One Feature to Win it All

Materialized View a low key winning feature

About: Nahwin Rajan

- CTO & Co-founder In Stealth Mode
- Software Architect
  - High frequency Trading System
  - (+-) Reverse Engineered 1 million Request/minute with Go worker
  - Several Tokopedia’s Hero Feature (Hourly Flash Sale, Shop Home, etc)
- Head of Engineering
- Motogp Fans

Outline

- ClickHouse Materialized View
  - Clickhouse Magical Feature
  - You can skip “TL” in “ETL” (extract, transform, load)
- ClickHouse Remote Ingestion
  - Send your system native data to clickhouse without extra pre-processing
  - Come with vast acceptable input format
- “Automatic” Schema
  - Easily create and sync your schema through code

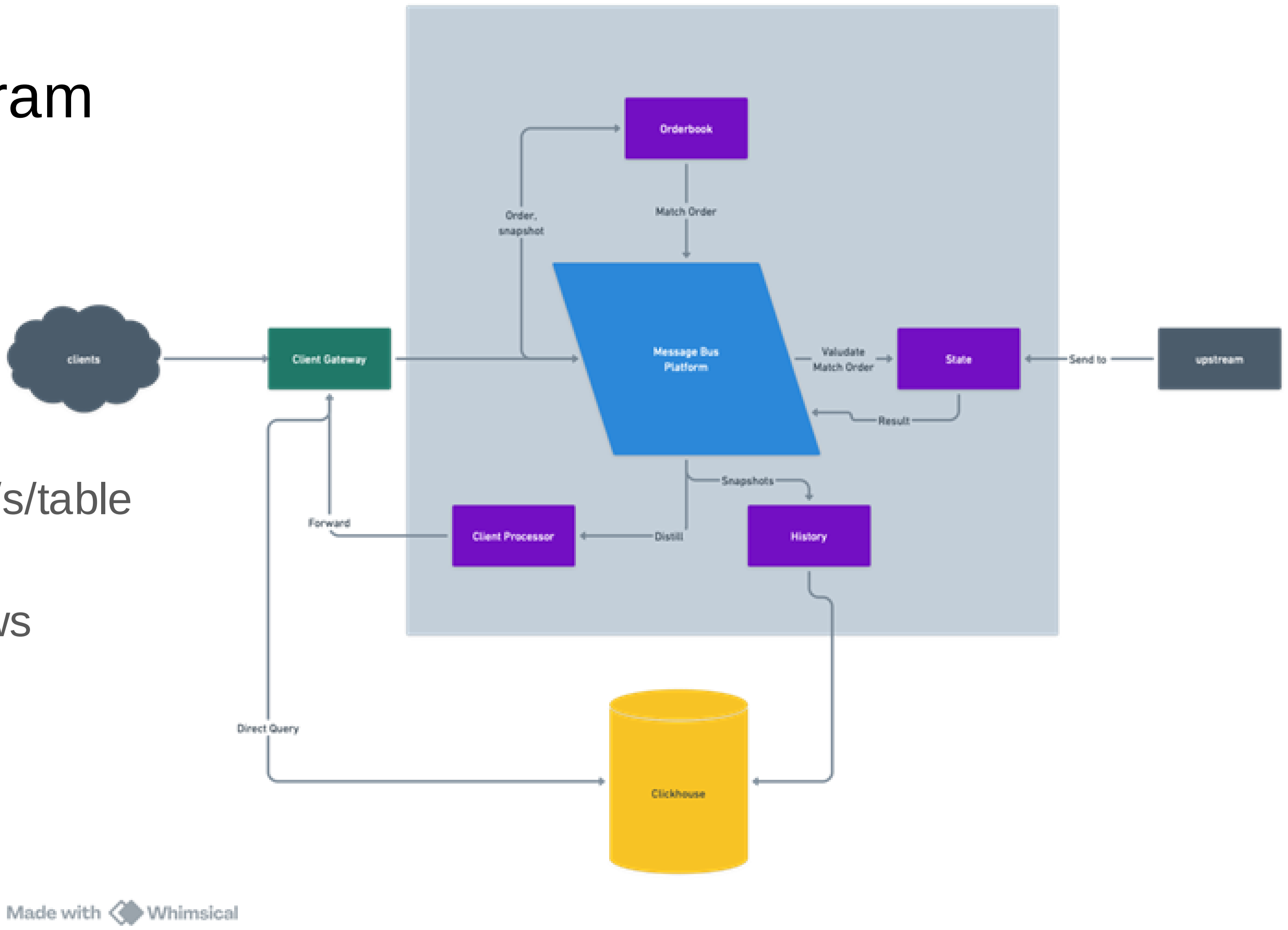
Technical Spec

- Stock Exchange Platform from Scratch
- Support High Frequency Trading
- 5ns to finish instruction once it reach our system
  - 1s/5ns => 200 million instruction per se second
- Everything on Memory
- 24/7
  - No Downtime
  - Stock Exchange does have Office Hours
    - Order for the day is written to Persistent Storage during Off time

System Diagram

System Diagram

- +- 200K rows/s/table
- 18 tables
- 3,6 Million rows



Why time series

- The right tools for the job
- Postgresql Ingestion Max Capacity +- 100K Rows
- Requirement
    - 200 Million Rows per Second
    - Near Real Time: Read (high Read and Write)
    - Processing Data for User View
    - 24/7

Why ClickHouse

Available Options:

InfluxDB, QuestDB, TimescaleDB, Clickhouse, Druid, Prometheus

Niche winning factor each db:

- Druid & Prometheus: yeah, lets stick them for monitoring purpose
- **InfluxDB**: is highly specialized for time-series data but lacks in relational or complex query use cases
- **TimescaleDB:** benefits from SQL compatibility and PostgreSQL's robust ecosystem

Why ClickHouse (2)

- **Clickhouse:** excels in real-time, high-performance analytics
- **QuestDB**: extreme fast ingestion

Basically Everyone Claim they are #1

- Uber's case with mysql and postgre

The differences are minutes

Clickhouse wasn't leading the selection 🤞

- When engineer fully focused working on work

INSIDER HELP!

The (original) Task

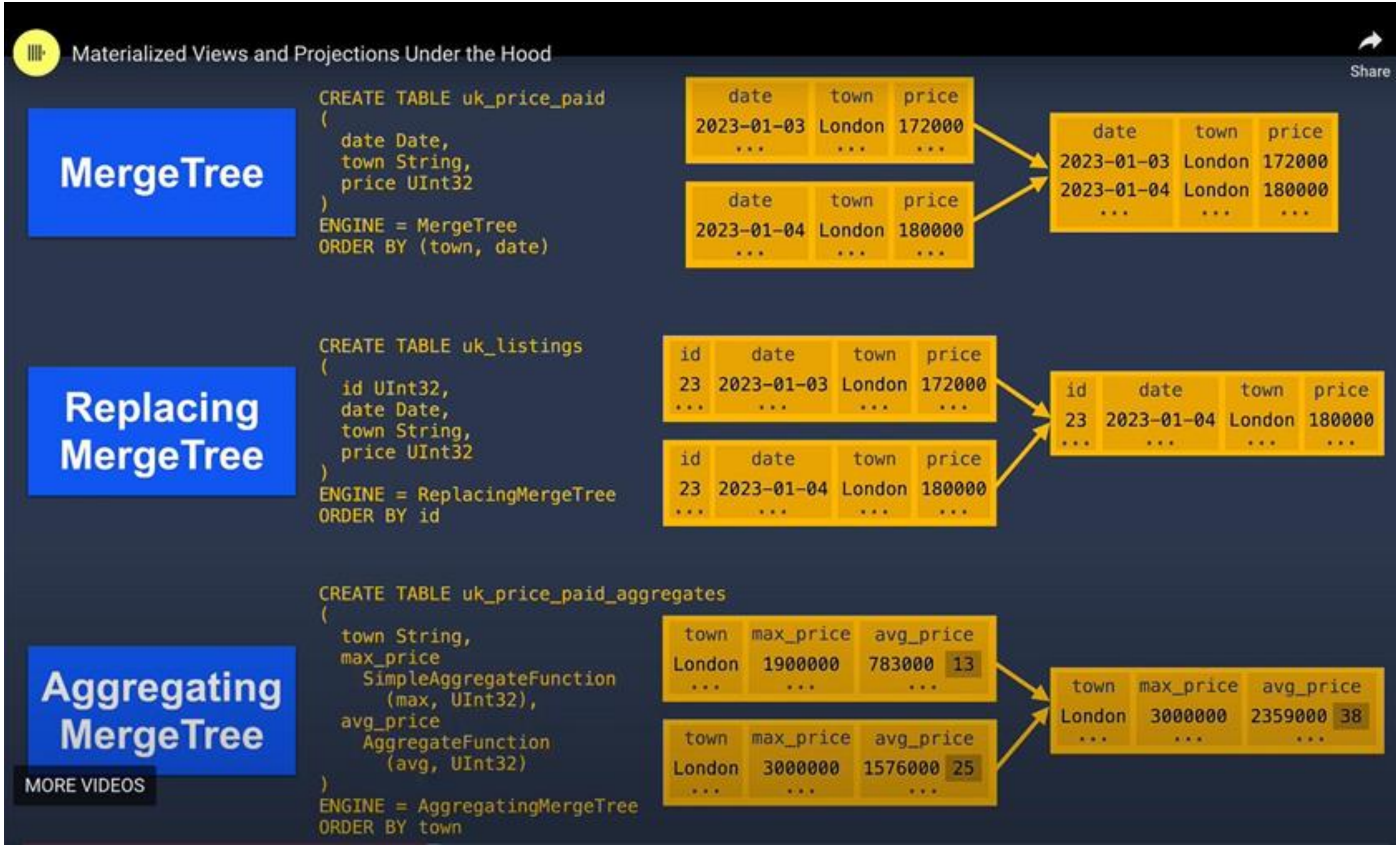
- Work on snapshot of the order
    - Create task to dump into disk
    - Create service to read the data
    - Transfer the data to database
- Creating schema for each service output
    - If payload structure changed, schema must be changed
    - Rippled to user view schema’s
- Creating useful schema for end-user (how trader will see their data)

Quotes

I will always choose a lazy person to do a difficult job because a lazy person will find an easy way to do it ~ Bill Gate (unsolicited)

Materialized View to the RESCUE

- Each TimeseriesDB does some sort of aggregation
- Clickhouse Materialized View is Low Key SUPER FEATURE
  - CLICKHOUSE GOT IT RIGHT WITH:
  - “AUTOMATIC INCREMENTAL DATA TRANSFORMATION”
- Pattern 3 table/views
  - **Source** -> **Materialized View** -> Target Table
  - Automate the Transformation of your Data using Materialized View



Materialized Views and Projections Under the Hood

MergeTree

```
CREATE TABLE uk_price_paid
(
  date Date,
  town String,
  price UInt32
)
ENGINE = MergeTree
ORDER BY (town, date)
```

date	town	price
2023-01-03	London	172000
...	...	...

date	town	price
2023-01-04	London	180000
...	...	...

date	town	price
2023-01-03	London	172000
2023-01-04	London	180000
...	...	...

Replacing MergeTree

```
CREATE TABLE uk_listings
(
  id UInt32,
  date Date,
  town String,
  price UInt32
)
ENGINE = ReplacingMergeTree
ORDER BY id
```

id	date	town	price
23	2023-01-03	London	172000
...	...	...	...

id	date	town	price
23	2023-01-04	London	180000
...	...	...	...

id	date	town	price
23	2023-01-04	London	180000
...	...	...	...

Aggregating MergeTree

```
CREATE TABLE uk_price_paid_aggregates
(
  town String,
  max_price
    SimpleAggregateFunction
      (max, UInt32),
  avg_price
    AggregateFunction
      (avg, UInt32)
)
ENGINE = AggregatingMergeTree
ORDER BY town
```

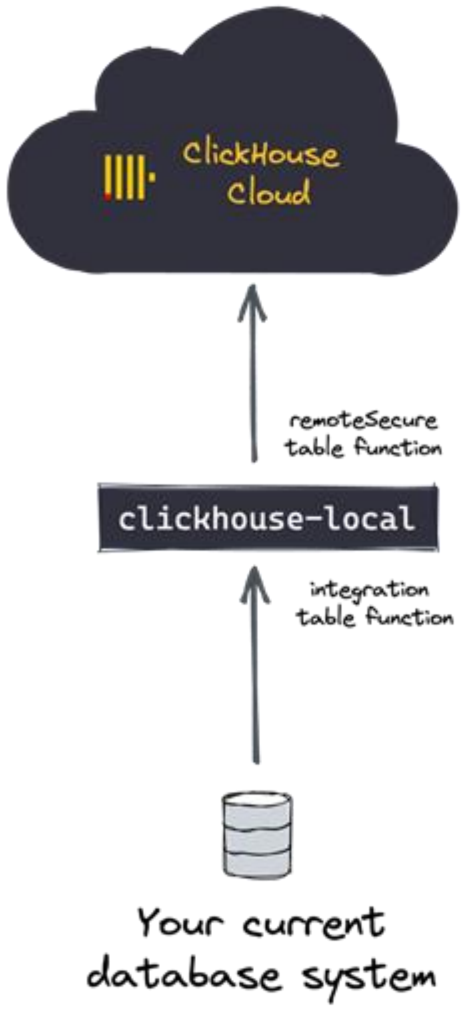
town	max_price	avg_price
London	1900000	783000 13
...	...	...

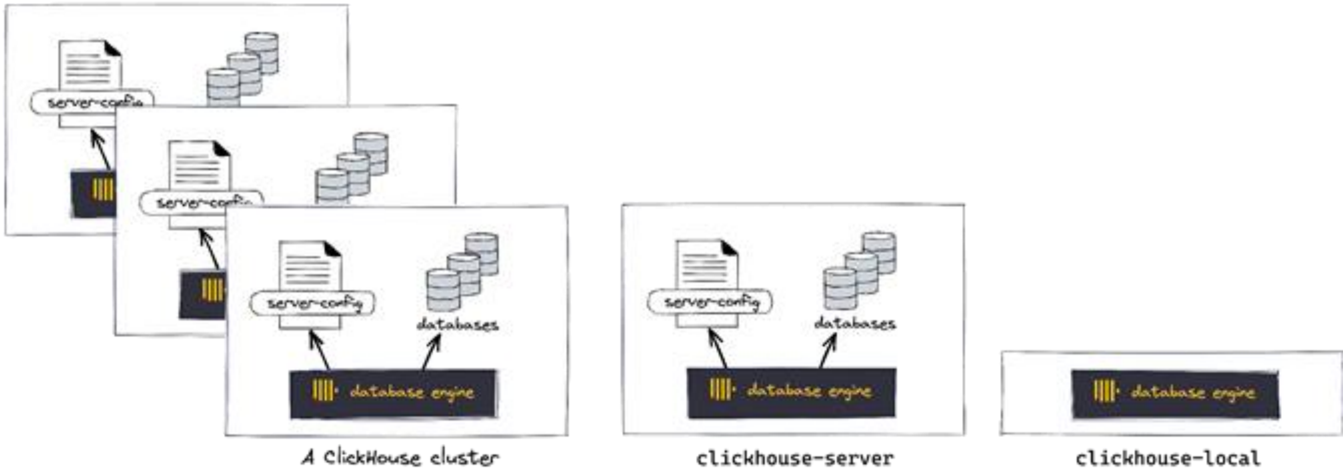
town	max_price	avg_price
London	3000000	1576000 25
...	...	...

town	max_price	avg_price
London	3000000	2359000 38
...	...	...

REMOTE INGESTION !!

- Turn out Clickhouse have REMOTE INGESTION Feature!
- And Clickhouse can work with multiple Format!!
    - Saving time to manually adjust the data into certain format
    - We use Binary format to directly inject to Clickhouse

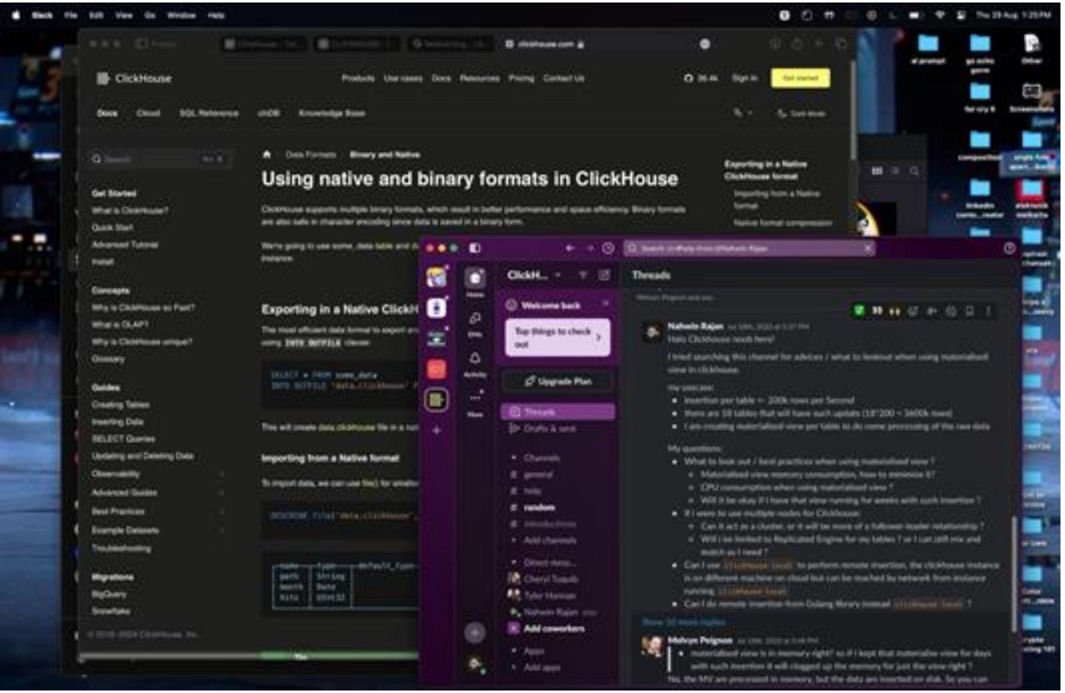

ClickHouse Cloud
remoteSecure table function
clickhouse-local
integration table function
Your current database system


server-config
databases
database engine
A ClickHouse cluster
clickhouse-server
clickhouse-local

The (original) Task

- Noticed the thing with Materialized View
  - Bye bye manual data processing
- Noticed the Remote ingestion & multiple Input Format
  - Bye bye labour intensive data loading
- Can be mix with flexible schema
  - Schema can be included with the data dump
  - Schema can be created from code (golang)
- Using the Self Hosting (open source option)
- Helpful community and tech affiliate in Slack Channel

Helpful Document and Community



ClickHouse

Products Use cases Docs Resources Pricing Contact Us

Docs Cloud SQL Reference chDB Knowledge Base

Data Formats / Binary and Native

Using native and binary formats in ClickHouse

ClickHouse supports multiple binary formats, which result in better performance and space efficiency. Binary formats are also safe in character encoding since data is saved in a binary form.

Exporting in a Native ClickHouse format

Importing from a Native format

Threads

Nahwin Rajan

Halo Clickhouse noob here!

I tried searching this channel for advices / what to lookout when using materialised view in clickhouse.

my usecase:

- Insertion per table += 200k rows per Second
- there are 18 tables that will have such update (18*200 = 3600k rows)
- I am creating materialised view per table to do some processing of the raw data

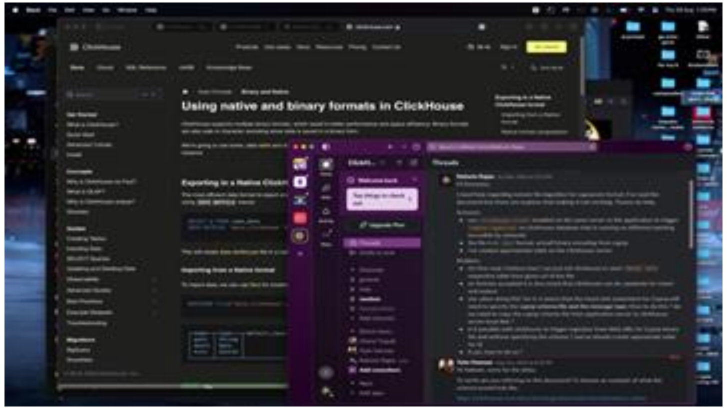
My questions:

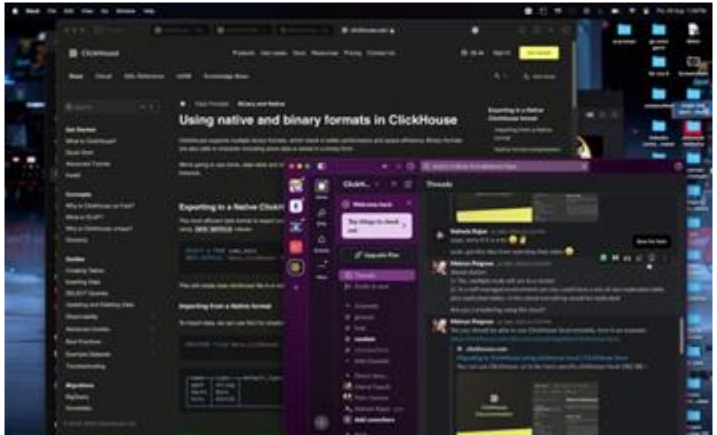
- What to look out / best practices when using materialised view ?
  - Materialised view memory consumption, how to minimize it?
  - CPU consumption when using materialised view ?
  - Will it be okay if I have that view running for weeks with such insertion ?
- If i were to use multiple nodes for Clickhouse:
  - Can it act as a cluster, or it will be more of a follower-leader relationship ?
  - Will i be limited to Replicated Engine for my tables ? or I can still mix and match as I need ?
- Can I use `clickhouse-local` to perform remote insertion, the clickhouse instance is on different machine on cloud but can be reached by network from instance running `clickhouse-local`
- Can I do remote insertion from Golang library instead `clickhouse-local` ?

Melvyn Peignon

- materialized view is in-memory right? so if i kept that materialize view for days with such insertion it will clogged up the memory for just the view right ?

No, the MV are processed in memory, but the data are inserted on disk. So you can





Possible Use case for Indonesia Tech Scene

- “Event Driven System” crave
    - Persist it to ClickHouse instead of Kafka
    - OR you no longer need all those intermediary service / process to persist your data
    - Save lot of work and pain when Persisted to Clickhouse with 3 tables Pattern
- Trading Fintech can also benefits
    - Play nieces with existing trading tech stacks and format
    - Can use managed service for ease of mind