

ClickHouse Cloud

Live

Feb 8, 2024



Agenda (60 mins)

Welcome (Tanya) - 3 mins

Cloud Platform Update (Krithika) - 5 mins

Cloud Console Update (Zach, Cristina) - 15 mins

Security & Reliability Update (Leticia) - 7 mins

Core Update (Melvyn and/or Alexey) - 10 mins

Integrations Update (Ryadh) - 10 mins

Q&A - 10 mins

For live social update links/polls, please add them to your section [here](#).

In red, topics that could be removed to save time



ClickHouse Cloud



Soon

Metrics Backups Connection Settings

Running

Location Ohio (us-east-2) Version ClickHouse 23.8

Last successful backup 6 hours ago

Created Oct 18, 2022

Data stored **650.99 GB**

Increase of 9 GB in last week

Data stored over time

Memory allocation **720 GiB**

Total read size **2.21 TB**

Average read size per second: 3.65 MB

Read throughput (bytes per second)

Total inserts **5 M**

Average inserts per second: 8.3

Total selects **0%**

Total inserts **100%**

SQL statements over time

Successful queries **100%**

Failed queries **0%**

Expensive properties

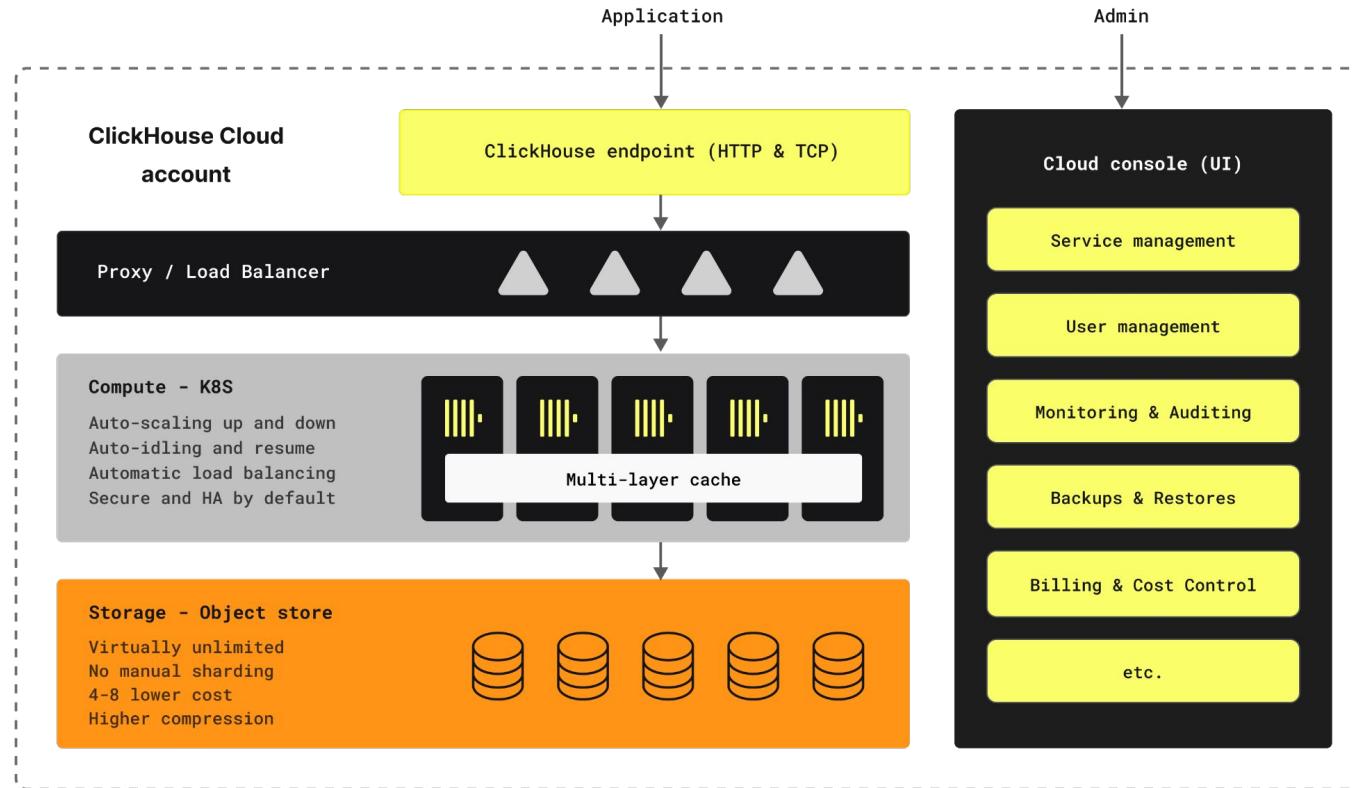
```
-- the price, street and district of the most expensive properties in London in 2020 where the post code starts with W1S
SELECT price, time, street, addr1, addr2, district, postcode
FROM pp_complete
WHERE town ILIKE '%London%' AND postcode1 ILIKE 'W1S%' AND time LIKE '2020%'
ORDER BY price DESC
LIMIT 10;
```

Search results... Elapsed: 0.785s Read: 27,450,499 rows (937.59 MB)

#	price	time	street	addr1	addr2
1	366180000	2020-10-08 00:00	NEW BOND STREET	159	
2	366180000	2020-10-08 00:00	NEW BOND STREET	158	
3	305211030	2020-10-08 00:00	NEW BOND STREET	158	
4	114712500	2020-07-02 00:00	NEW BOND STREET	141	
5	95086435	2020-10-07 00:00	SAVILE ROW	25	
6	56000000	2020-03-20 00:00	MADDOX STREET	23	
7	56000000	2020-03-20 00:00	MADDOX STREET	29	
8	56000000	2020-03-20 00:00	MADDOX STREET	27	
9	421080000	2020-06-30 00:00	SACKVILLE STREET	1	

All Rows

ClickHouse Cloud Architecture



Cloud Pricing

Development

Great for smaller workloads and starter projects

\$50 - \$193 / Month

- ✓ Up to 1 TB storage
- ✓ 16 GiB total memory
- ✓ Burstable CPU
- ✓ Backups every 24h, retained 1 day
- ✓ Replicated across 2 AZs
- ✓ Expert support with 24h response time

Storage

\$35.33 per TB/mo

Compute

\$0.2160 per unit/hr

Production

Designed to handle production workloads

Usage Based

- ✓ Unlimited storage
- ✓ 24GiB+ total memory
- ✓ Dedicated CPU
- ✓ Backups every 24h, retained 1 day
- ✓ Replicated across 3 AZs
- ✓ Expert support with 1h SLA
- ✓ AWS private link support
- ✓ Automatic scaling

Storage

\$47.10 per TB/mo

Compute

\$0.6888 per unit/hr

Dedicated

Designed for the most demanding latency-sensitive workloads

Capacity Based

- ✓ Unlimited storage
- ✓ Custom compute options
- ✓ Dedicated environment
- ✓ Advanced isolation and security
- ✓ Customized backup controls
- ✓ Scheduled upgrades
- ✓ Uptime SLAs
- ✓ Consultative migration guidance
- ✓ Dedicated support engineers

Contact us



Agenda

01

Cloud Platform

Krithika

02

Cloud Console

Zach, Cristina

03

Security & Reliability

Leticia

04

Core

Melvyn, Alexey

05

Integrations

Ryadh



Cloud Platform Update

Krithika



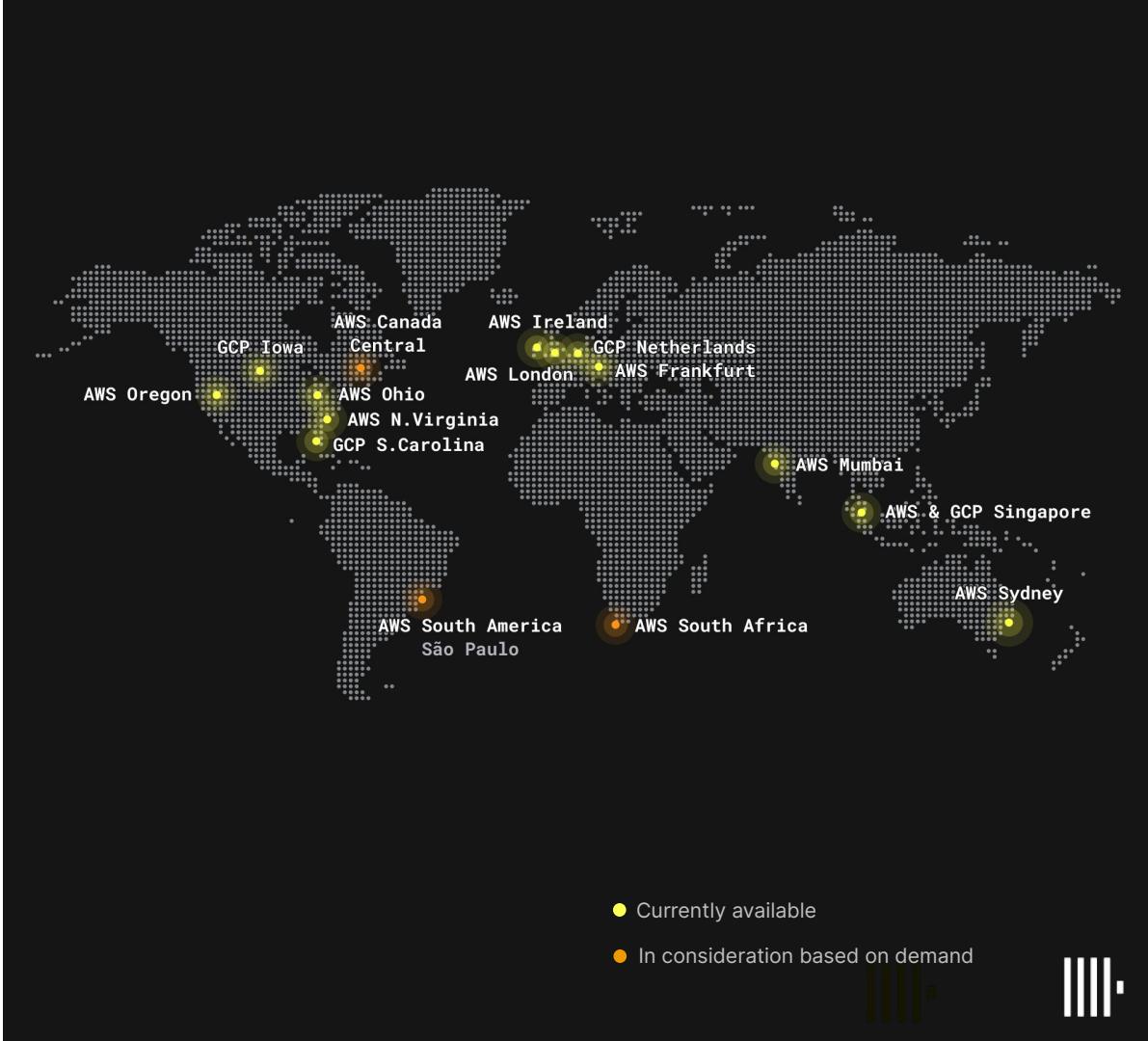
Supported Cloud Providers and Regions



- Available direct and via Marketplace
- Supported in 9 regions across United States, United Kingdom, Europe, Asia and Australia



- Available direct and via Marketplace
- Supported in 4 regions across United States, Europe and Asia



Cloud Provider Expansion

Microsoft Azure

Work in progress

Private Preview CYQ1 2024

Tracking interest in the waitlist



BYOC - AWS

Planning in progress

Private Preview CYQ2 2024

Recruiting design partners

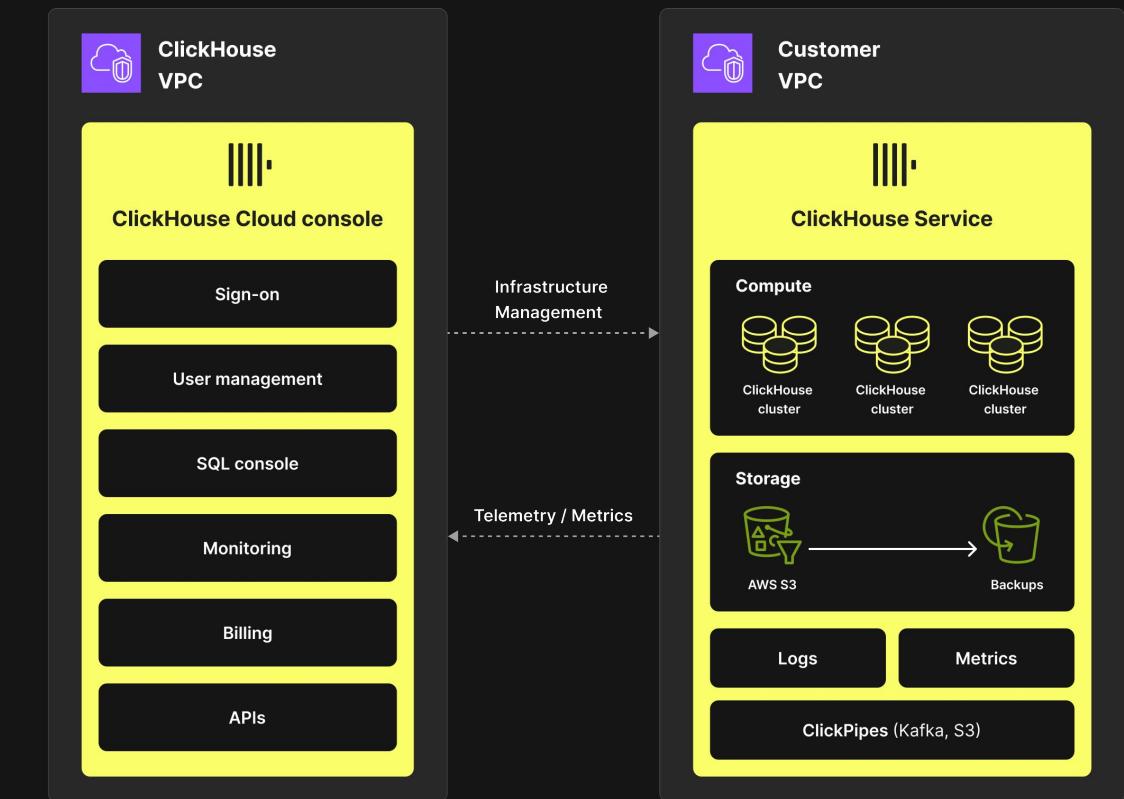


Note: All timelines are tentative and subject to change.

BYOC

Architecture(AWS)

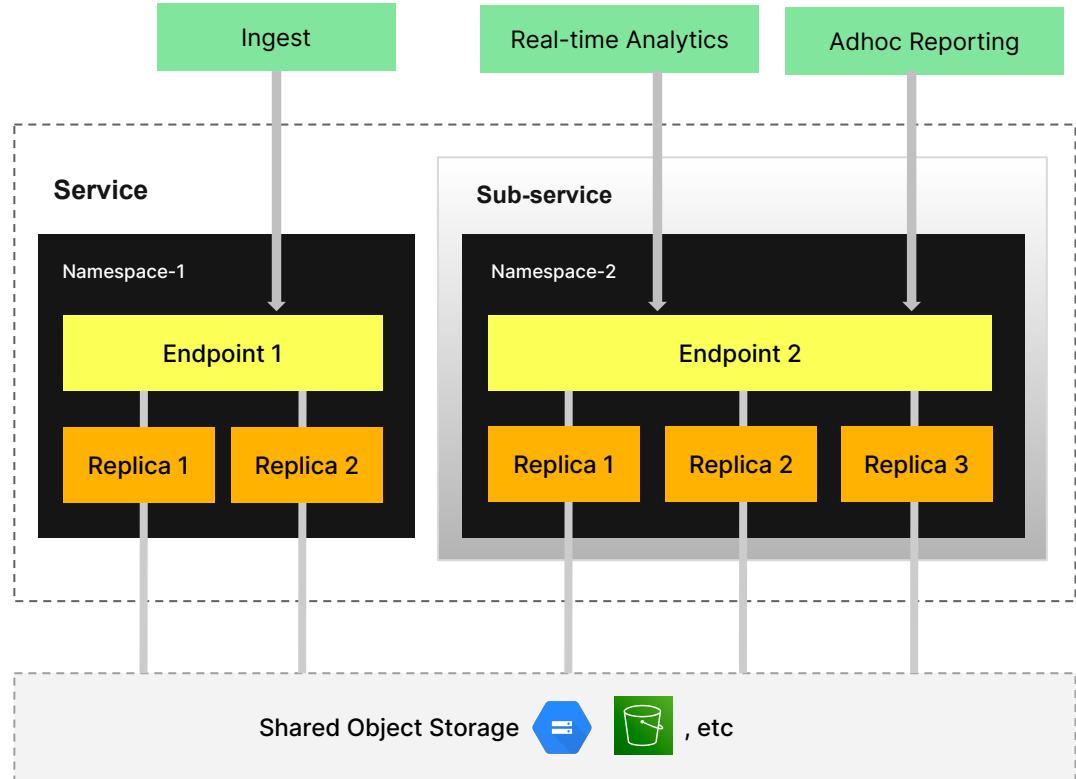
- Compute and Storage will reside in Customer' VPC
- Only service health metrics will be accessible from the ClickHouse VPC for Monitoring and Management
- All web assets will be hosted on ClickHouse VPC - includes User Management, Service Configuration, SQL Console,..
- Secure by design



Compute-Compute Separation

With Compute-Compute separation, services can allocate dedicated compute for specific operations - eg: Streaming Ingest vs Adhoc Reporting ,while sharing the same storage

- Compute units can be scaled independently
- Eliminates bottlenecks due to resource contention



Cloud Console Update

Zach, Cristina

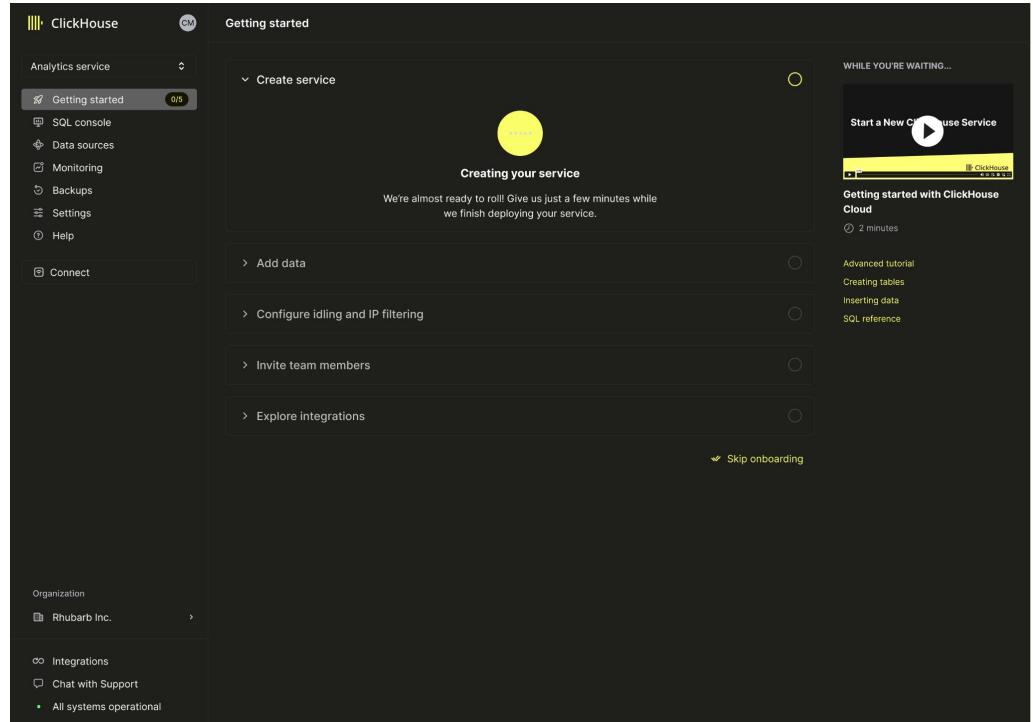


Unified Console Demo



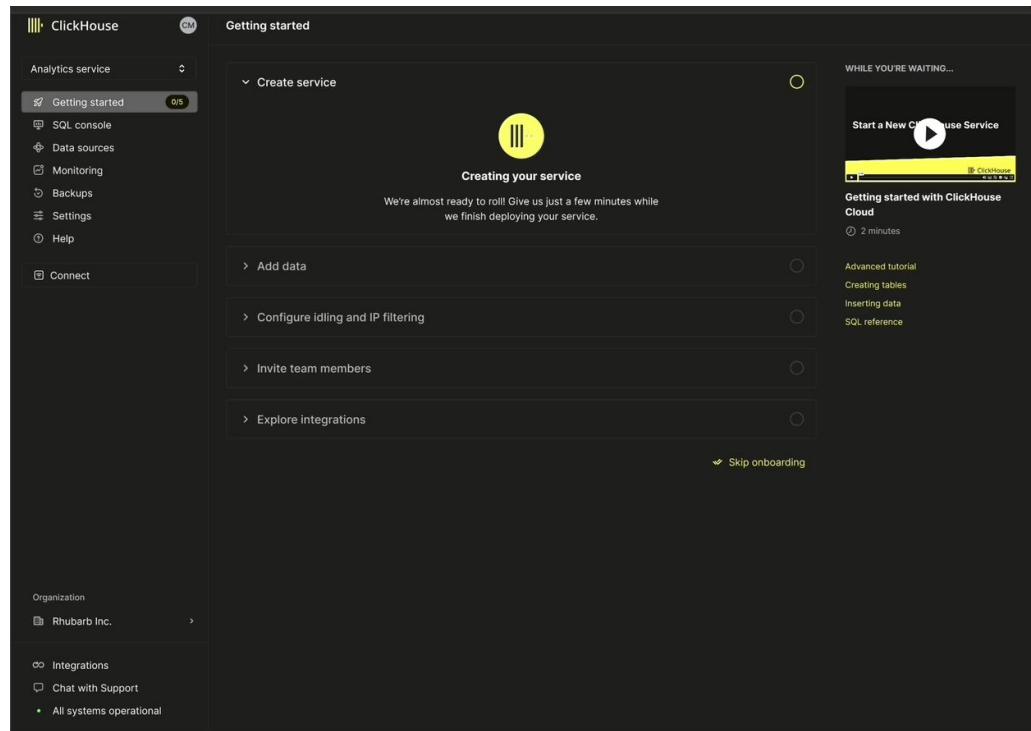
New onboarding flow

- Guided steps to help users be successful using ClickHouse Cloud
- Dedicated entry in the sidebar
- 5 steps, listed in our preferred order, but can be executed in any order
- Complementary info (videos/links) in a dedicated section on the right
- Can be skipped



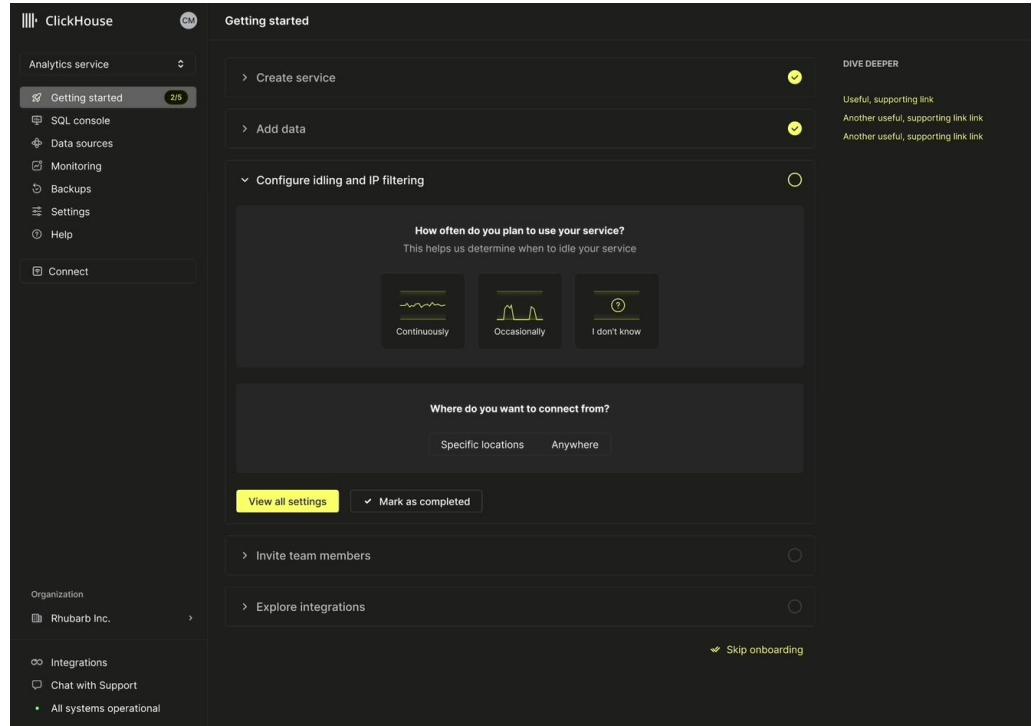
Adding data

- Only with data can one experiment the true potential of ClickHouse
- 4 main ways to add data:
 - ClickPipes (Kafka, S3, GCS, etc)
 - File Upload
 - Connect with your app/language client
 - Start with an empty table
- 2 secondary options:
 - Link to the “Data sources” page
 - View integrations



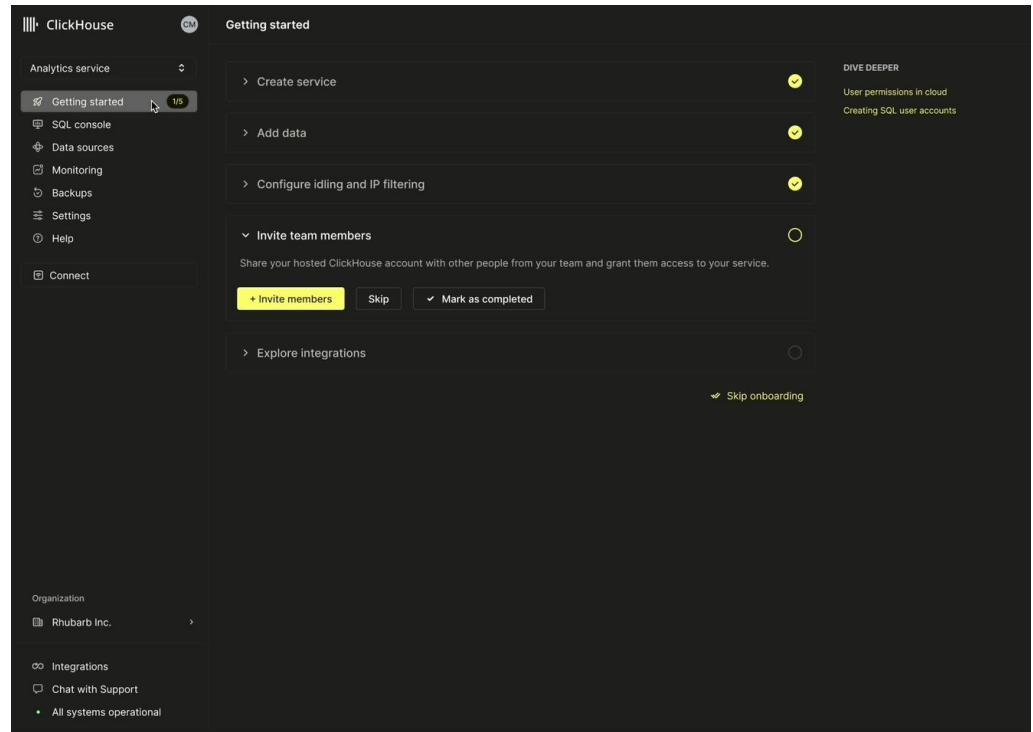
Shed light on important settings

- Configure the right idling period based on usage patterns
- Configure IP filtering for an extra layer of security
- Link to the Settings page



Invite team members

- Invite team members as Admins or Devs to your organization
- Can be skipped



Explore integrations

- Help users discover our vast ecosystem of integrations
- 4 groups:
 - Databases and data warehouses
 - Data visualization tools
 - Streams, logs and analytics
 - Language clients and drivers

The screenshot shows the ClickHouse 'Getting started' onboarding page. On the left, a sidebar menu includes 'Analytics service' (selected), 'Getting started' (marked as step 1/5), 'SQL console', 'Data sources', 'Monitoring', 'Backups', 'Settings', 'Help', and 'Connect'. Below the sidebar, the main content area has a title 'Getting started' and a list of tasks: 'Create service', 'Add data', 'Configure idling and IP filtering', and 'Invite team members', each with a yellow checkmark. A 'DIVE DEEPER' section lists 'ClickPipes', 'Other useful link', and 'Other useful link'. Under 'Explore integrations', there's a note: 'Use our table builder to create your first table, or type in your query manually in the SQL console'. Below this are four sections: 'Databases and data warehouses' (with icons for MySQL, PostgreSQL, Oracle, and others), 'Data visualization tools' (with icons for Grafana, Kibana, and others), 'Streams, logs and analytics' (with icons for Kafka, Logstash, and others), and 'Language clients and drivers' (with icons for Python, Java, Go, and others). At the bottom are buttons for 'View all integrations', 'Mark as completed', and 'Skip onboarding'.



Query sharing and access control

- Share saved queries with roles or individual users on your team
- Configure access levels for each role or user (full access, read-only, etc.)

The screenshot shows the ClickHouse Analytics service interface. On the left, there's a sidebar with 'Analytics service' navigation (SQL console, Data sources, Monitoring, Backups, Settings, Help) and a 'Connect' section. Below it, 'Organization' is set to 'Rhubarb Inc.' with 'Integrations', 'Chat with Support', and 'All systems operational' listed. The main area has tabs for 'Queries' (selected) and 'Tables'. Under 'Queries', there's a 'New query' button and a search bar ('Search queries...'). A 'My queries' section lists 14 recent queries, including 'Average cost', 'Pickups by hour of the day', and 'Average tip amount'. Each query card shows the SQL code, a 'Share' button, and a 'Delete query' button. A modal window titled 'Share query' is open, showing the 'With users' tab selected. It lists 'Anyone from Rhubarb Inc. with the link has' and 'Read only access'. Below this are sections for 'Add a team member' (with 'Daniel Xavier (Developer)' and 'zach@clickhouse.com (You)' listed), 'Query owner' (with 'Gareth Jones' and 'cristina@clickhouse.com (Developer)' listed), and two more 'Read only access' dropdowns. The background shows other query cards like 'Daily pickups per neighbourhood' and 'Another query'.

Query API endpoints

- Speed up development by building queries in the SQL console and exposing them as API endpoints
- Control access using existing ClickHouse Cloud OpenAPI keys
- Monitor endpoint request volume

The screenshot shows the ClickHouse Cloud interface. On the left, there's a sidebar with 'Analytics service' dropdown, 'SQL console', 'Data sources', 'Monitoring', 'Backups', 'Settings', 'Help', and a 'Connect' button. The main area has tabs for 'Queries' (selected) and 'Tables'. A search bar at the top says 'Most busy origin airports'. Below it is a code editor with the following SQL query:

```
1 SELECT origin, count(), round(avg(geoDistance(longitude_1, latitude_1, longitude_2, latitude_2))) AS distance,
2     bar(distance, 0, 10000000, 100)
3 FROM opensky
4 WHERE origin != ''
5 GROUP BY origin
```

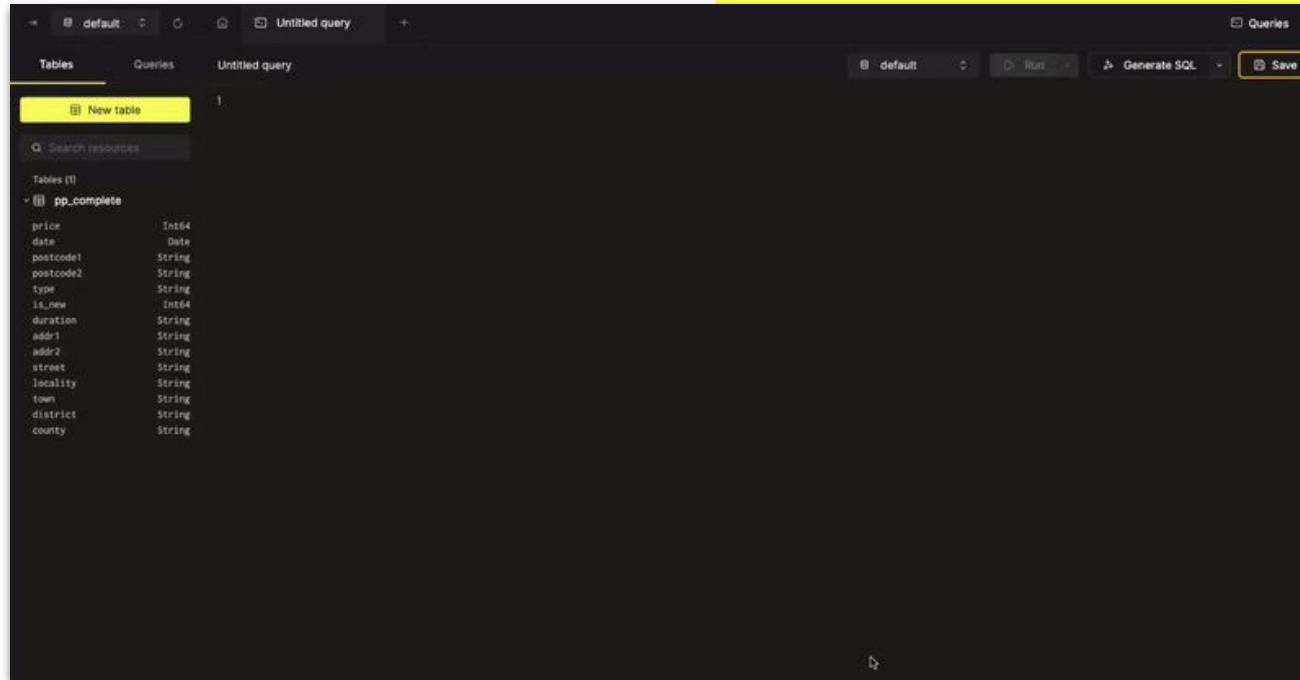
A modal window titled 'Share query' is open, showing two steps: 'Select an API key' (with 'Gareth's API Key' selected) and 'API Endpoint' (with 'Result format: CSVWithNames'). It also contains curl commands and a note about monitoring.

On the right, there's a monitoring section with a chart for 'Last hour' showing request volumes for 'Gareth's API Key' (blue line) and 'HOUSEClick Application' (pink line). Metrics shown include 'Total requests: 1,412,743' and 'Last request: 4 minutes ago'.



Copilot-style SQL suggestions

Intelligent in-line
ClickHouse SQL
suggestions as you write
your queries



The screenshot shows a ClickHouse interface with a dark theme. At the top, there are tabs for 'Tables', 'Queries', and 'Untitled query'. Below the tabs, a search bar says 'Search resources' and a button says 'New table'. Underneath, there's a section titled 'Tables (1)' with one item: 'pp_complete'. A detailed table structure is shown for this table:

price	Int64
date	Date
postcode1	String
postcode2	String
type	String
is_new	Int64
duration	String
addr1	String
addr2	String
street	String
locality	String
town	String
district	String
county	String



Security & Reliability Update

Leticia



Added authentication options

Strong authentication

- NIST 800-63 compliant passwords
- Multi-factor authentication options
- Social login with Google

More to come

- Social login with Microsoft
- Social login with GitHub
- SAML broad release
- SCIM enabled SSO

Enterprise Single Sign-On (SSO)

- SAML enabled (limited release)



Microsoft



More control for backups

- On-demand backups
- Custom configurations for frequency and retention
- Custom scheduling
- Save to your own S3 bucket

Backup

Frequency

Choose daily, weekly, or create a custom schedule that suits your data protection strategy. Your control, your pace.

Backup every

Week

Retention period

Define how long your backup history should be retained. Strike the right balance between data accessibility and efficient storage management.

Keep the backups for a

Week

Increasing the retention period of the backups might impact your monthly costs. Check the [billings page](#) for more information.

Scheduling

Personalize the exact time of day for your backups. Minimize workflow disruption by selecting a time that aligns seamlessly with your operational needs and preferences.

Backup should start at (local time)

02:00 AM

UTC -09:00



Bring your data, all of it

HIPAA

Protected Health Information

- Secure connections with IP filters
- SHA256 hashed passwords
- Customer managed encryption
- Detailed logging

Targeting H1 2024

Federal

U.S. Federal Government

- Self-managed install
- Separation of compute + storage
- FIPS 140-3 encryption
- APIs to enable automation
- FedRAMP in cloud proposed

Planning & Scoping

PCI

Credit Card Information

- Designed for PCI 4.0
- Improved user management
- Outbound traffic control
- Granular encryption options

Planning & Scoping

Note: All timelines are tentative and subject to change.



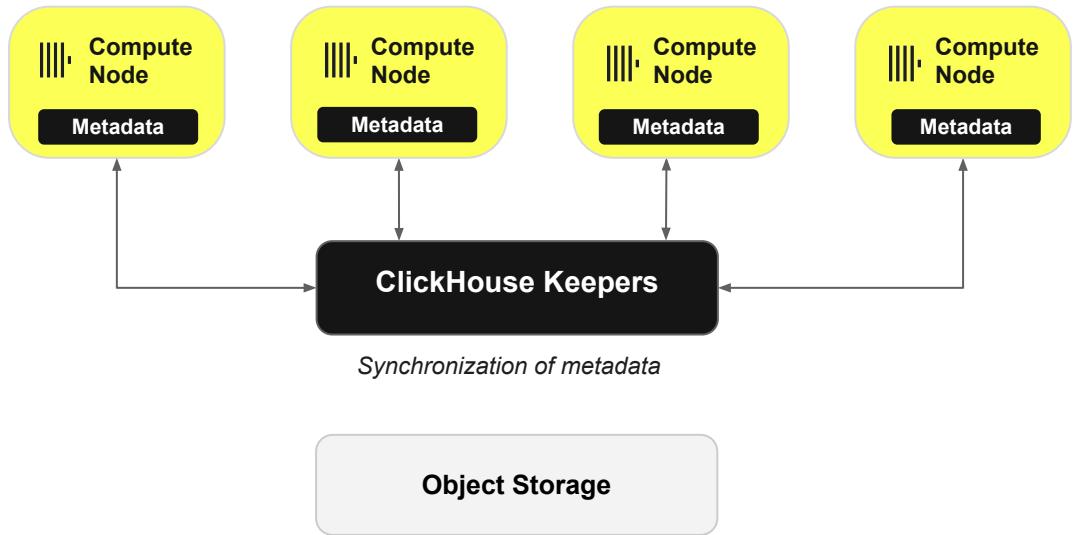
Core Update

Melvyn, Alexey



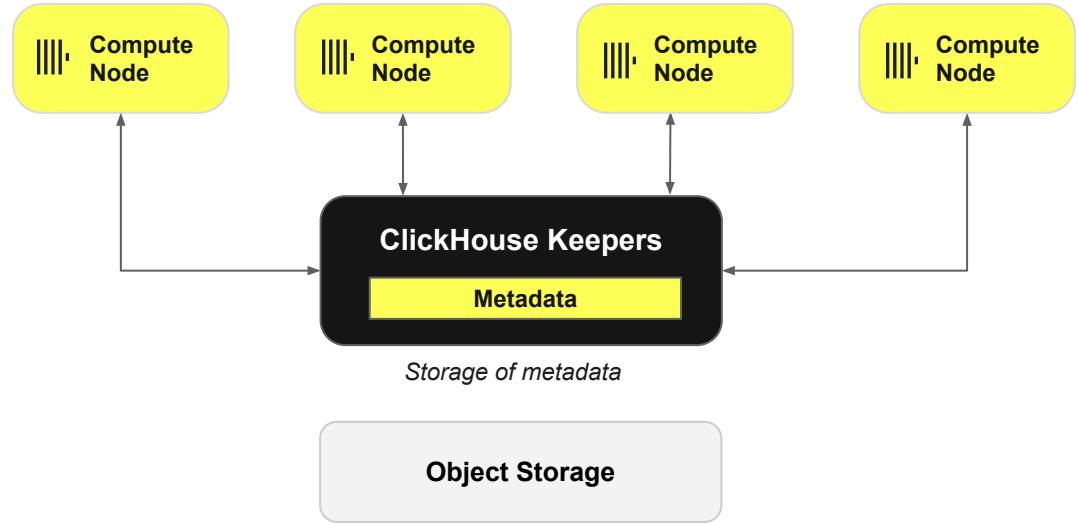
Before SharedMergeTree

- Metadata needs to be replicated to every clickhouse-server
- Adding or removing a node is a slow operation
- The higher the number of servers is, the more replication of metadata is needed, it impacts:
 - ✓ insert throughput
 - ✓ background operations



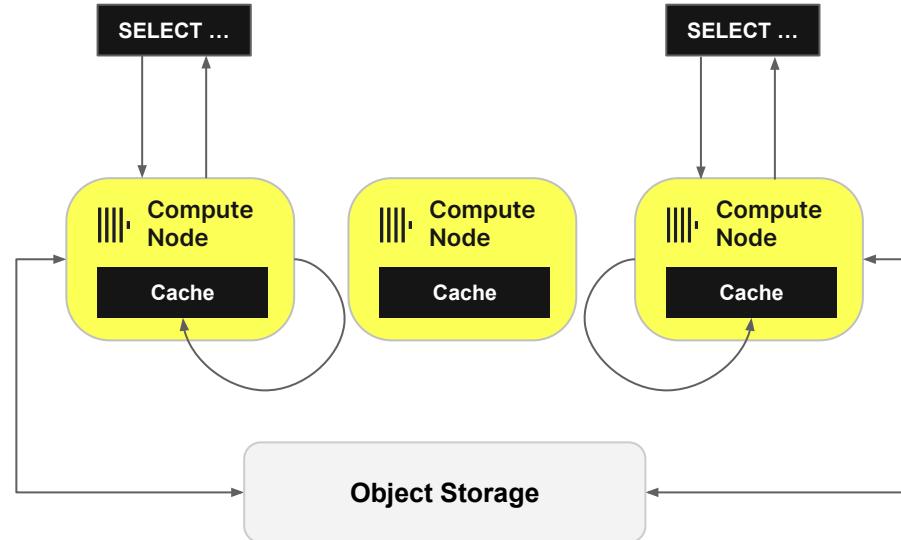
Introducing SharedMergeTree

- SharedMergeTree a cloud native replacement of ReplicatedMergeTree
- Faster scale-up and down operations
- Improved throughput for:
 - ✓ Background Merges
 - ✓ Inserts
 - ✓ Mutations



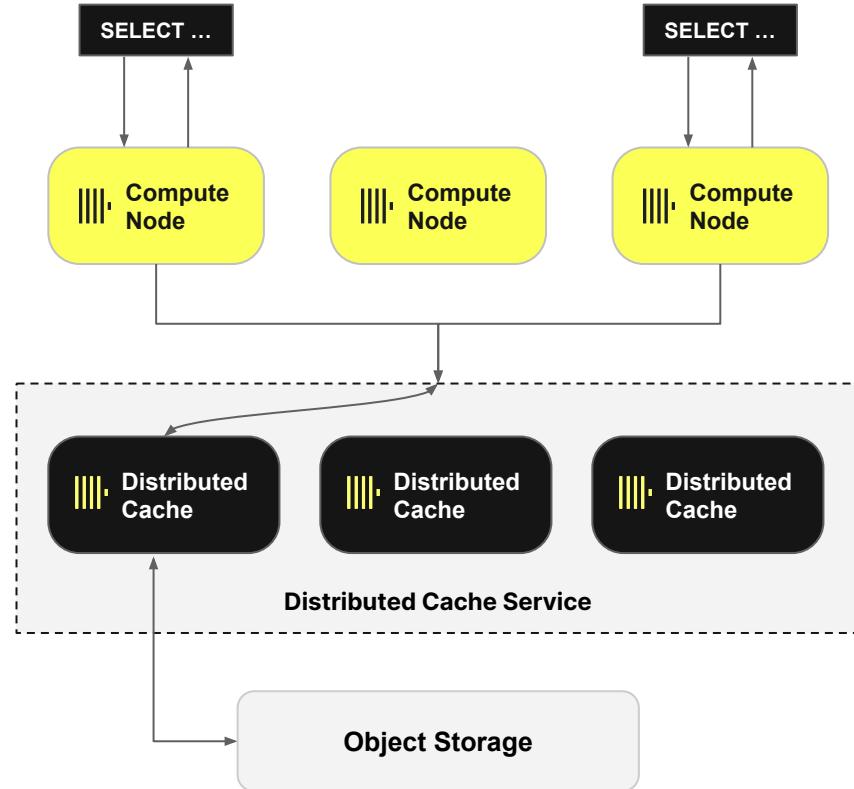
The Current State of our Cache

- Each node has their own cache
- The content of the cache varies depending on the node
- Cache needs to be repopulated after each restart, scale up/down, etc...



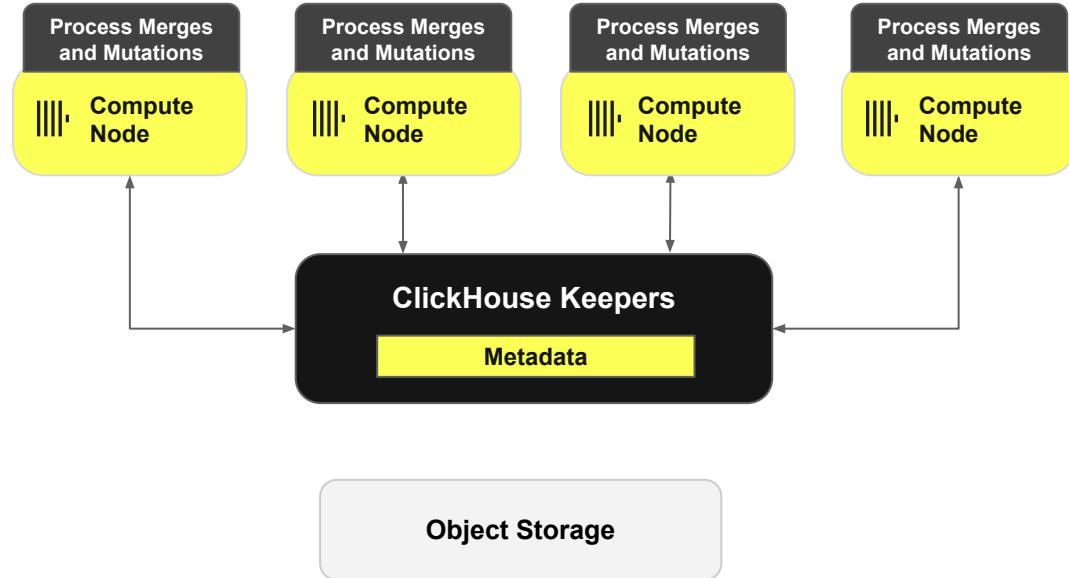
Distributed Cache

With a new distributed cache service, the cache is now separated from the compute node. It means that it does not need to be repopulated after scale up/down, restart.



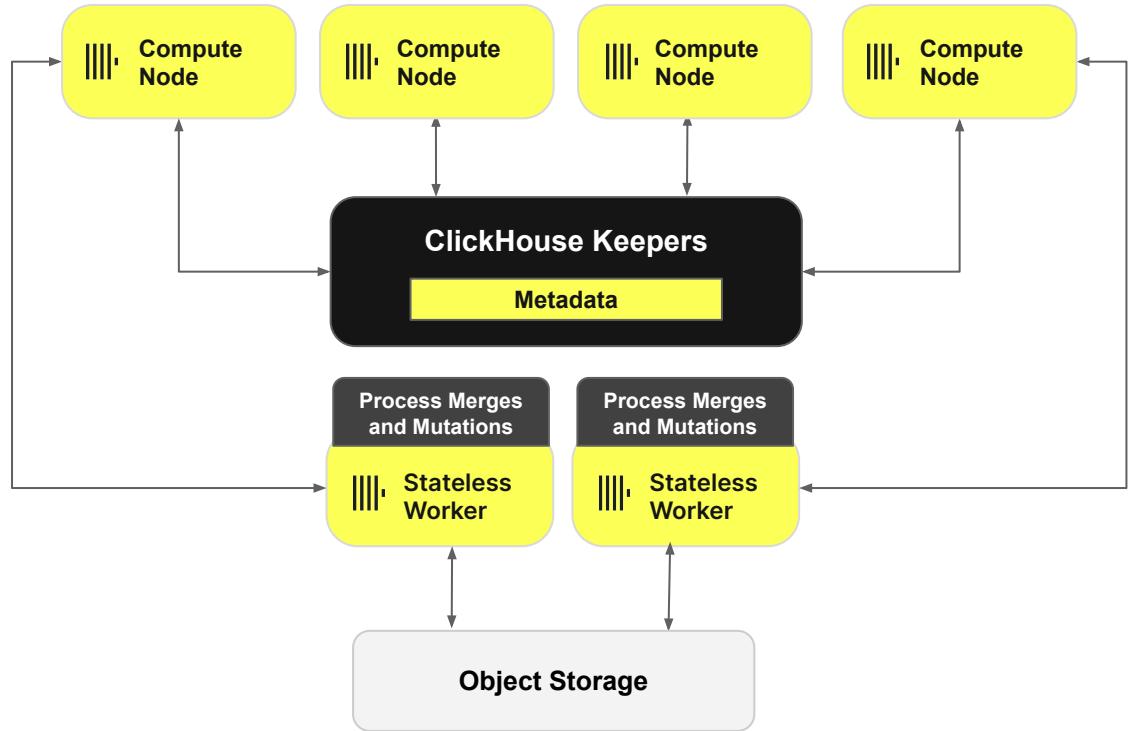
The Current State of Background Operations

- Background operations can add surprising load to a server executing end-user queries
- Resource utilization is not directly controlled by the end-user



Stateless Workers (R&D)

- Better separation of resource usage
- Compute nodes are only used for user workload
- Stateless worker are only used for background operations



Advancements In Performance

How *fast* is ClickHouse Cloud?



Insert Performance

How quickly can you insert data?

One year ago, we created a task, "Insert Scalability Challenge" to test the new **SharedMergeTree** engine.

The goal was to insert **400 billion records as quickly as possible** and find all possible ways how it could break.



Insert Scalability Challenge

```
CREATE TABLE wikistat
(
    time DateTime CODEC(Delta, ZSTD(3)),
    project LowCardinality(String),
    subproject LowCardinality(String),
    path String CODEC(ZSTD(3)),
    hits UInt64 CODEC(ZSTD(3))
)
ORDER BY (path, time);

INSERT INTO wikistat WITH
    parseDateTimeBestEffort(extract(_file, '^pageviews-(\d-+)\.gz$')) AS time,
    splitByChar(' ', line) AS values,
    splitByChar('.', values[1]) AS projects
SELECT
    time,
    projects[1] AS project,
    projects[2] AS subproject,
    decodeURIComponent(values[2]) AS path,
    CAST(values[3], 'UInt64') AS hits
FROM s3Cluster(default, 's3://clickhouse-public-datasets/wikistat/original/pageviews*.gz', LineAsString)
WHERE length(values) >= 3;
SETTINGS parallel_distributed_insert = 1;
```



Insert Scalability Challenge: Requirements

1. The cluster successfully scales to 100 replicas spread across different physical hosts.
2. It does not break during the insertion. Insert query should finish successfully.
3. Health checks work during the insertion, and connections do not block.
4. The speed remains mostly uniform during the insertion.
5. All replicas participate in parts merging during the insertion.
6. The number of active data parts after the insertion is less than 5000.
7. The service does not take long to restart if you do it just after the insertion.
8. Real time queries finish in less than a second during the whole insertion process.
9. Backups work on this service.
10. The progress bar and metrics work correctly. :)



Insert Scalability Challenge

April 2023

- 👍 **400 bn rows inserted in 3230 sec** on 50 replicas
- 👍 **125 million rows/sec** insertion speed
- 👎 didn't use all the resources for merges
- 👎 has **too many parts** after then insertion
- 👎 the startup time is **over 6 minutes**
- 👎 SELECTs are slow during the insertion
- 👎 there was **no progress bar** for a query



Insert Scalability Challenge

July 2023

We made it harder:

- fault injection in the network between clickhouse-server and clickhouse-keeper;

And it didn't pass :)



Insert Scalability Challenge

January 2024

- 💪 440 bn rows inserted in **890s on 100 replicas**
- 💪 **498 million rows/sec** insertion speed
- 💪 all replicas participate in merges
- 💪 keeper is chilling during the load
- 💪 the startup time is **2 seconds**
- 💪 realtime SELECTs are real
- 💪 the progress bar works :)



Insert Scalability Challenge

It is a pretty nice test, but customers' workloads are even better.

A case study from a customer:

- migrated **1.5 PiB** of (compressed) data **in two weeks**;
- used **20 replicas** during the insertion;
- after the initial import, scaled down to 10 replicas;
- **10 replicas serve all** the real-time load, keeping the costs low;



Select Performance

How quickly do you get the answer for a SELECT query?

- Can we get a sub-second performance on tens of billions of records scanned?



How well does ClickHouse Cloud scale?

- Does it scale to hundreds of replicas?



Does it maintain low latency for massively distributed queries?

- Does it feel snappy with a **separation of storage and compute** with object storage + multi-level cache on SSD and memory?



**The best way to answer these questions
is to create a demo.**



Alexey DEMO

||||· ClickHouse

Select Performance

Speed up for parallel replicas:

Better cache locality

- it reads the same ranges from the same replicas if they are available.

Better tail latency

- faster replicas will steal tasks from slow ones.

Example from a cluster with 100 pods, 30 CPU cores each:

Processed 38.23 billion rows, 570.69 GB (**24.51 billion rows/s.**, **365.90 GB/s.**)



Performance of Background Operations

ClickHouse can insert data so fast, but what does it have to do to accommodate this data?

Insertions have to be coordinated in Keeper

- so now we optimized Keeper for memory usage!

Inserted data parts have to be merged in time

- a cluster can handle over 1000 merges in parallel;
- background merges are optimized for memory usage.

Bonus: adaptive mode for asynchronous inserts (coming soon).



ClickHouse Community Call (Monthly)

Highlights from the last few months 

New features

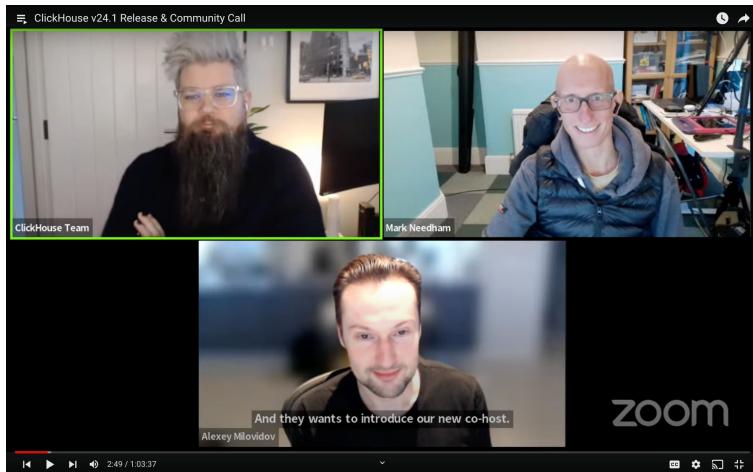
- Type inference for JSON
- S3Queue Table Engine
- Asynchronous loading of tables
- PASTE JOIN
- Temporary user credentials

Performance improvements

- Optimization for SELECT FINAL
- Optimization for external aggregations
- Improvement on the pipeline for short queries
- Condition pushdown for ORC and Parquet
- Faster window functions
- Optimization for reading from cache
- Adaptive timeout for S3

SQL compatibility improvements

- New functions: TO_DAYS, addDate, subDate
- toDayOfWeek, toYearWeek, toWeek now support String argument
- Aliases (in order to avoid breaking migrations): STD
- Support for BI tools: Information schema & prepared statements



Join us on the last Thursday each month for:

<https://www.youtube.com/c/ClickHouseDB>

- Coverage of new features in monthly core database releases
- Cool demos, discussion about the power of SQL, and database trivia
- Sneak peak into roadmap and AMA with Alexey, creator of ClickHouse



Integrations Update

Ryadh





ClickPipes Expansion

- Added support for additional Kafka flavours, 7 in total!

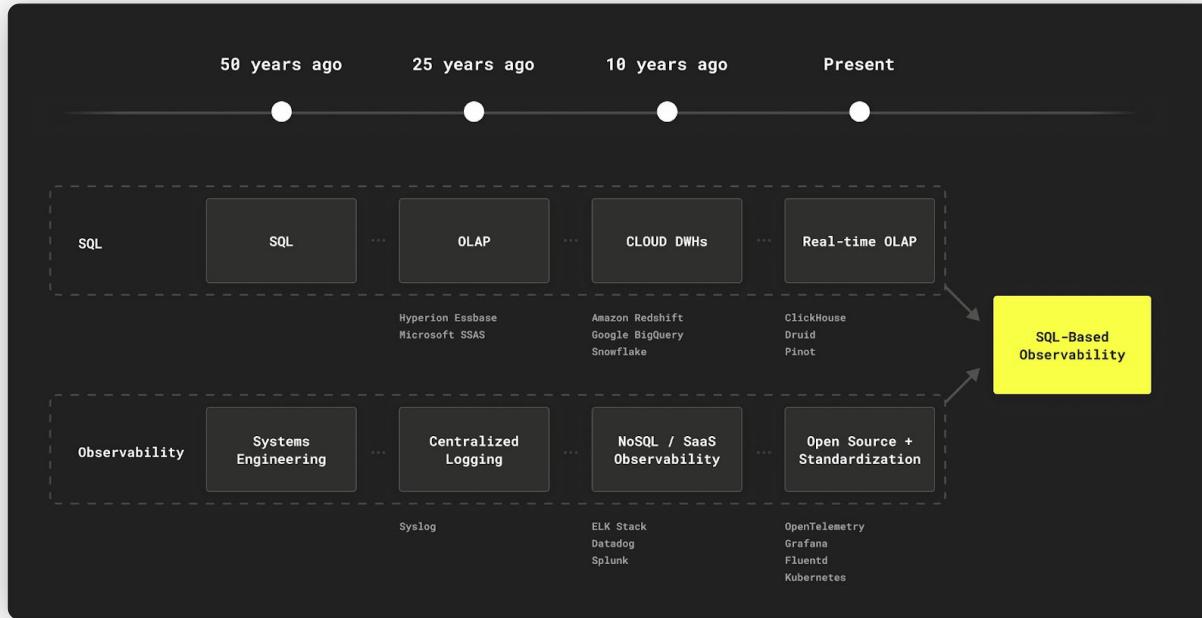
Coming soon

- Amazon S3 connector for bulk data loading
- Avro format support for Kafka
- Amazon Kinesis connector

The screenshot shows the ClickHouse ClickPipes interface. On the left, there's a sidebar with navigation links: Ryadhi Integrations ..., SQL Console, Data sources (which is currently selected), Backups, Settings, Monitoring, Help, and a 'Connect' button. The main area is titled 'Data sources / ClickPipes'. It features a heading 'Select the data source' with a sub-instruction: 'If you need any help to setup ClickPipes, you can, [access the documentation](#) for more details.' Below this, there's a grid of eight icons representing different data sources: Confluent Cloud (blue circle icon), Apache Kafka (black cluster icon), AWS MSK (purple triangle icon), Redpanda (red 'R' icon), Upstash (green swirl icon), WarpStream (red waveform icon), Azure Event Hub (blue square icon), and AWS S3 (green bucket icon). To the right of the grid, a vertical list of steps is shown: 1. Select the data source, 2. Setup your ClickPipe Connection, 3. Incoming Data, 4. Parse Information, and 5. Details and Settings.



SQL-Based Observability



<https://clickhouse.com/blog/the-state-of-sql-based-observability>

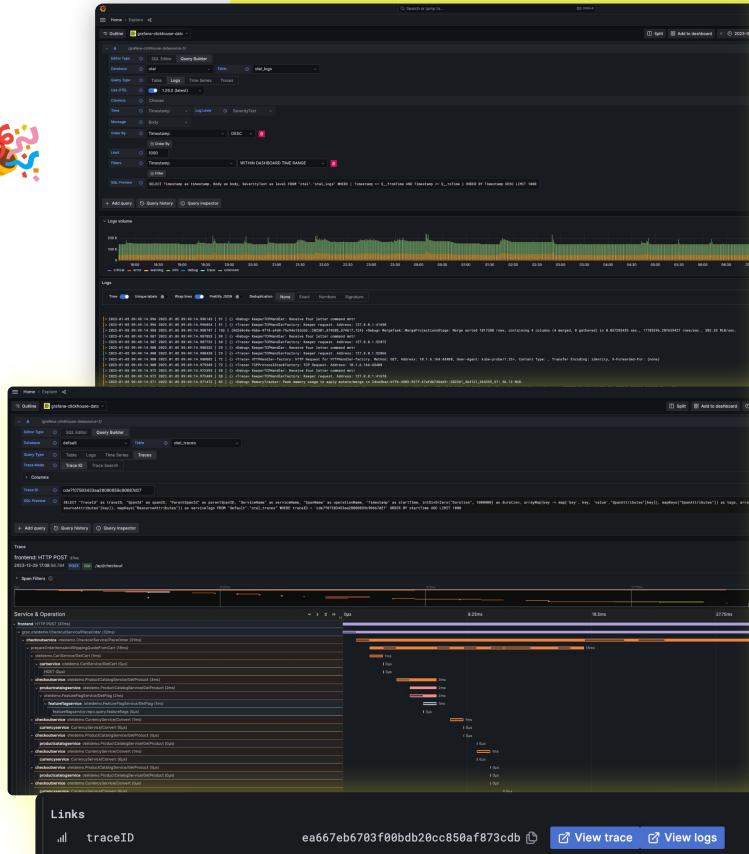




Grafana datasource V4



- Improved UI/UX for query builder, datasource configuration, logs and traces explorer
- Improved support for the OpenTelemetry standard
- Aligned with the SQL-Observability workflow with a guided experience



|||· ClickHouse

Grafana Plugin DEMO





Also happening in integrations ...

- Extended supported Data Types for Apache Beam
- Kafka Connector Sink performance improvements
- MySQL Interface Improvements
- DBT upgrade to 1.7 with support for Materialized Views
- clickhouse-java : refactoring in progress
- Golang/Python/JS: reliability and performance improvements

The image shows two screenshots of GitHub pages for ClickHouse integrations.

The top screenshot is for the "ClickHouse Java Libraries". It displays the README, license (Apache-2.0), and security information. It highlights the latest release (v0.6.0), downloads (269), coverage (51.4%), and a nightly build (v0.6.0 SNAPSHOT). A "Connect to Ryadh's Demo" button is present. The page lists various Java clients: clickhouse-data (format and streaming), clickhouse-parser (<SQL parser>), clickhouse-client (<Generic clients>), clickhouse-client-native (native C/C++ wrapper), clickhouse-http-client (HTTP client), clickhouse-grpc-client (gRPC client), and core-clickhouse-client (core ClickHouse client). A summary sidebar on the right shows the service is named "Ryadh's Demo", status is "Running", plan is "Serverless", and provider is "aws".

The bottom screenshot is for the "dbt-clickhouse" plugin. It shows the README, license (Apache-2.0), and a logo featuring the ClickHouse four-bar icon and the dbt logo. It describes the plugin as porting dbt functionality to ClickHouse. Installation instructions via pip are provided: "pip install dbt-clickhouse".



Ecosystem Integrations status and roadmap

	Available	Recent/Next
Data Ingestion / Orchestration	<ul style="list-style-type: none">🔥 DBT Adapter (core)🔥 MySQL Interface (core)🔥 Fivetran (partner)♦ OpenTelemetry (community)✓ Airbyte (partner)✓ Vector (community)✓ PostgreSQL (core)✓ MySQL (core)✓ FluentBit (community)✓ EMQX (partner)✓ Apache Nifi (community)✓ Apache Spark (community)✓ Apache Flink (community)	<ul style="list-style-type: none">🟡 ClickPipes - S3 / GCS🟡 ClickPipes - Avro for Kafka🟡 ClickPipes - Kinesis♦ ClickPipes - CDC Postgres♦ Apache Beam (community)♦ Google Dataflow (partner)✓ ClickPipes - Azure Event Hub✓ ClickHouse-Kafka-Connect GA✓ Apache Airflow (community)✓ RisingWave (community)✓ Apache Sea Tunnel (community)
Language Clients	<ul style="list-style-type: none">✓ Node.js (Added browser support) (core)✓ Common Lisp (go Julio!)✓ Golang (core)✓ Python (core)✓ C++✓ PHP (community)✓ Ruby (community)✓ Rust (community)✓ R (community)✓ Erlang (community)	<ul style="list-style-type: none">🔥 Java (Refactoring) (core)♦ C# .NET (community)♦ ODBC (Maintenance) (core)

Legend

- ✓ Completed
- 🟡 In development
- ♦ Coming Next / Evaluating
- 🔥 Recently improved



Ecosystem Integrations status and roadmap

	Available	Recent/Next
Data Visualization	<ul style="list-style-type: none">🔥 Tableau Desktop (core)🔥 Google Looker Studio (partner)✓ Superset / Preset (partner)✓ Deepnote (partner)✓ HEX (partner)✓ Mode (partner)✓ Metabase (core)✓ Cube.js (partner)✓ EXPLO (partner)✓ Mode.com (partner)✓ Redash (partner)✓ Sisense (partner)✓ Tooljet (partner)✓ Zing Data (partner)✓ Observable (partner)✓ OpenBlocks (community)	<ul style="list-style-type: none">🔥 Grafana (Redesign v4) (partner)🟡 Tableau Online (partner)🟡 Google Looker (partner)🔥 AWS QuickSight (partner)🔥 PowerBI (core)♦ Azure Power BI (partner)
SQL Clients	<ul style="list-style-type: none">✓ ClickHouse Client✓ DBeaver (partner)✓ Datagrip (partner)✓ JupyterSQL (community)	

Legend

- ✓ Completed
- 🟡 In development
- ♦ Coming Next / Evaluating
- 🔥 Recently improved

See 110+ integrations on
<https://clickhouse.com/docs/en/integrations>



Thanks for joining!



clickhousedb.slack.com



@ClickHouseDB

