



ClickHouse +



Sydney Meetup
September 5, 2024



ROKT

Thank You



DataEngBytes 2024

Run by data engineers, for data engineers



City: Sydney



Tue 24 September 2024



**Sofitel Sydney Wentworth,
2000**



9:00 AM - 5:30 PM AEST



Hosts & Speakers



Paul Davis

AVP Sales, APAC
ClickHouse



Peter Hanssens

Founder & Principal Architect
Cloud Shuttle



Jarryd Timm

Account Executive
ClickHouse



Johnny Mirza

Solution Architect
ClickHouse



Sharat Madanapalli

Head of Data & AI
Canopus



Dan Draper

Founder & CEO
CipherStash



Tech Talks



Our Journey to Ingest and Analyse 5 Billion Network Events / Day

Sharat Madanapalli, Head of Data & AI @ Canopus



Searchable encryption in-use for ClickHouse: what, how and why

Dan Draper, Founder & CEO @ CipherStash



What's new with ClickHouse: Updates, Integrations & Acquisitions

Johnny Mirza, Solutions Architect @ ClickHouse

Our Journey to Ingest and Analyse 5 Billion Network Events / Day



Sharat Madanapalli
Head of Data & AI @ Canopus



What's new with ClickHouse: Updates, Integrations & Acquisitions

September 5 , 2024



What is ClickHouse?

Your (soon-to-be) favorite database!

Open source **column-oriented** **distributed** **OLAP** **database**

Since 2009
31,000+ GitHub stars
1300+ contributors
500+ releases

Best for aggregations
Files per column
Sorting and indexing
Background merges

Replication
Sharding
Multi-master
Cross-region

Analytics use cases
Aggregations
Visualization
Mostly immutable data



ClickHouse Release 24.8 LTS



<https://www.youtube.com/watch?v=AeLmp2jc51k>

https://presentations.clickhouse.com/release_24.8/index.html#cover

What's new in ClickHouse

✓ 19 new features ✓

- ★ Multiquery Mode By Default
- ★ Virtual Column_etag
- ★ Hive-Style Partitioning
- ★ printf function
- ★ Tagged Query Cache
- ★ Control Of Projections During Merges

📈 8 performance optimisations 📈

- ★ Optimisations for Subcolumns
- ★ OPTIMIZE Query For Join Tables
- ★ Faster Merges
- ★ Faster Drop Database

🔧 65 bug fixes 🔧

★ Interesting new features ★

- ★ Analyzer In Production
- ★ New Kafka Engine
- ★ TimeSeries Engine
- ★ JSON Data Type

The New Kafka Engine

The Kafka engine exists in ClickHouse since 2017
— it implements streaming consumption and data pipelines from Kafka.

Its downside: non-atomic commit to Kafka and to ClickHouse, leading to the possibility of duplicates in the case of retries.

Now there is an option to manage the offsets in Keeper:

```
SET allow_experimental_kafka_offsets_storage_in_keeper = 1;  
  
CREATE TABLE ... ENGINE = Kafka(  
    'localhost:19092', 'topic', 'consumer', 'JSONEachRow')  
SETTINGS  
    kafka_keeper_path = '/clickhouse/{database}/kafka',  
    kafka_replica_name = 'r1';
```


The New Kafka Engine

```
CREATE TABLE ... ENGINE = Kafka(  
    'localhost:19092', 'topic', 'consumer', 'JSONEachRow')  
SETTINGS  
    kafka_keeper_path = '/clickhouse/{database}/kafka',  
    kafka_replica_name = 'r1';
```

With the new option it does not rely on Kafka to track the offsets, and does it by itself with ClickHouse Keeper.

If an insertion attempt fails, it will take exactly the same chunk of data and repeat the insertion, regardless of network or server failures.

This enables deduplication and makes the consumption **exactly-once**.

Developer: János Benjamin Antal.

TimeSeries Engine

Now ClickHouse supports **Prometheus protocols** for remote write and read.

The new, **TimeSeries** Engine implements storage for metrics.

```
SET allow_experimental_time_series_table = 1;

CREATE TABLE tbl ENGINE = TimeSeries; -- the default options.

CREATE TABLE tbl ENGINE = TimeSeries
    DATA ENGINE = MergeTree
    TAGS ENGINE = ReplacingMergeTree
    METRICS ENGINE = ReplacingMergeTree;
```

Developer: Vitaly Baranov.

TimeSeries Engine

```
$ cat /etc/clickhouse-server/config.d/prometheus.yaml
prometheus:
  port: 8053
  handlers:
    my_rule_1:
      url: '/write'
      handler:
        type: remote_write
        database: default
        table: tbl
    my_rule_2:
      url: '/read'
      handler:
        type: remote_read
        database: default
        table: tbl
    my_rule_3:
      url: '/metrics'
      handler:
        type: expose_metrics
```

TimeSeries Engine

ClickHouse is listening the Prometheus protocol and ready to receive metrics.

TimeSeries engine is simple to use, but allows many customizations:

- put some tags (e.g., hostname) into separate columns;
- adjust table's primary key;
- adjust column types;
- ...

But there is more work to do:

- support for PromQL;

Developer: Vitaly Baranov.

JSON Data Type

```
SET allow_experimental_json_type = 1;

CREATE TABLE test (time DateTime, data JSON) ORDER BY time;

-- or with parameters:
data JSON(
    max_dynamic_paths = N,
    max_dynamic_types = M,
    some.path TypeName,      -- a type hint
    SKIP path.to.skip,       -- ignore some paths
    SKIP REGEXP 'paths_regexp')
```

Developer: Pavel Kruglov.

JSON Data Type

How it works:

- Analyzes the JSON and infers data types for every path.
- Stores every path and every distinct type as a subcolumn.
- Up to the maximum number, when it will fallback to storing the rest of the paths together.

It enables fast column-oriented storage and queries on arbitrary semistructured data!

Developer: Pavel Kruglov.

JSON Data Type

How to insert:

- insert with the JSONEachRow format;
- insert a string containing JSON to the column of JSON type;
- insert with the JSONAsObject format
to put the whole object into the JSON column;
- cast from String to JSON;

```
CREATE TABLE test (data JSON) ENGINE = Memory;  
INSERT INTO test VALUES ('{"a" : {"b" : 42}, "c" : [1, 2, 3]}');  
INSERT INTO test FORMAT JSONEachRow  
{"data": {"a" : {"b" : 42}, "c" : [1, 2, 3]}};  
SELECT data FROM test;
```

Developer: Pavel Kruglov.

JSON Data Type

How to select:

— read a certain path as a **Dynamic** column:

```
SELECT data.a AS x, toTypeName(x) FROM test;
```

— read a certain path and cast to the desired data type:

```
SELECT data.a.b::UInt32 AS x, toTypeName(x) FROM test;
```

— read a certain path and assume its data type:

```
SELECT data.a.b::Int64 AS x, toTypeName(x) FROM test;
```

— read a subobject as JSON:

```
SELECT data.^a AS x, toTypeName(x) FROM test;
```

Other Updates

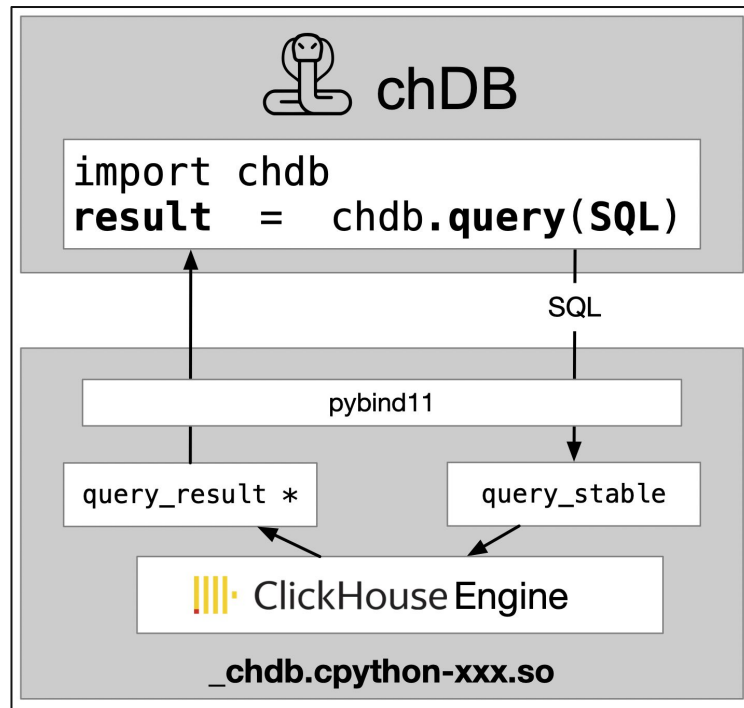


chDB Acquisition

What is chDB?

chDB is an in-process OLAP SQL engine.

The creator of chDB's initial goal was to make ClickHouse available as an “out-of-the-box” Python module. Now, it also supports other programming languages including C/C++, Golang, Rust, NodeJS, Bun, and .NET



System & Machine

Relative time (lower is better)

chDB (EPYC 9654, 128G, 4TB):		×1.38
DuckDB (EPYC 9654, 128G, 4TB):		×1.49
Polars (EPYC 9654, 128G, 4TB):		×3.72
Pandas (EPYC 9654, 128G, 4TB):		×16.12

Detailed Comparison

	chDB (EPYC 9654, 128G, 4TB)	DuckDB (EPYC 9654, 128G, 4TB)	Polars (EPYC 9654, 128G, 4TB)	Pandas (EPYC 9654, 128G, 4TB)
<input checked="" type="checkbox"/> Load time:	0	0	23s (×1.00)	0
<input checked="" type="checkbox"/> Data size:	0.86 GiB (×1.00)	0.86 GiB (×1.00)	0.86 GiB (×1.00)	0.86 GiB (×1.00)
<input checked="" type="checkbox"/> Q0.	0.065s (×7.50)	0.034s (×4.43)	0.000s (×1.00)	8.789s (×878.50)
<input checked="" type="checkbox"/> Q1.	0.027s (×1.13)	0.027s (×1.10)	0.023s (×1.00)	0.162s (×5.19)
<input checked="" type="checkbox"/> Q2.	0.024s (×1.92)	0.025s (×1.99)	0.038s (×2.70)	0.008s (×1.00)
<input checked="" type="checkbox"/> Q3.	0.027s (×2.58)	0.022s (×2.25)	0.004s (×1.00)	0.008s (×1.24)
<input checked="" type="checkbox"/> Q4.	0.183s (×2.19)	0.078s (×1.00)	0.133s (×1.63)	0.211s (×2.51)
<input checked="" type="checkbox"/> Q5.	0.149s (×1.70)	0.084s (×1.00)	0.304s (×3.36)	0.669s (×7.25)
<input checked="" type="checkbox"/> Q6.	0.026s (×2.22)	0.025s (×2.14)	0.006s (×1.00)	0.020s (×1.87)
<input checked="" type="checkbox"/> Q7.	0.054s (×1.70)	0.046s (×1.48)	0.028s (×1.00)	0.071s (×2.13)
<input checked="" type="checkbox"/> Q8.	0.083s (×1.00)	0.091s (×1.09)	0.261s (×2.91)	0.579s (×6.34)
<input checked="" type="checkbox"/> Q9.	0.092s (×1.00)	0.127s (×1.34)	0.248s (×2.52)	0.682s (×6.76)
<input checked="" type="checkbox"/> Q10.	0.094s (×1.66)	0.053s (×1.00)	0.129s (×2.21)	0.840s (×13.54)
<input checked="" type="checkbox"/> Q11.	0.058s (×1.00)	0.059s (×1.01)	0.121s (×1.91)	0.870s (×12.84)
<input checked="" type="checkbox"/> Q12.	0.123s (×1.22)	0.099s (×1.00)	0.200s (×1.92)	2.771s (×25.48)
<input checked="" type="checkbox"/> Q13.	0.114s (×1.00)	0.150s (×1.36)	22.401s (×180.77)	2.760s (×22.35)

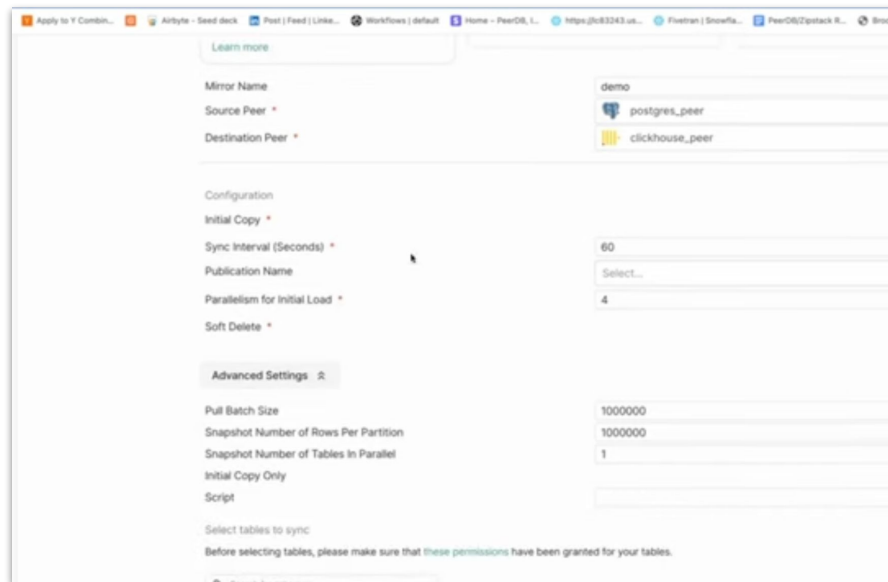
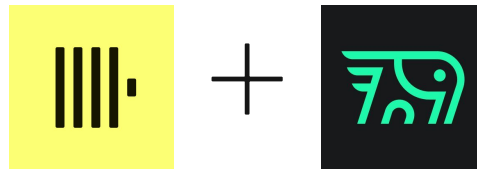
PeerDB Acquisition

What is PeerDB?

PeerDB is a Change Data Capture (CDC) provider focused on Postgres as a data source

Why acquire PeerDB?

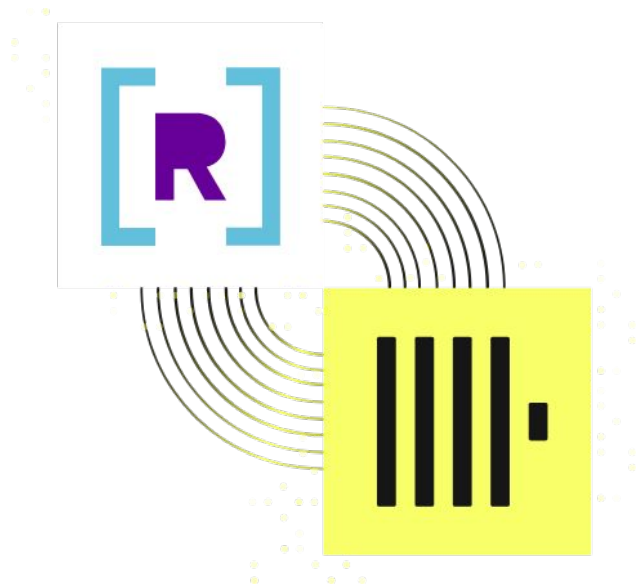
This will enable bridging the gap between transactional and analytical workloads and unlock more value for users and developers.



Rockset Migration Guide

Following Rockset's acquisition by OpenAI in June, users have until September 30 to off-board from the service. In comes, ClickHouse...

<https://clickhouse.com/docs/en/migrations/rockset>



BYOC on AWS

in private preview

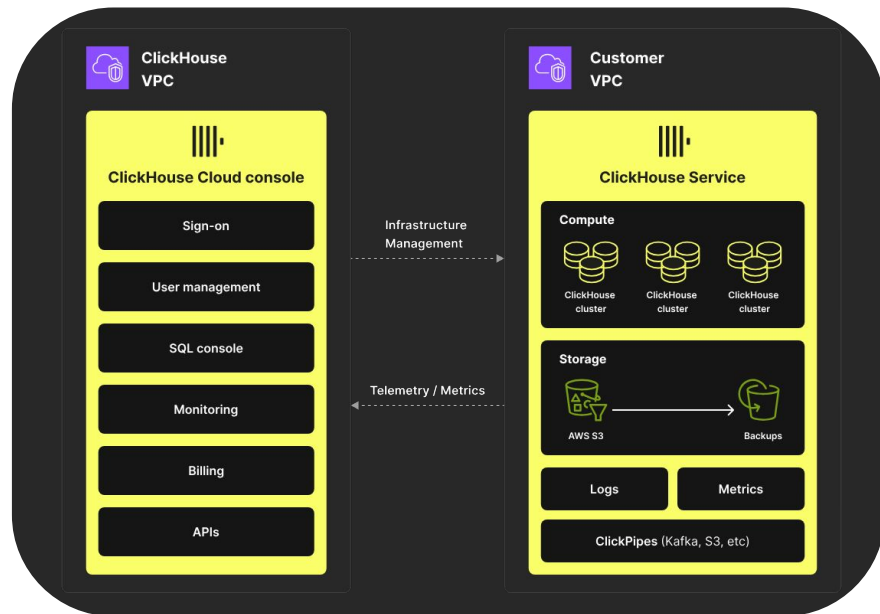
What is this?

Allows you to experience the advantages of ClickHouse Cloud within your own VPC.

Who is this for?

Organisations with strict data residency and compliance requirements.

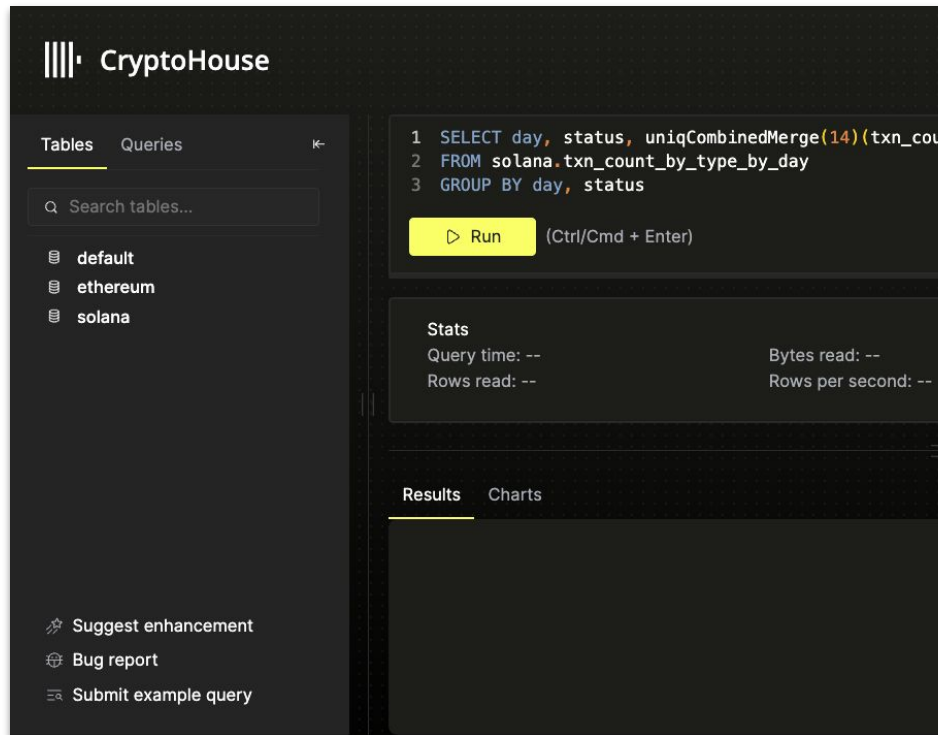
<https://clickhouse.com/cloud/bring-your-own-cloud>



New Demo: CryptoHouse

CryptoHouse is a free blockchain analytics service powered by ClickHouse with the aim of democratizing blockchain analytics.

<https://crypto.clickhouse.com/>



What's New with ClickHouse Academy

Existing Workshops:



Recently Added Workshops:



2-hour getting-started workshop
Level: Beginner

2-hour PostgreSQL to ClickHouse migration
Level: Intermediate



2-hour From BigQuery to ClickHouse migration
Level: Intermediate

Thank you!

Keep in touch!



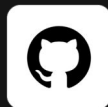
clickhouse.com/slack



clickhouse-australia-user-group



#clickhouseDB @clickhouseinc



clickhouse

ClickHouse +



Q&A