Unified data platform with Clickhouse

Gaurav Nigam, SMTS, Nutanix
Sachidananda Maharana, MTS 4, Nutanix

Gaurav Nigam

- SMTS at Nutanix
- Distributed system fanatic
- Works on Backend system, Distributed time-series DB

linkedin.com/in/gauravjpn/

Sachidananda Maharana

- Platform Engineer at Nutanix
- Loves Open source
- Works on Distributed OLAP databases

linkedin.com/in/sachidanandamaharana/

https://www.flaticon.com/free-icon/linkedin_174857

Legal Disclaimer

Forward Looking Statements

This presentation and the accompanying oral commentary may contain express and implied forward-looking statements, including, but not limited to, statements relating to: our business plans, initiatives and objectives; ability to execute such plans, initiatives and objectives in a timely manner, and the benefits and impact of such plans, initiatives and objectives, including our ability to manage our expenses in future periods; our financial targets and our plans to achieve those targets; the benefits and capabilities of our platform, products, services and technology; our plans and expectations regarding new products, services, product features and techno including those that are still under development or in process; the timing of any product releases or upgrades or announcements; anticipated trends, growth rates and challenges in our business and in the markets in whi operate; our ability to develop new solutions, product features and technology and bring them to market in a timely manner, as well as the impact and/or benefits of including additional solutions or features in our produ portfolio; market acceptance of new technology and recently introduced solutions; the interoperability and availability of our solutions with and on third-party platforms, including public cloud platforms; our ability to ma and strengthen our relationships with our channel partners, OEMs and other third parties, and the impact of any changes to such relationships on our business, operations and financial results; the competitive market, inc our competitive position and ability to compete effectively, our projections about our market share in future periods, the competitiveness of our future cost structure with those of other companies, and the competitive advantages of our products; our plans and timing for, and the success and impact of, our transition to a subscription-based business model; and macroeconomic trends and geopolitical environment, including the on-go global supply chain disruptions.

These forward-looking statements are not historical facts and instead are based on our current expectations, estimates, opinions, and beliefs. Forward-looking statements should not be considered as guarantees or pred of future events. Consequently, you should not rely on these forward-looking statements. The accuracy of these forward-looking statements depends upon future events and involves risks, uncertainties, and other factor including factors that may be beyond our control, that may cause these statements to be inaccurate and cause our actual results, performance or achievements to differ materially and adversely from those anticipated o implied by such statements, including, among others, the risks detailed in our most recent Annual Report on Form 10-K and Quarterly Reports on Form 10-Q, each as filed with the U.S. Securities and Exchange Commissi the SEC, which should be read in conjunction with the information in this presentation and accompanying oral commentary. Our SEC filings are available on the Investor Relations section of our website at ir.nutanix.com a the SEC's website at www.sec.gov. These forward-looking statements speak only as of the date of this presentation and accompanying oral commentary and, except as required by law, we assume no obligation, and exp disclaim any obligation, to update, alter or otherwise revise any of these forward-looking statements to reflect actual results or subsequent events or circumstances.

Product or Roadmap Information

Any future product or roadmap information included in this presentation and the accompanying oral commentary is (i) intended to outline general product directions, (ii) not a commitment, promise or legal obligation fo Nutanix to deliver any material, code, or functionality, and (iii) not intended to be, and shall not be deemed to be, incorporated into any contract. This information should not be used when making a purchasing decision. note that Nutanix has made no determination as to whether separate fees will be charged for any future products, product enhancements and/or functionality which may ultimately be made available. Nutanix may, in its discretion, choose to charge separate fees for the delivery of any future products, product enhancements and/or functionality which are ultimately made available.

Third Party Reports and Publications

Certain information contained in the presentation and accompanying oral commentary made available as part of this digital event may relate to or be based on reports, studies, publications, surveys and other data obtai from third-party sources and our own internal estimates and research. While we believe these third-party reports, studies, publications, surveys and other data are reliable as of the date of the applicable presentation, the not independently verified, and we make no representation as to the adequacy, fairness, accuracy, or completeness of any information obtained from third-party sources.

Trademark Disclaimer

Show of Hands

- Have you heard about on-premise/private cloud?
- Do you know about Nutanix ?
- Have you used clickhouse ?
- Any other analytics database ?

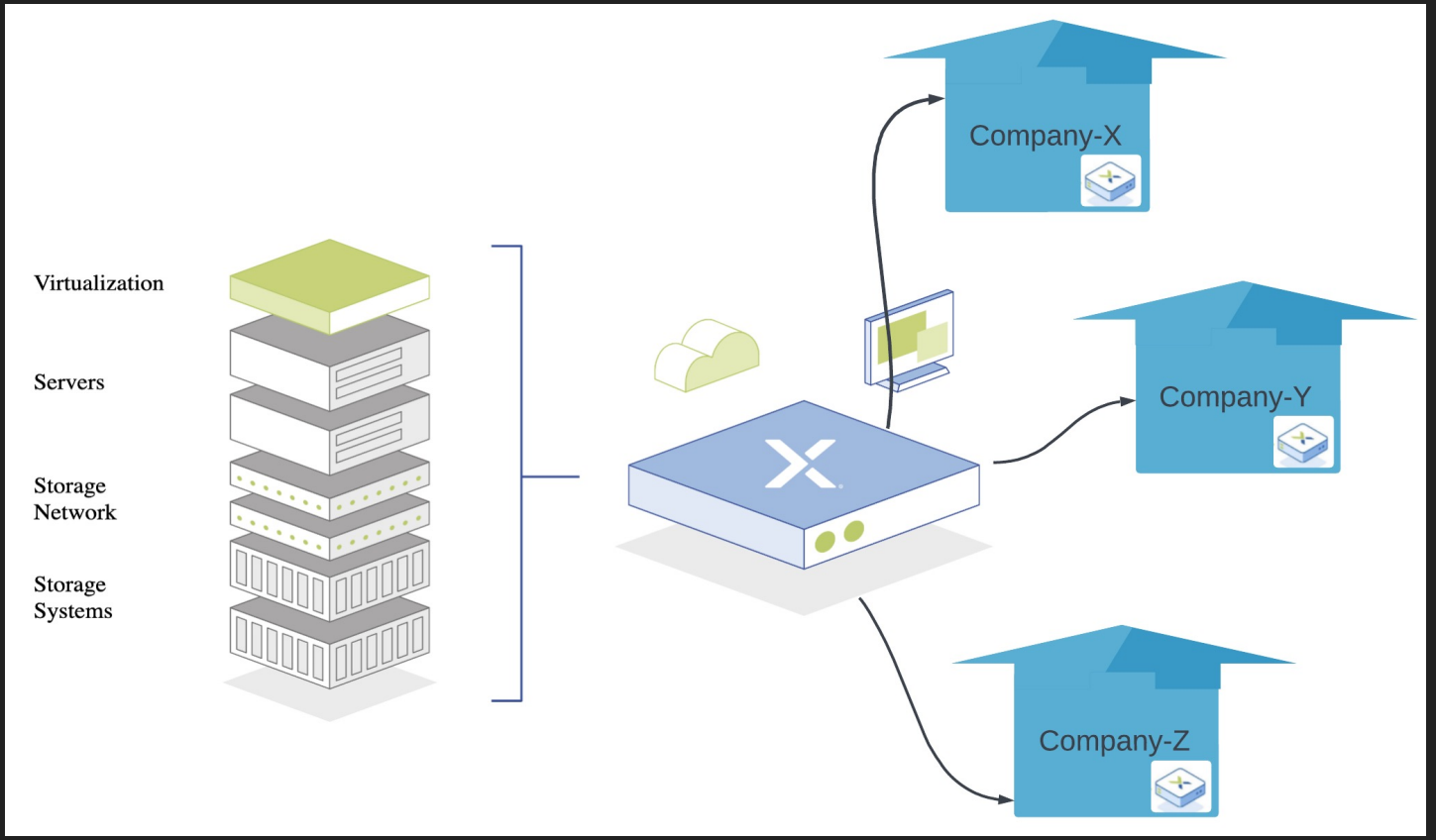
https://www.youtube.com/watch?v=6O7-nzXPAzs
https://www.nutanix.com/company/customers

Agenda

- Setting the context
- Need for Unified Data platform
- Workload and use cases
- Challenges
- Q & A

About Nutanix

- Known for on-premise business
- Provides modern and cost-effective solution to manage data-center

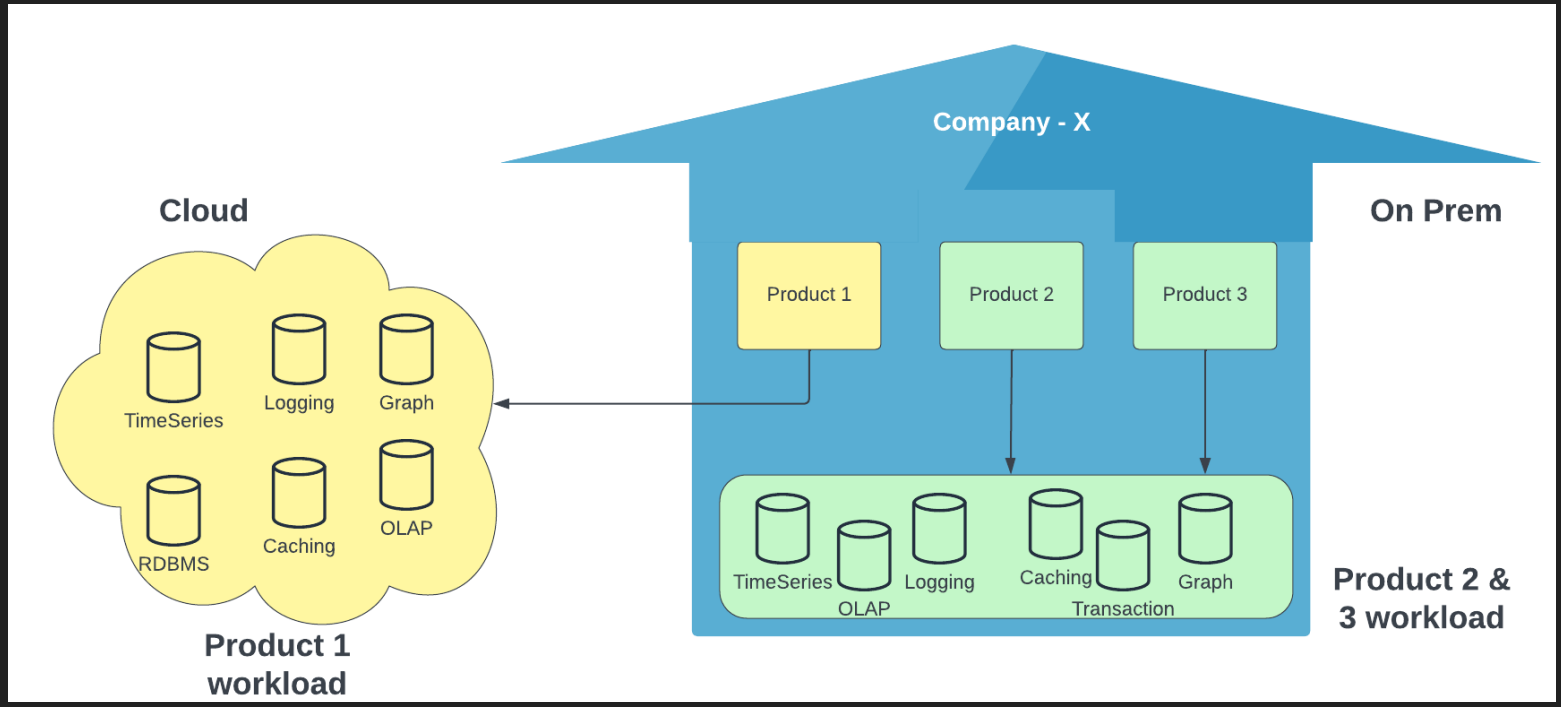


& many more

https://www.youtube.com/watch?v=6O7-nzXPAzs
https://www.nutanix.com/company/customers

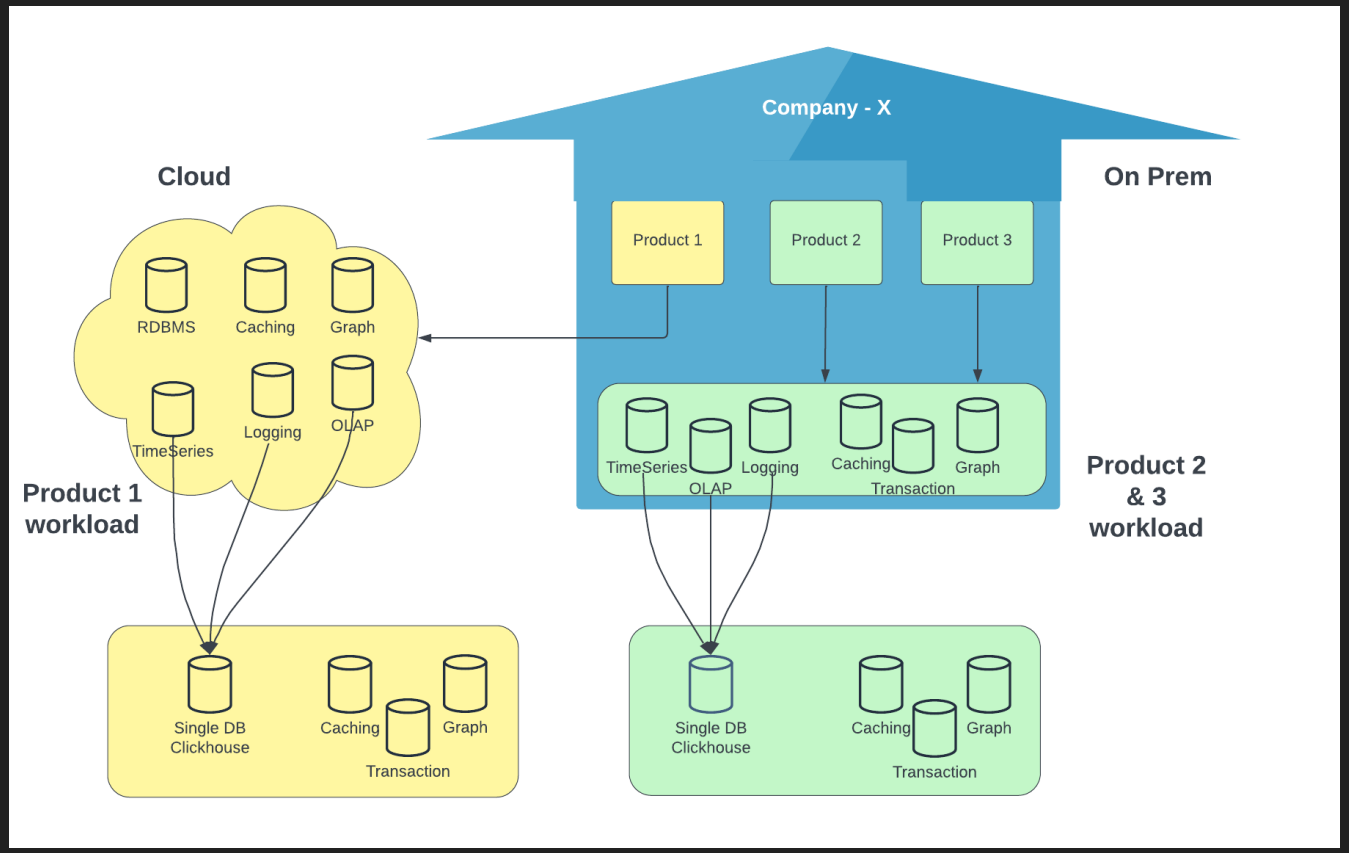
Setting the context

- Multiple database for different use case
  - Analytics – Druid
  - Timeseries - Victoria Metrics, Inhouse DB
  - Logging and tracing – Elastic search



Need for unified data platform

- On-prem and Cloud database challenges
  - So many databases
  - So many components per database
  - Resource intensive
  - Cost intensive
  - Redundant data
- Pros of Unified data platform
  - Ease of Management
  - Less time for day 1 activities
  - Cost effective
  - Resource efficient



Unified data platform

Workload Numbers

Analytics

Data size	#Data sources	# Ingestion tasks (24 hrs)	# Queries (24 hrs)
~20 TB	294	423k	~8m

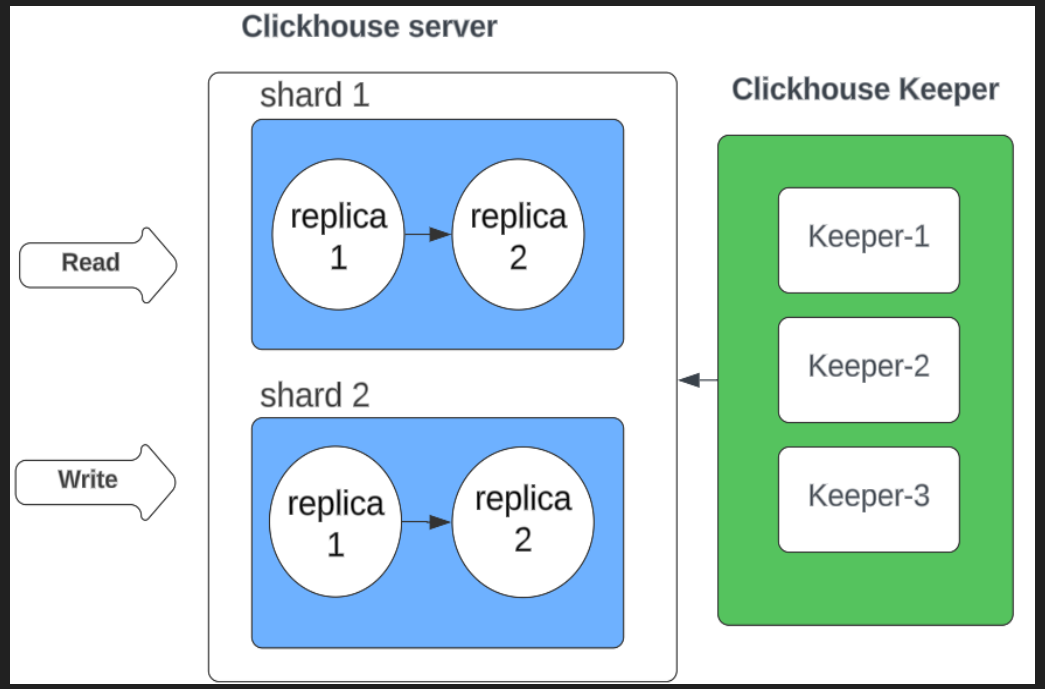
Logging and Tracing

Use case	Data size	Retention
Logging	~60TB	15 days, rest s3
Tracing	~1TB	3 days, rest s3

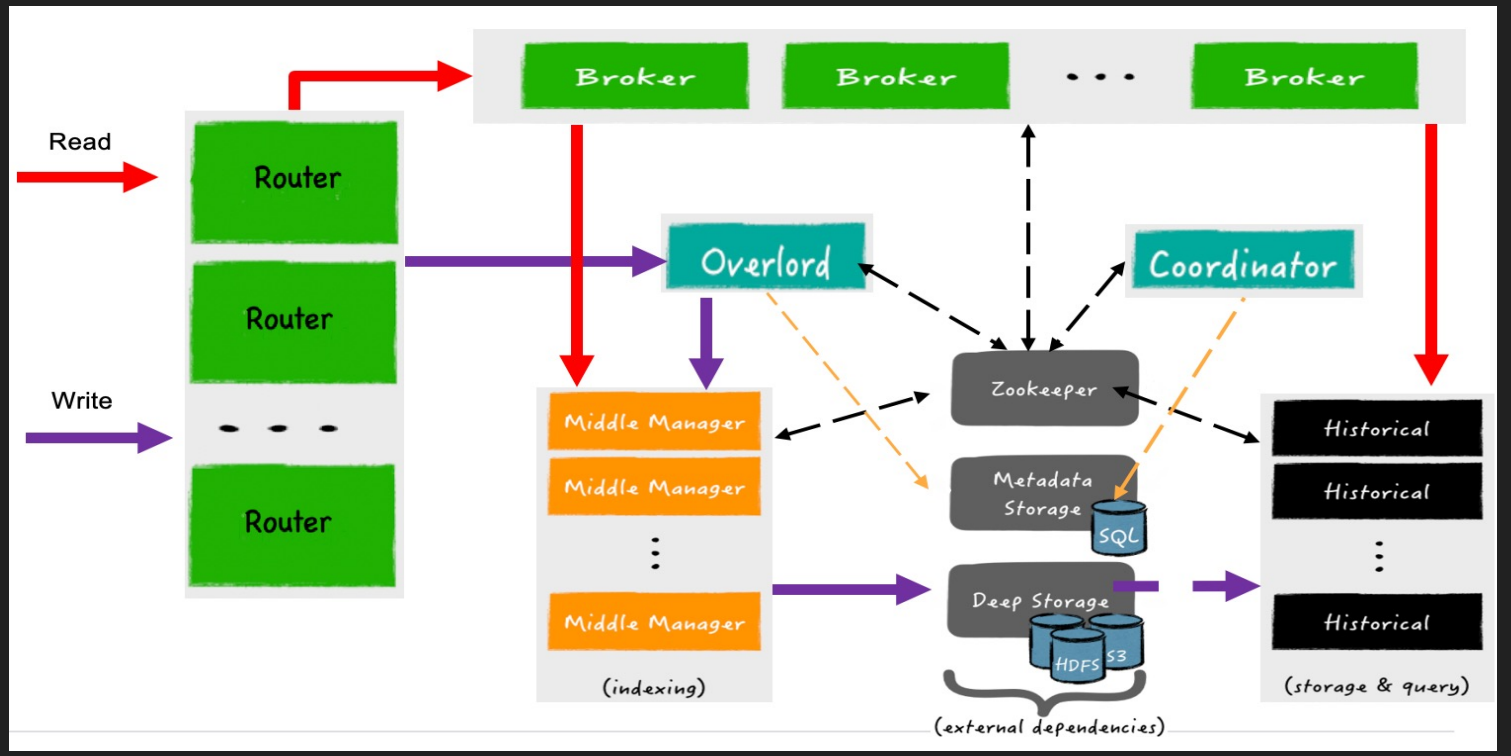
Time Series

Use case	Data size	Active Time series	Data points per 5min
X-Large (current) on-prem	3-5 TB	25m	25m
X-Large (future) on-prem	10-15 TB	50m	110m
Cloud	~7TB	~22m	~150m

Analytics: Clickhouse vs Druid



- 2 components
- Less maintenance
- EBS vs instance store



- 9 components - Resource hungry
- More maintenance in on-prem
- Read-write segregation
- Data rebalance

Analytics: Clickhouse vs Druid

	Clickhouse	Druid	Improvement
P99 Ingestion Task Latency	34.989s(16 parallel tasks)	68.688s*	~50 %**
Memory avg***	4.6 GiB	16 GiB	
CPU avg	1.33 vCPU	6.4 vCPU	

Use case 1:
53.3 million rows
1 month data

	Clickhouse	Druid	Improvement
Ingestion Task Latency	~4.5hrs(with 4 parallel tasks) total time	~9hrs(total time)	~50 %**

Use case 2:
1.2 billion rows
13 months data

*Doesn't include compaction time for druid
**Numbers are from perf environment
***Include resource requirement for Middle manager

Show of Hands

- Have you used any time series DB?
- How many of you have used clickhouse for time-series use case?

https://www.youtube.com/watch?v=6O7-nzXPAzs
https://www.nutanix.com/company/customers

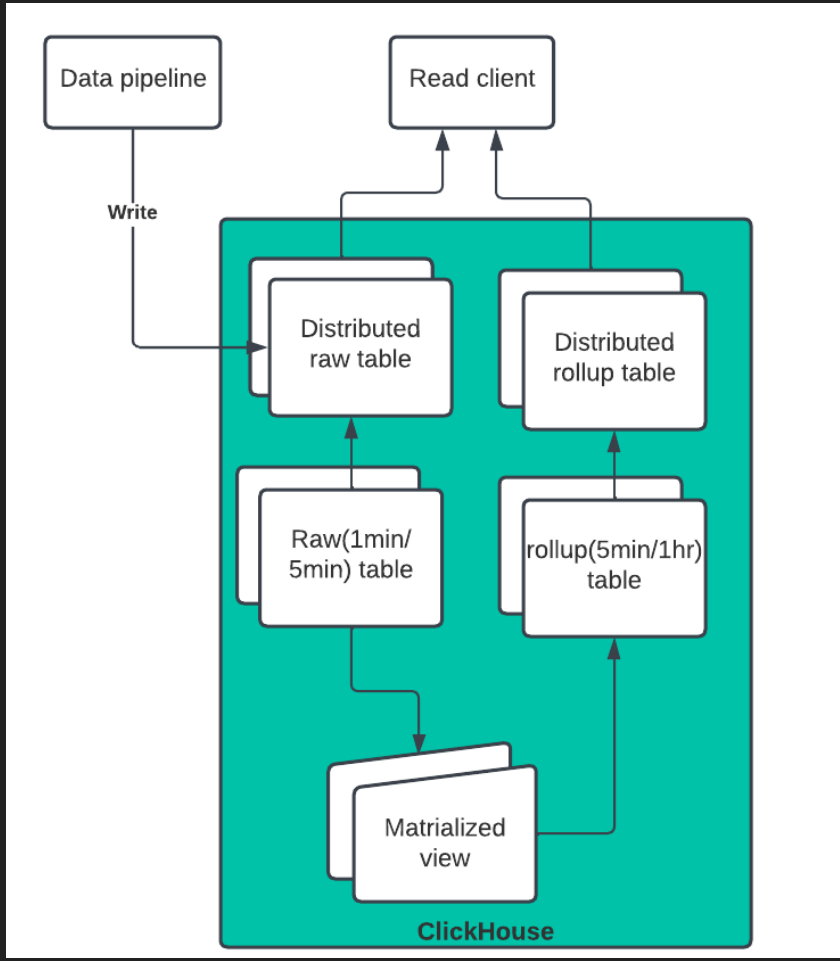
Time Series: Use case

- ClickHouse schema modelling to fit timeseries use case
- Optimized for time-series query patterns and desired latency and QPS
- Optimized the ingestion for our scale
- Optimized the storage

Requirements

Ingestion	Query
• High ingestion rate (40m data points per min) • Batch ingestion • No fix batching • Multiple clients with different rate • Historical ingestion	• 60 Queries per second • Latency requirement(P80) of <200ms • Mix for Time-series and OLAP query

Time Series: ClickHouse Schema



- Distributed table
  - Sharding logic
- Raw tables (ReplicatedMergeTree)
  - Keeps raw data
  - 1-month data (5 min granularity)
- Rollup tables (ReplicatedMergeTree)
  - Populated using materialized view
  - Keeps aggregated data
  - 1-year data (hourly granularity)
- Materialized view
  - Aggregation logic

Time Series: ClickHouse Schema

```
CREATE TABLE test.my_table ON CLUSTER 'test-cluster' (
    timestamp DateTime DEFAULT now(),
    id String,
    type String,
    ....
    domain_id String,
    metric_name String,
    metric_value Float64,
)
ENGINE = ReplicatedMergeTree('<...>', '<...>')
PARTITION BY toYYYYMM(toDate(timestamp))
ORDER BY (domain_id, type, metric_name, id, timestamp)
TTL timestamp + INTERVAL 3 WEEK
```

Order-by Should depend on

- Cardinality (Low to High)
- Query pattern

Time Series: ClickHouse Schema

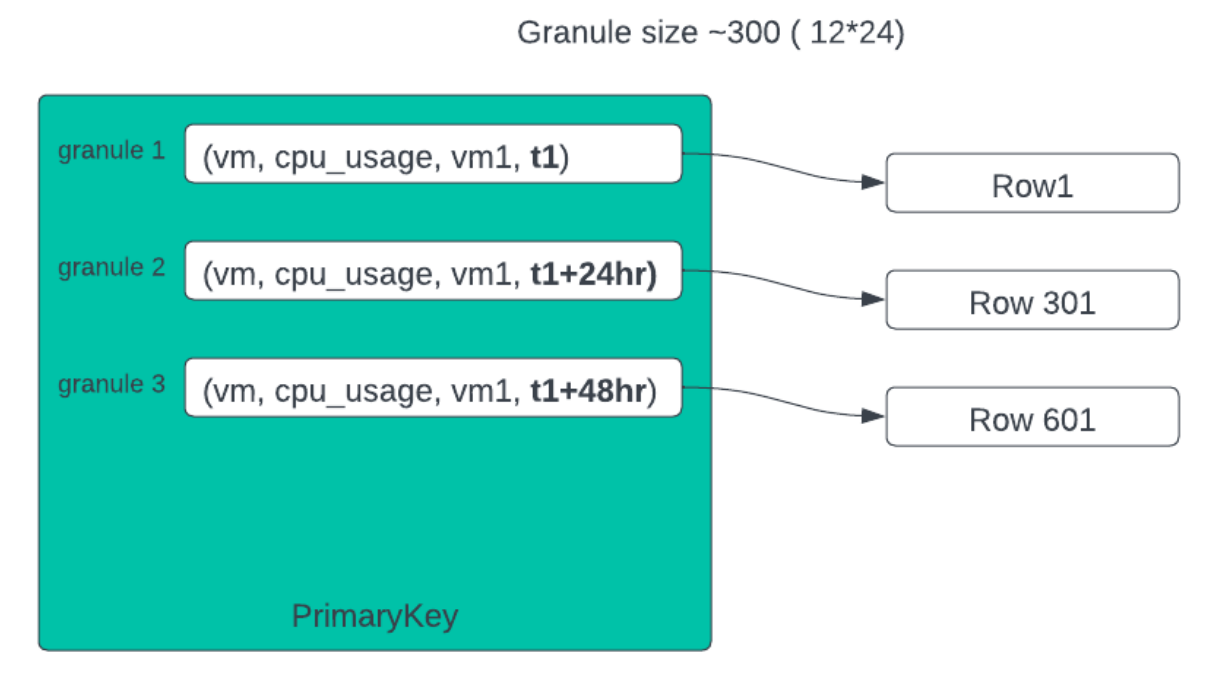
```
CREATE TABLE test.my_table ON CLUSTER 'test-cluster' (
    timestamp DateTime DEFAULT now(),
    eid String,
    etype String,
    domain_id String,
    metric_name String,
    metric_value Float64,
)
ENGINE = ReplicatedMergeTree('<...>', '<...>')
PARTITION BY toYYYYMM(toDate(timestamp))
ORDER BY (domain_id, etype, metric_name, eid, timestamp)
TTL timestamp + INTERVAL 3 WEEK
SETTINGS index_granularity = 300, index_granularity_bytes = 0;
```

Refined Granule Size

Less data scanned

Faster query

Optimizing ClickHouse Query



- Every granule contains 1 day data
- With default granule size(8192) – 28 days data

A granule refers to smallest unit of data storage in a clickhouse table.

Time Series: ClickHouse Schema

```
CREATE TABLE test.my_table ON CLUSTER 'test-cluster' (
    timestamp DateTime DEFAULT now(),
    __timestamp DateTime MATERIALIZED toStartOfInterval(timestamp, INTERVAL 60 second)
    id String,
    type String,
    domain_id String,
    metric_name String,
    metric_value Float64,
)
ENGINE = ReplicatedMergeTree('<...>', '<...>')
PARTITION BY toYYYYMM(toDate(__timestamp))
ORDER BY (domain_id, type, metric_name, id, __timestamp)
TTL timestamp + INTERVAL 3 WEEK
SETTINGS index_granularity = 300, index_granularity_bytes = 0;
```

Normalized timestamp column

Improved Data quality and aggregation

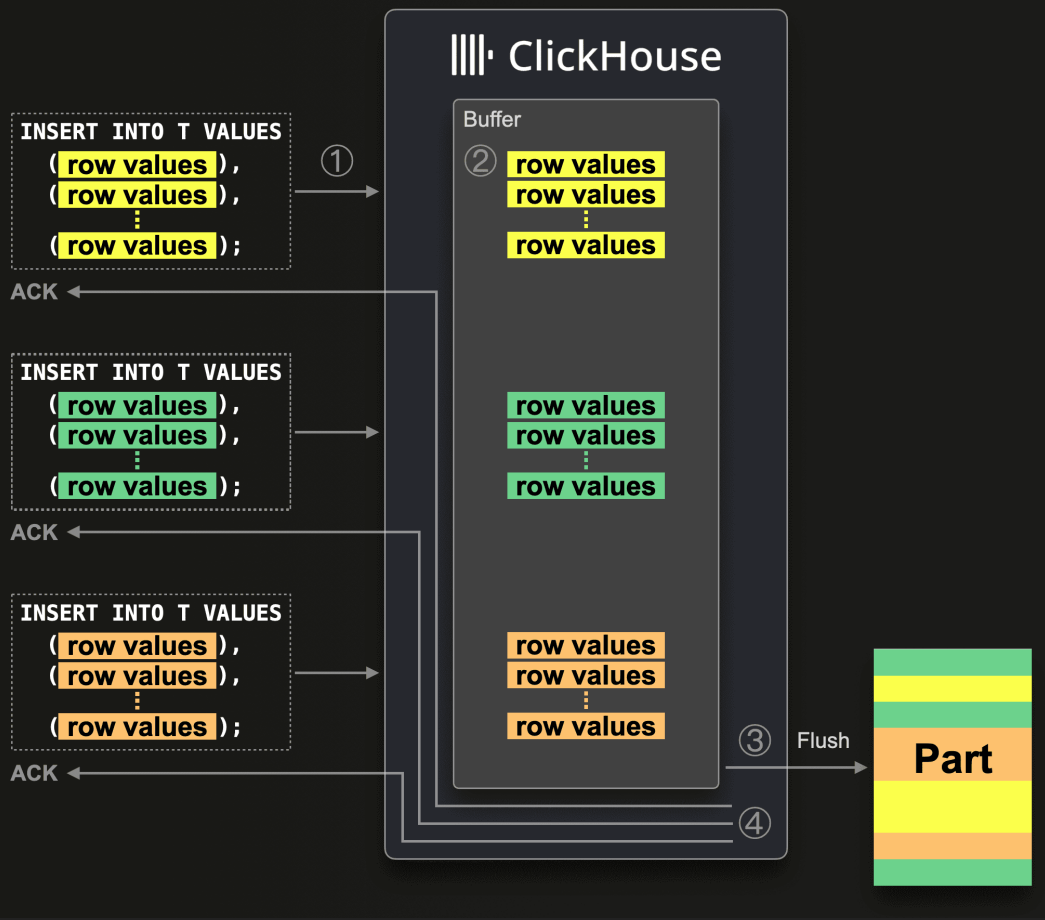
Time Series: ClickHouse Schema

```
CREATE TABLE test.my_table ON CLUSTER 'test' (
    timestamp DateTime DEFAULT now() CODEC(DoubleDelta),
    __timestamp DateTime MATERIALIZED
            toStartOfInterval(timestamp, INTERVAL 300 second) CODEC(DoubleDelta),
    eid FixedString(36),
    etype LowCardinality(String),
    domain_id LowCardinality(FixedString(36)),
    metric_name LowCardinality(String),
    metric_value Float64 CODEC(Gorilla),
)
ENGINE = ReplicatedMergeTree('<...>', '<...>')
PARTITION BY toYYYYMM(toDate(__timestamp))
ORDER BY (domain_id, etype, metric_name, eid, timestamp)
TTL timestamp + INTERVAL 3 WEEK
SETTINGS index_granularity = 300, index_granularity_bytes = 0;
```

Applied compression and encoding

Improved storage in turn improves query latency

Ingesting into ClickHouse for Time Series



Achieved 25% improvement in ingestion using native Async

① async_insert = 1
② wait_for_async_insert = 1
③ async_insert_busy_timeout_ms = 200
④ async_insert_max_data_size = 1_000_000
⑤ async_insert_max_query_number = 20

https://clickhouse.com/blog/asynchronous-data-inserts-in-clickhouse

Ingesting into ClickHouse for Time Series

	Standard sync	Native Sync	Async insert	Native async v23.10+
driver	database/sql driver	Clickhouse-go native driver	Clickhouse-go native driver	Clickhouse-go native driver
Batching support	Yes, using prepared stmt	Yes, using prepareBatch	No, supported using inline sql.	Yes, using preparedBatch
ingestion performance	Baseline	20% improved	Perf degraded	25% improved

Ingestion	Data points (1min scale)	Batch size	Total parts	Ingestion time
Native Async	17m(~900mb)	~10000	1365	~38 sec
Native Sync	17m(~900mb)	~10000	1530	~48 sec

Challenges & Learnings

- Rebalancing after adding more shards
- Read-write segregation
- Lots of knobs (swiss-knife), know your configs!
- Replication & sharding in clickhouse has a steep learning curve.
- Use the Altinity Kubernetes Operator for Replication & Sharding
- If you only have time-series use case, a dedicated Time series DB might be easy

Thank You

Q & A