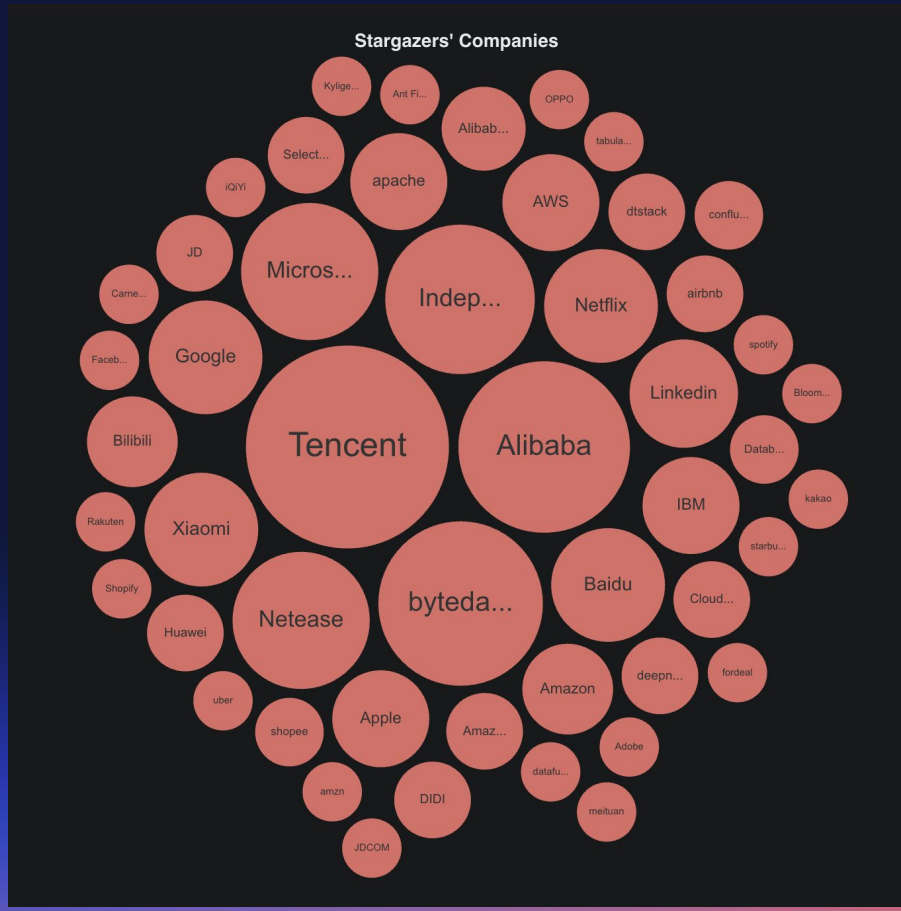
Multiple engines

Data retention

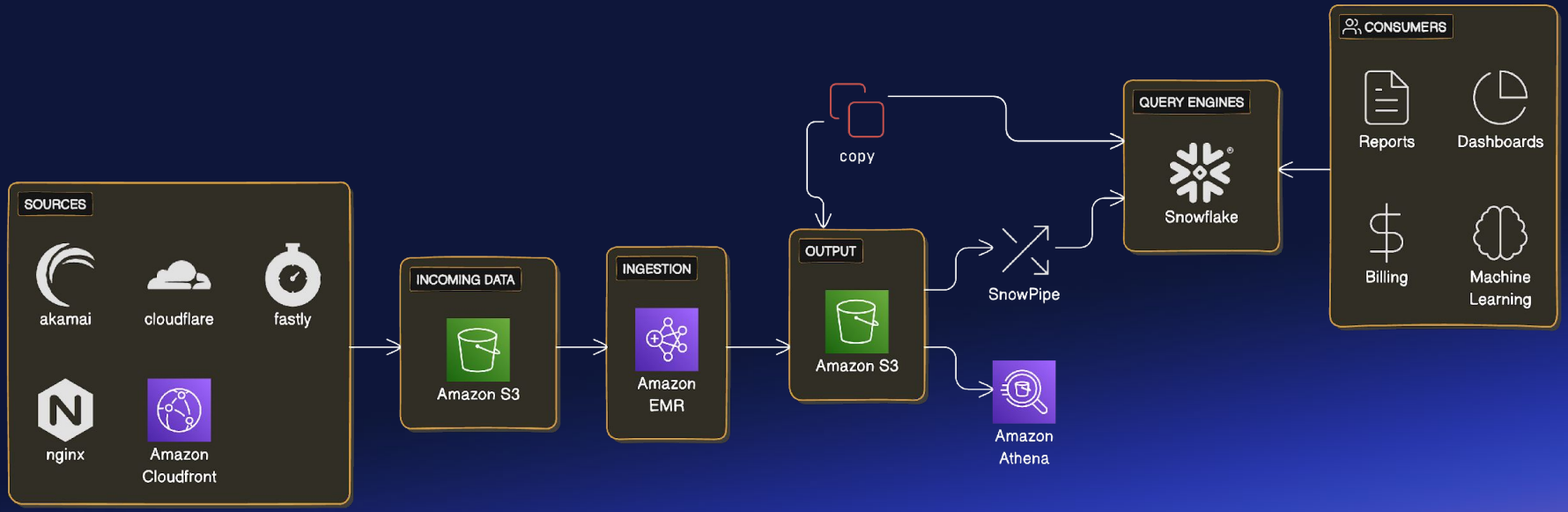
Delta



Apache Iceberg



Previous Architecture



End Result

- Reduce Storage cost by 25%
- Data retention 6X
- Query cost reduced by 25-40%
- Single copy
- Reduce query execution time by 30% to 50%

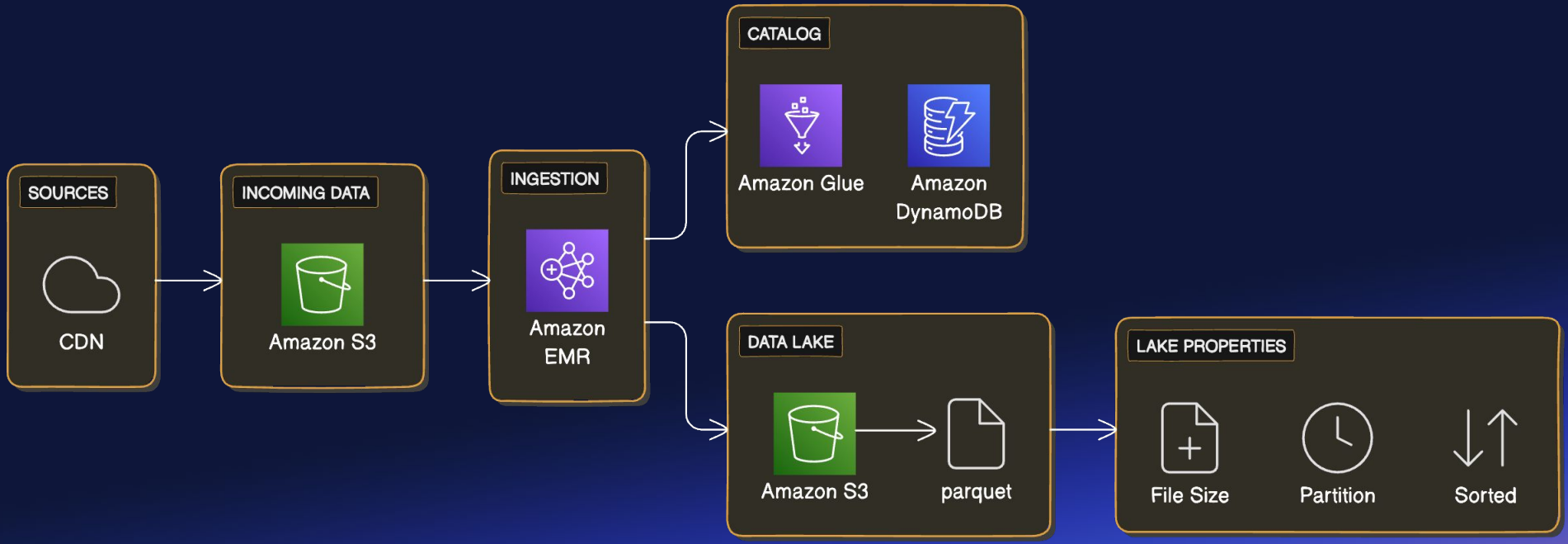
Cloudinary

Getting ready

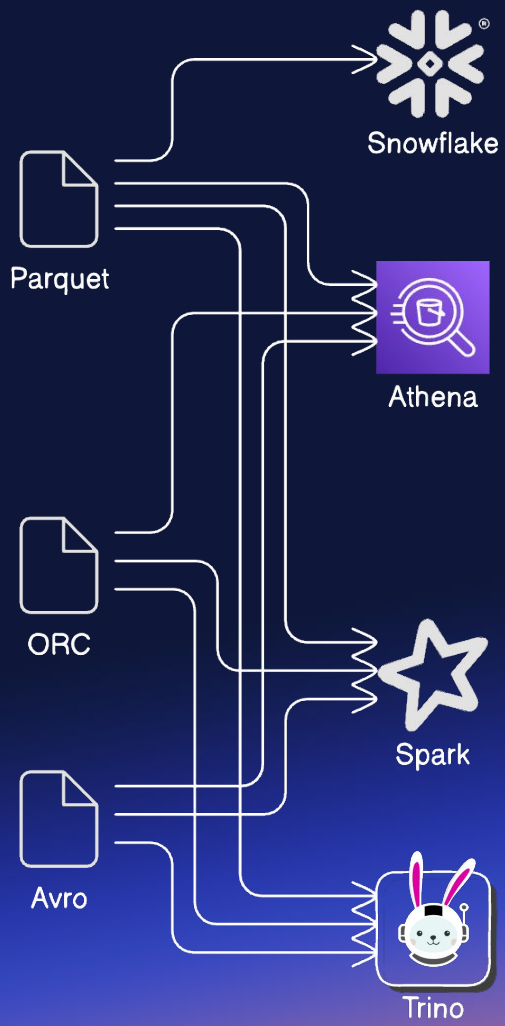
- Select tables for migration
- Mission critical queries
- Cost
- Execution time
- Identify users & what tools they use

The Journey begins

Ingestion


SOURCES
CDN
INCOMING DATA
Amazon S3
INGESTION
Amazon EMR
CATALOG
Amazon Glue
Amazon DynamoDB
DATA LAKE
Amazon S3
parquet
LAKE PROPERTIES
File Size
Partition
Sorted

File Format



Compression ?

Cloudinary

Table configuration

- write.target-file-size-bytes = 1073741824
- write.distribution-mode = hash
- write.parquet.compression-codec = zstd
- write.metadata.delete-after-commit.enabled = true
- write.metadata.previous-versions-max = 500

Adaptive Query Execution = Spark > 3.0.0

- Spark.sql.adaptive.enabled
- spark.sql.adaptive.coalescePartitions.enabled

} SMALL FILES

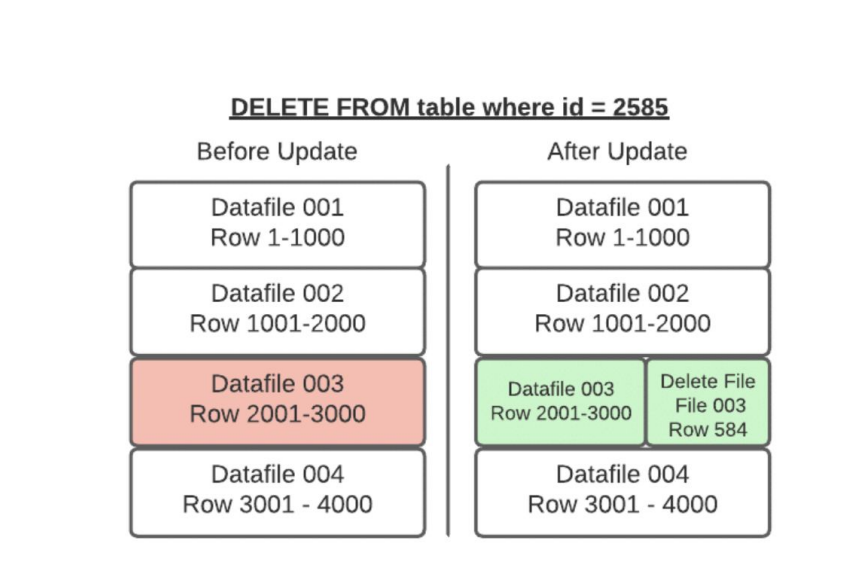
- spark.sql.adaptive.skewJoin.enabled

} BIG FILES

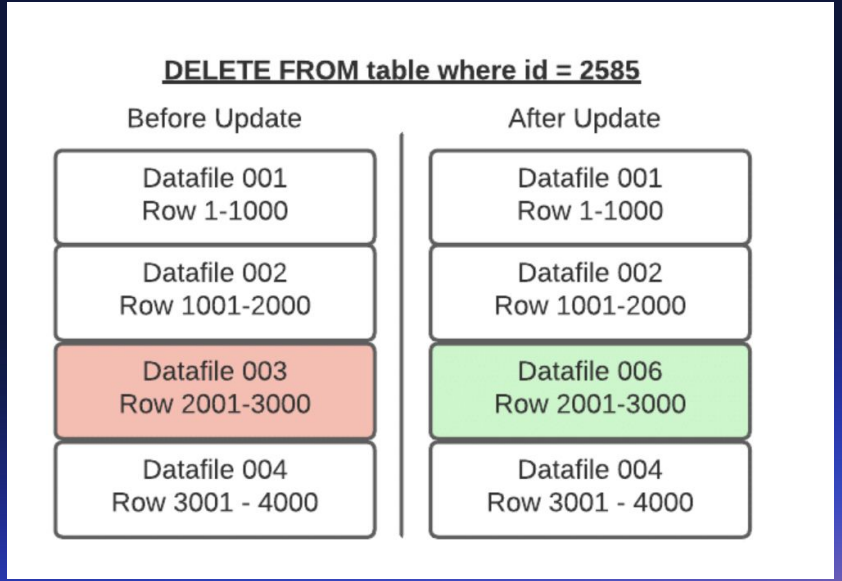
Cloudinary

CoW vs MoR

Merge-on-Read

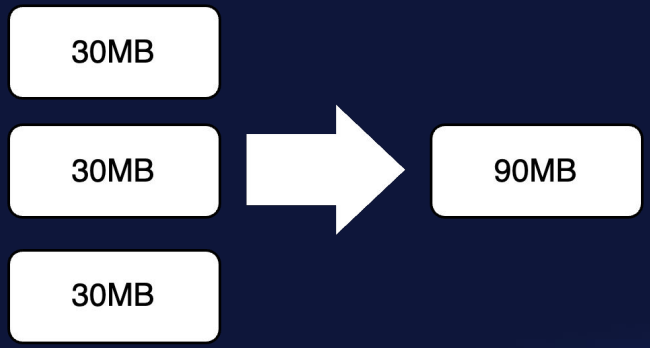


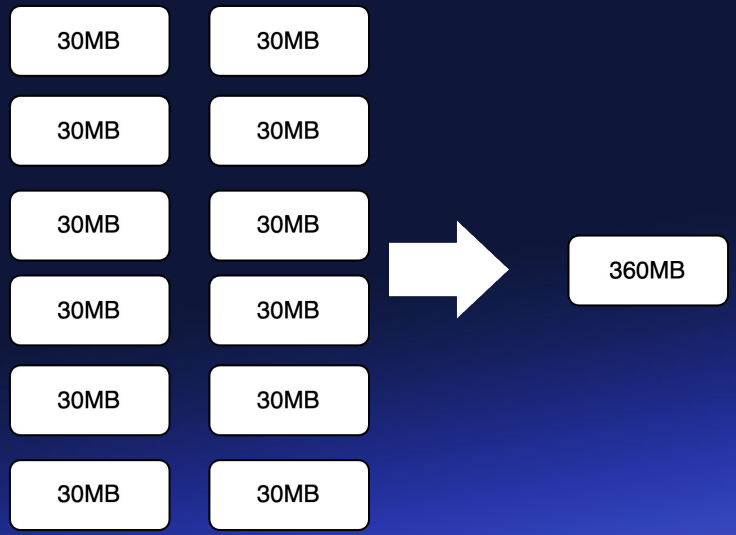
Copy-on-Write



Rewrite data files (Compaction)

Compact Small files





Rewrite deletes

`delete-file-threshold` - 2147483647 b

Rewrite data file (compaction) startegy

BinPack

Simple merge or split of files in targeted partitions

- Light operation
- No shuffles

Sort

Shuffle the data in targeted partitions based on hierarchical sort key(s)

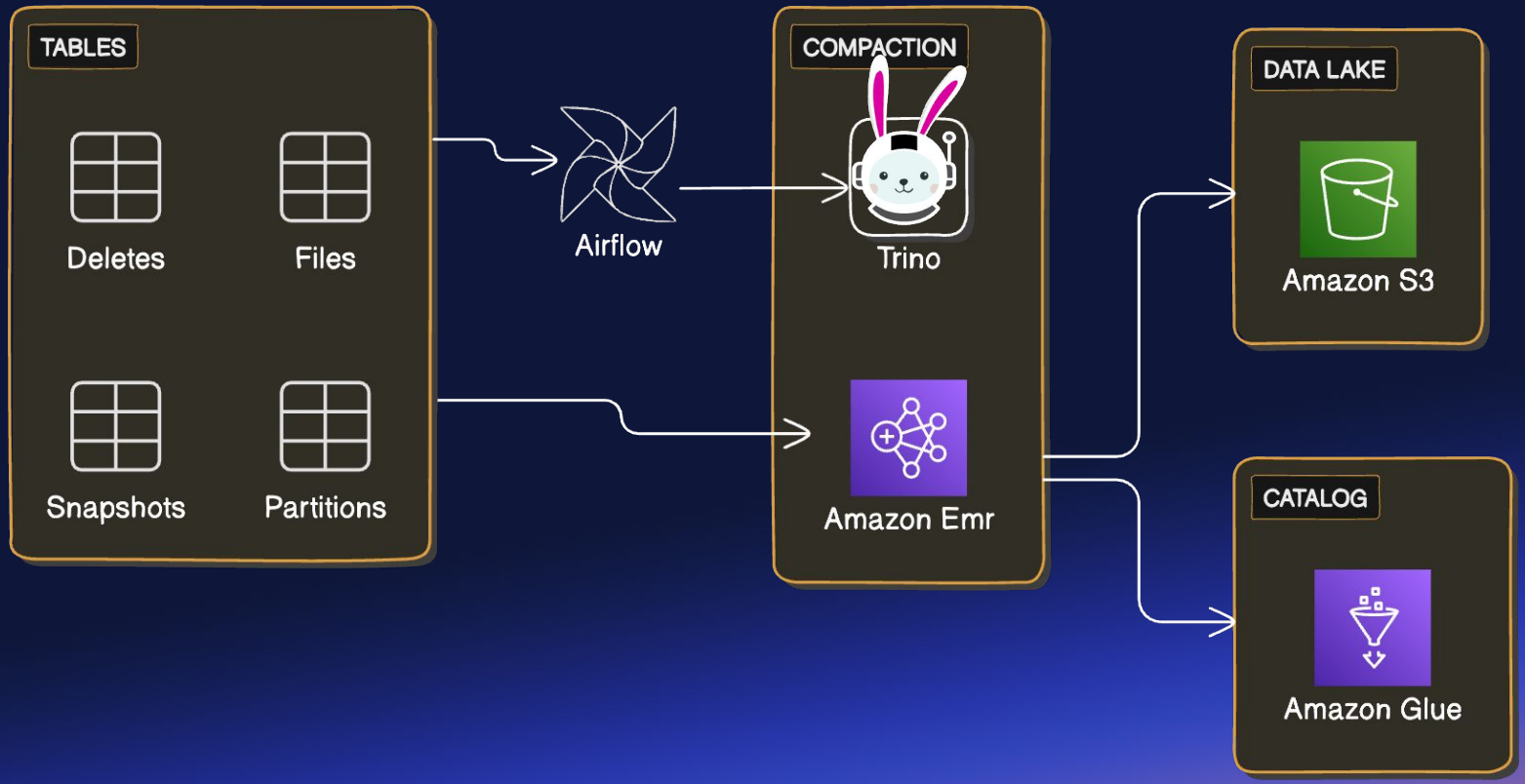
- Medium operation
- Range based shuffle
- Efficient read against sorted columns

Z-order

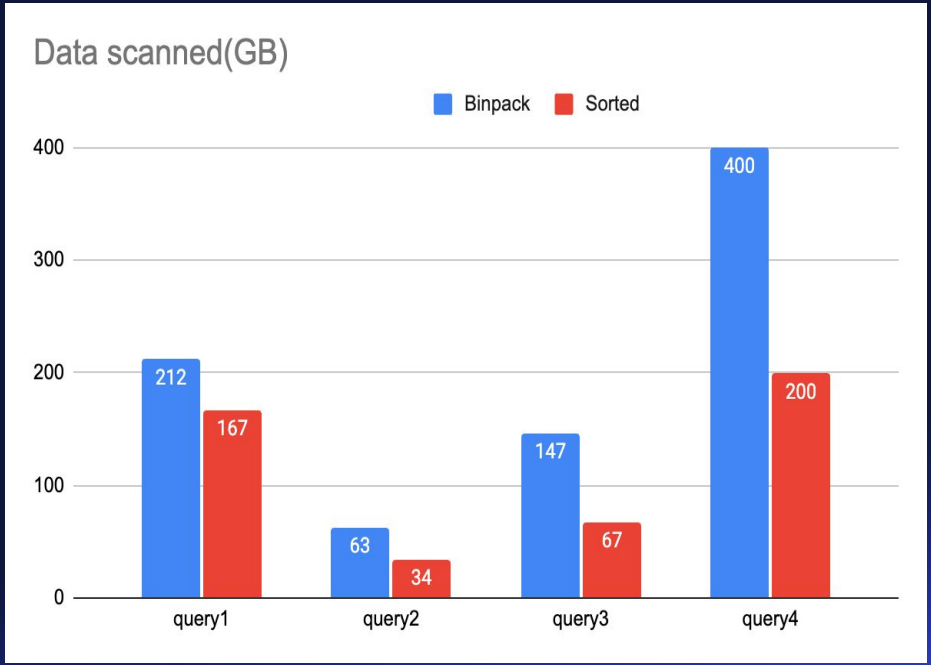
Cluster the data in targeted partitions based on **multiple columns**

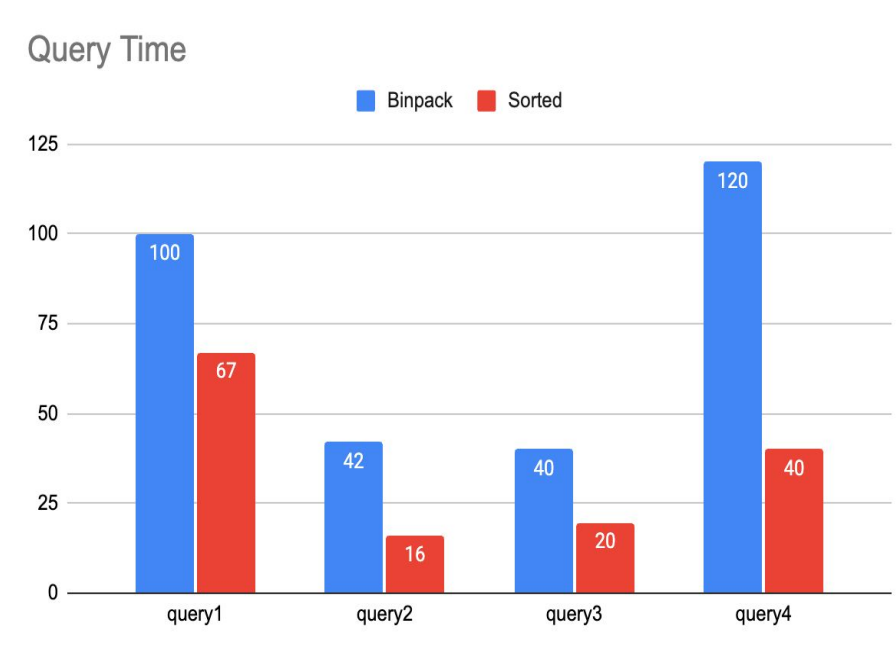
- Expensive operation
- Consider Z-order when using filters on **multiple dimensions.**
- Columns with high cardinality are best suited for Z-ordering.

Compaction Architecture



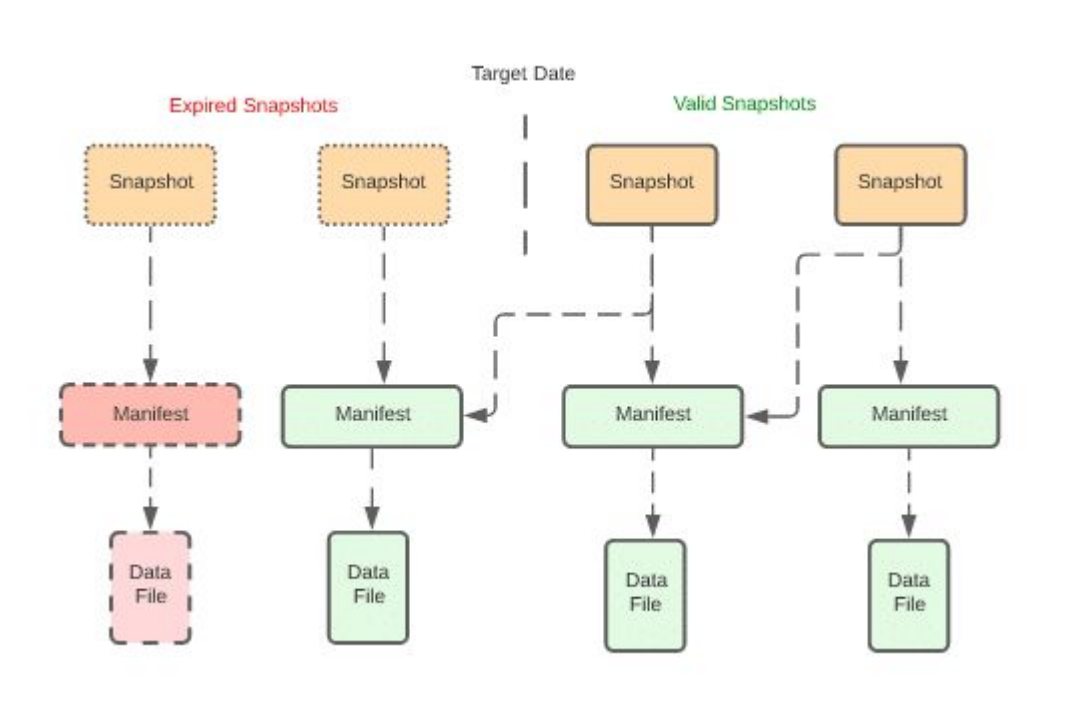
Binpack vs Sorting



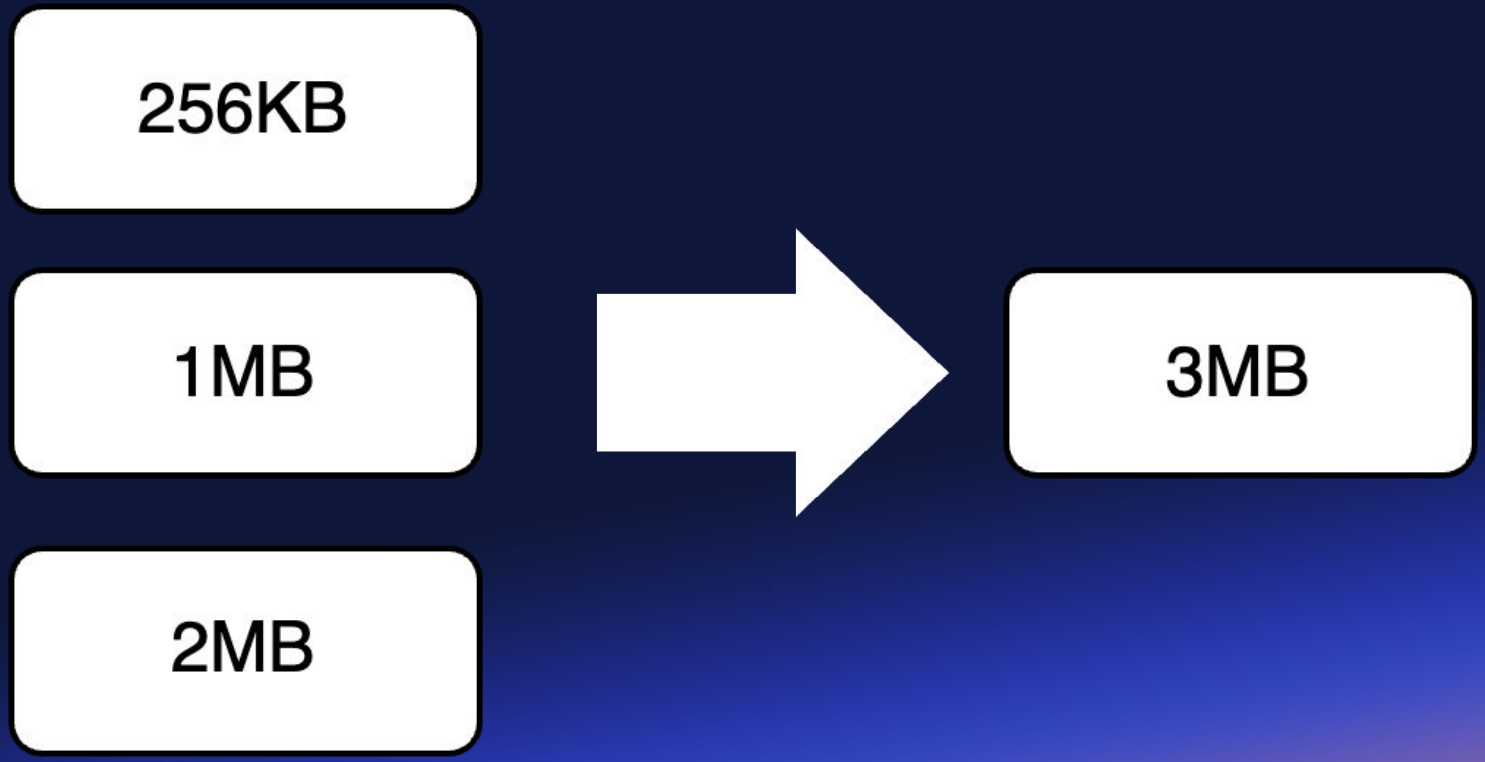


Maintenance

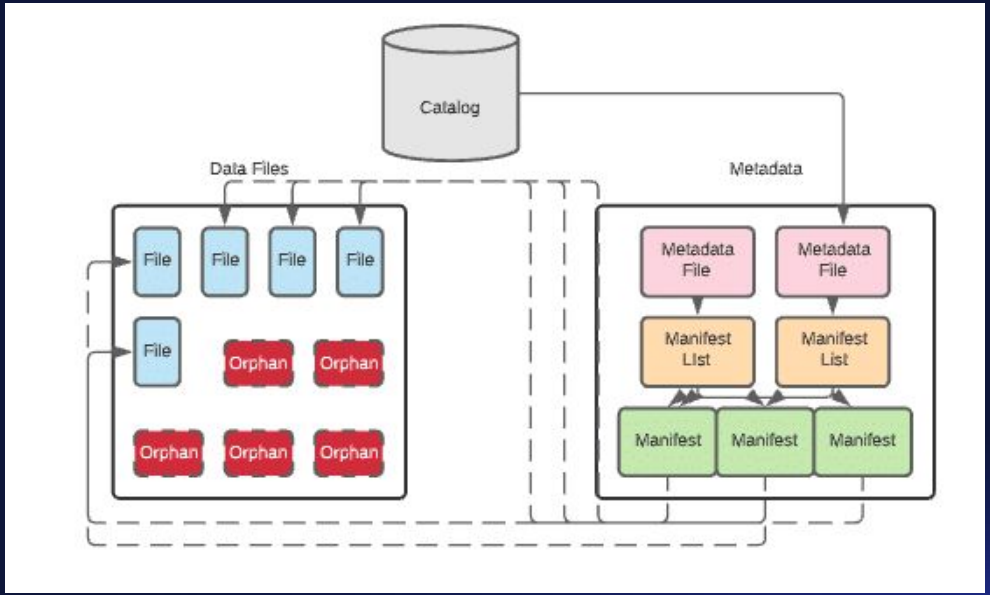
Expire snapshots

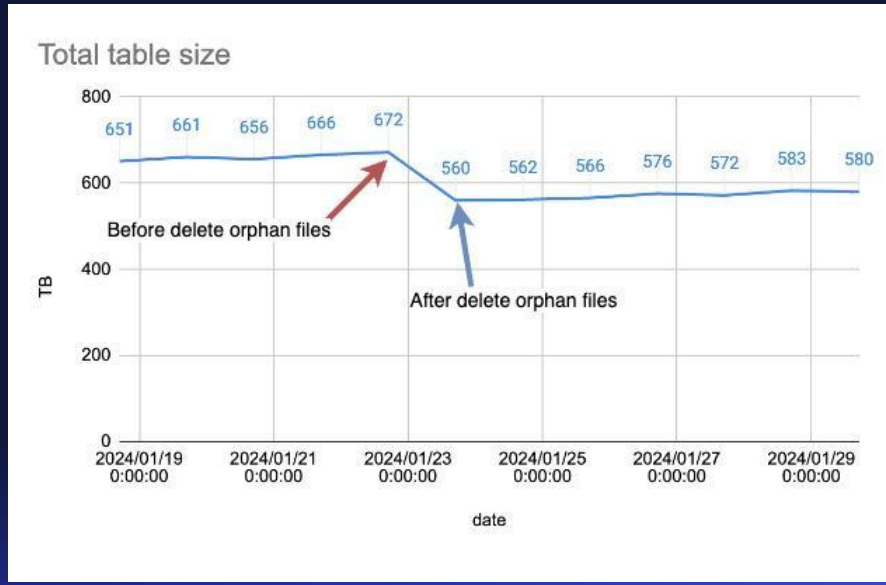


Rewrite manifests

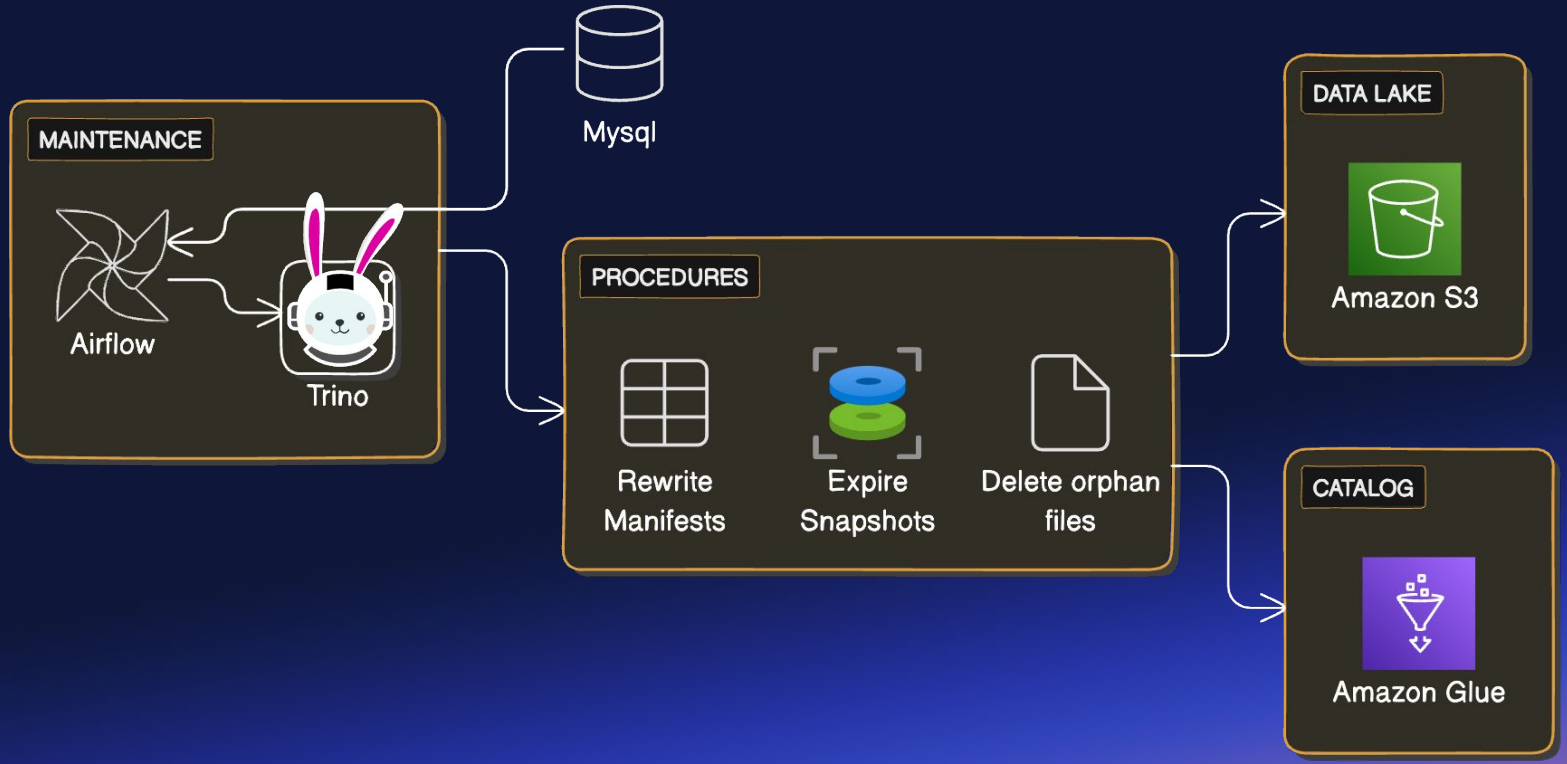


Delete orphan files





Maintenance Architecture

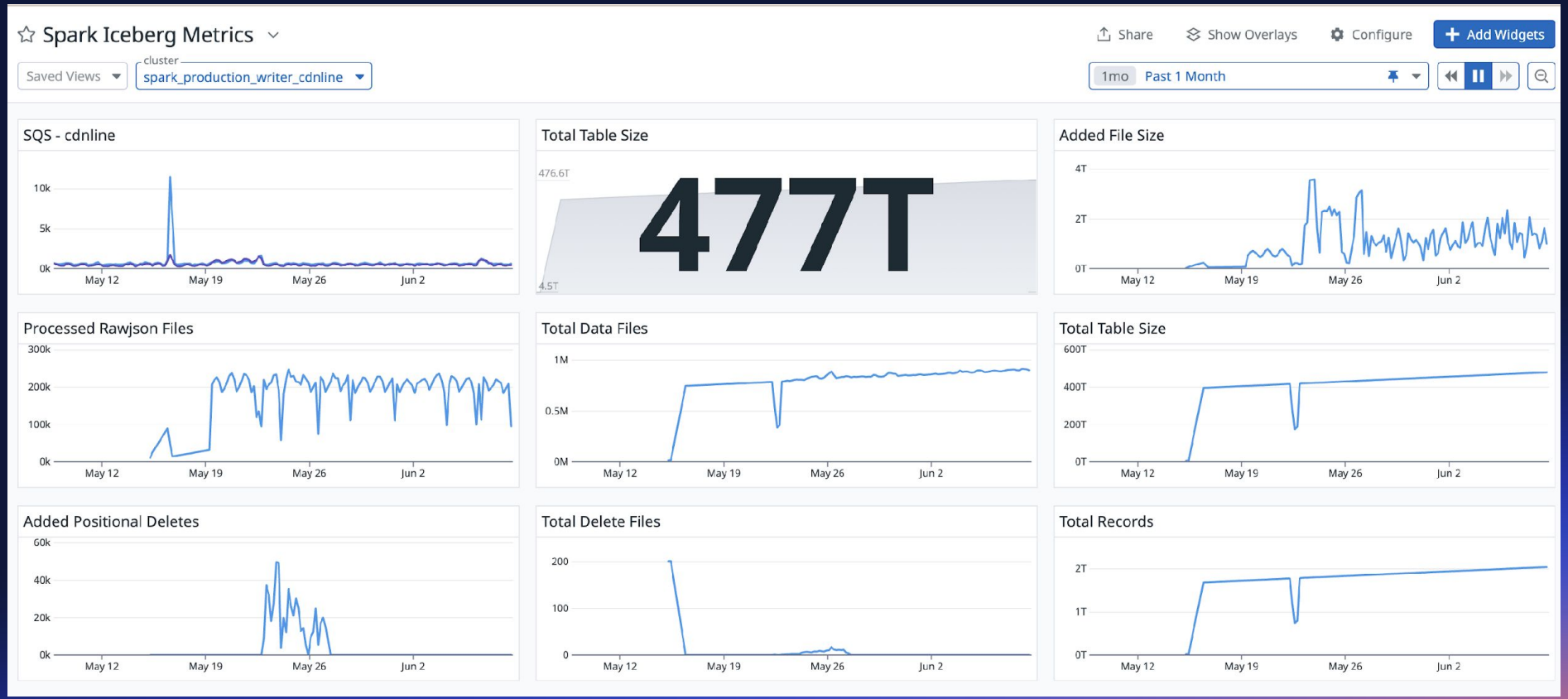


Monitoring

Metadata tables- (snapshots)

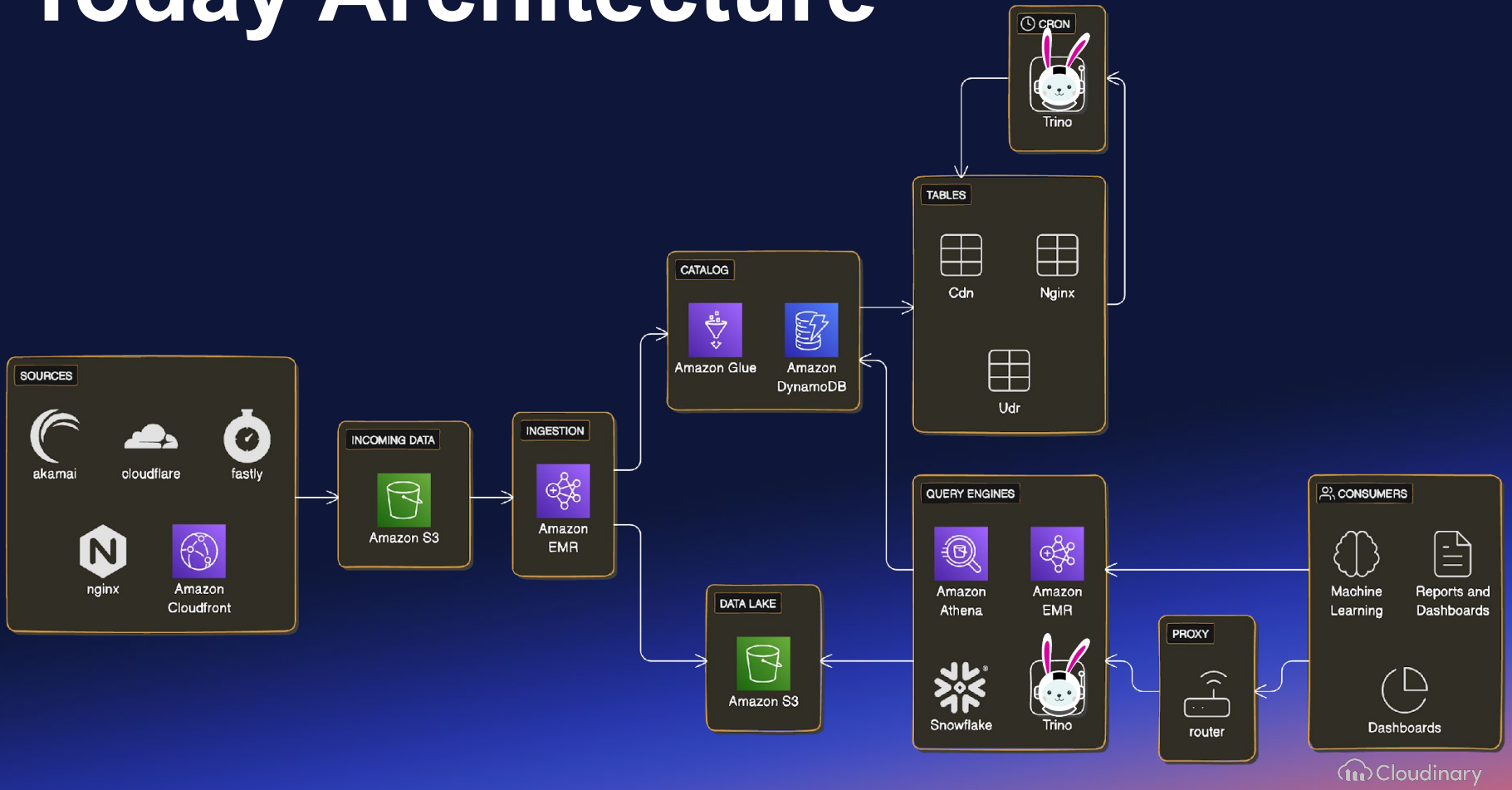
committed_at_hour	operation	added_records	added_data_files	deleted_data_file	size_gb	avg_file_size_mb
2024-03-10 21:00:00.000 UTC	replace	81026952	40	679	19	505
2024-03-10 21:00:00.000 UTC	append	94690651	781		21	29
2024-03-10 20:00:00.000 UTC	replace	704952135	333	5742	165	510
2024-03-10 20:00:00.000 UTC	append	700412961	5824		161	29
2024-03-10 19:00:00.000 UTC	append	894800167	7485		192	27
2024-03-10 19:00:00.000 UTC	replace	1024656767	458	8451	229	513
2024-03-10 18:00:00.000 UTC	replace	809592621	361	6868	180	513

Metrics reporter api



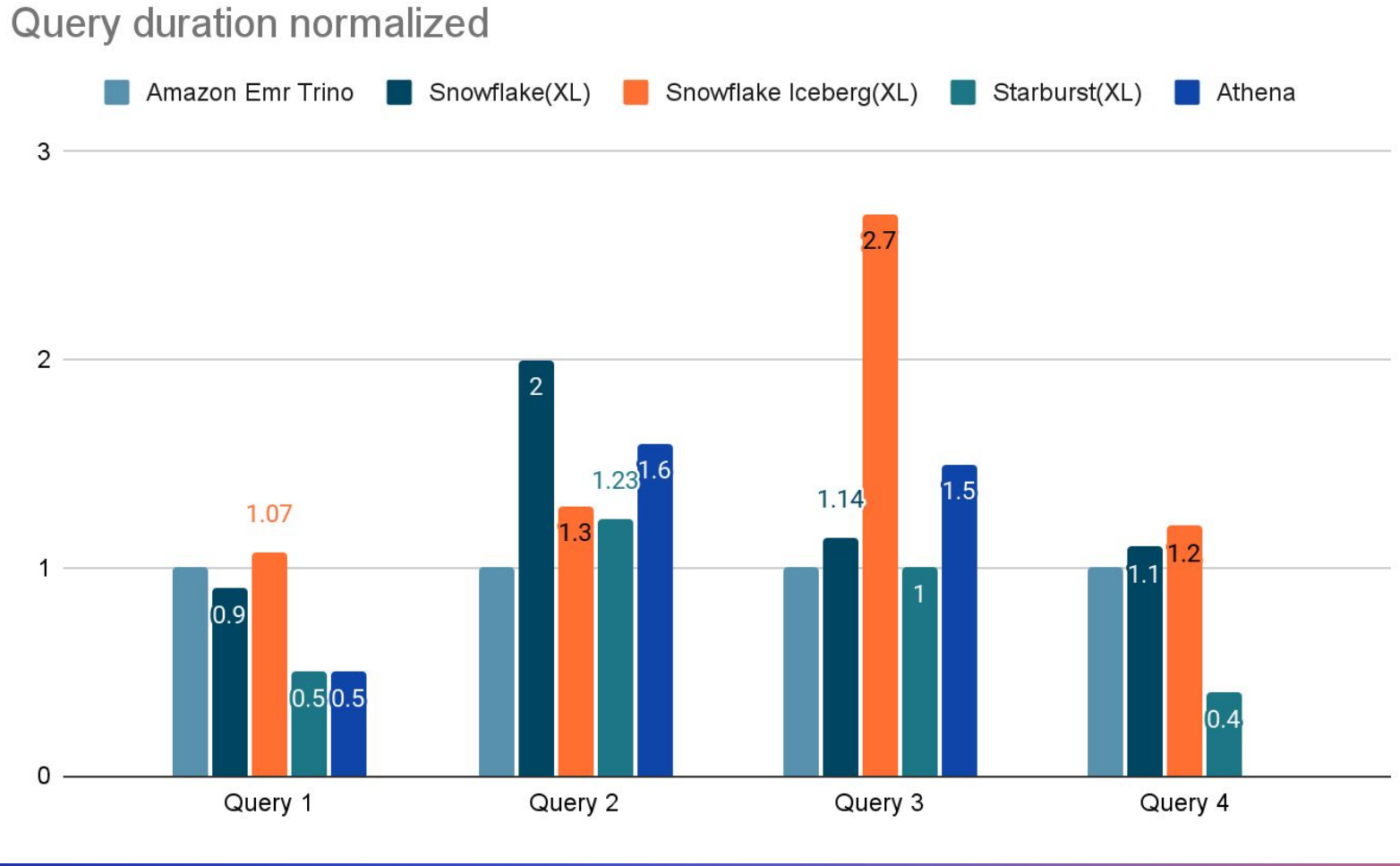
Cloudinary

Today Architecture



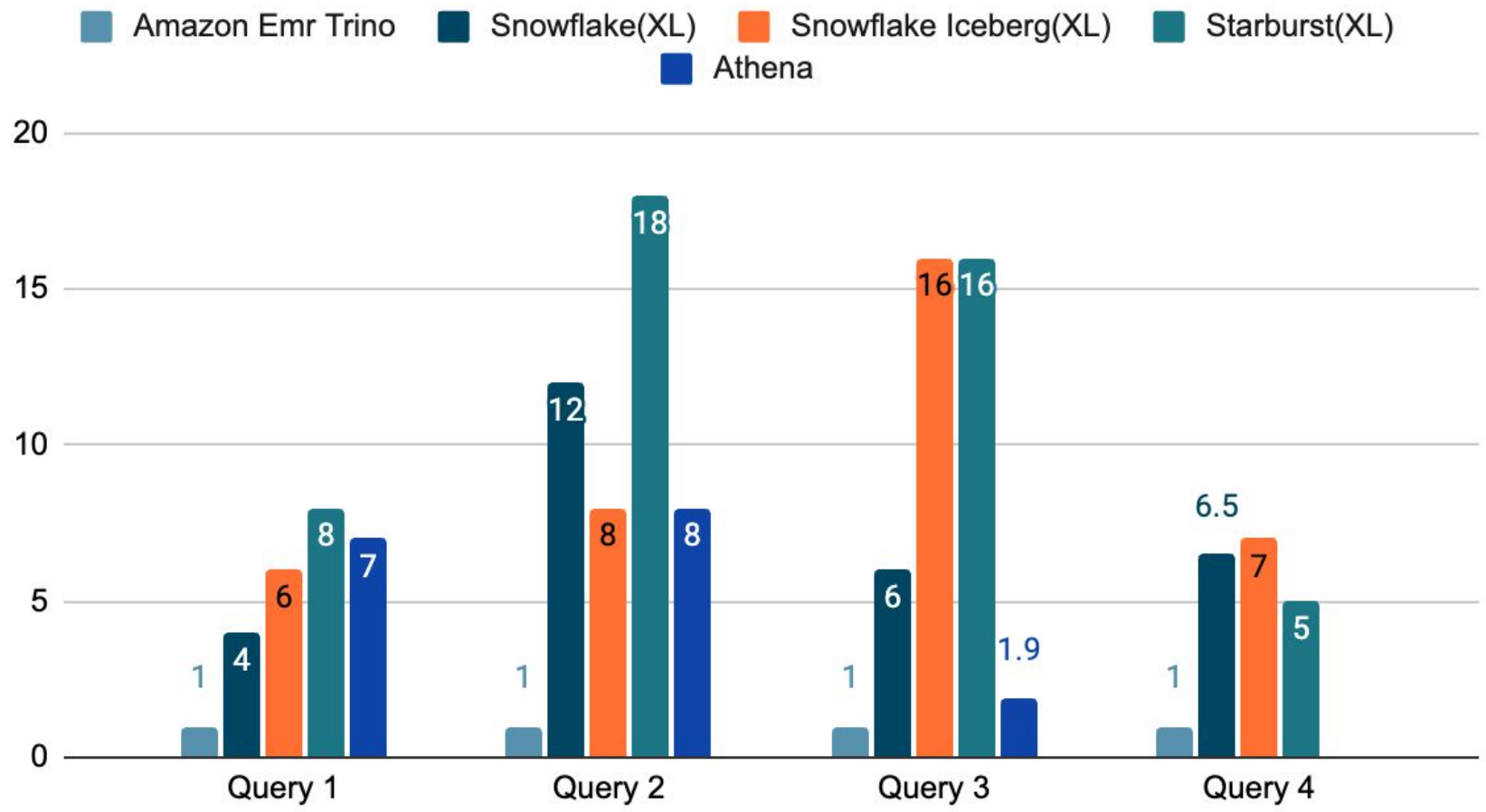
Benchmarks

Cloudinary


Query duration normalized
Amazon Emr Trino
Snowflake(XL)
Snowflake Iceberg(XL)
Starburst(XL)
Athena
3
2
1
0
1.07
0.9
0.5
0.5
2
1.3
1.23
1.6
1.14
2.7
1
1.5
1.1
1.2
0.4
Query 1
Query 2
Query 3
Query 4

/0.4

Query cost normalized



Questions?
Thanks

Amit Gilad