How we processed 12 Trillion Rows during Black Friday

tinybird.co - @javisantana
12,213,675,632,435
12T - ~6 hours
(modest laptop)
The problem
Tinybird

SaaS product to build real time data products with high amounts of data

Not the sales guy here, but this gives context to this talk
4 weeks before Black Friday
Client [who does not want to who they are] in the retail space:

“Hey, we have a project for you...”
The use case
Report sales in real time for different countries for different areas in the company.

Our job: expose an API to feed their dashboards
The use case
That's easy
The use case: but wait, here is the reality

This is not a pretty dashboard with big numbers to post screenshots on twitter
The use case: but wait, here is the reality

Data origin: [legacy] transactional database (aka no change events, no CDC)

500+ concurrent users during that night

Real time

Multiple filters and configurable options, not just global counters
The use case: the data stream
Events with sales: product, units, amount... 12 columns
5 different data sources
5 batches per second with the last 5 minutes of data: lot of duplicated rows
The use case: the problems

Lot of read concurrency

Millions of rows per minute input

Deduplicate those rows
Ok, what’s the plan?
The approach: ideally

Serve static JSON files generated from the input.

Cheap, scalable and simple

Way too many different combinations (filters + configuration)
The approach: the decision, some background
We are boring, we tend to use technology with 15+ years. Also we did this in the past in previous companies
No actual time to make complex decisions and our product solves this problem
We use Clickhouse in our product since 2018 and looks a good fit
ClickHouse® is a fast open-source OLAP database management system. It is column-oriented and allows to generate analytical reports using SQL queries in real-time.

- Blazing fast
- Linearly scalable
- Feature-rich
- Hardware efficient
- Fault-tolerant
- Highly reliable

ClickHouse works 100-1000x faster than traditional approaches.

ClickHouse’s performance exceeds comparable column-oriented database management systems that are available on the market. It processes hundreds of millions to over a billion rows and tens of gigabytes of data per server per second.

Detailed comparison
The approach: the decision

Clickhouse has a nice feature: [Incremental] Materialized views

<table>
<thead>
<tr>
<th>slower // flexible</th>
<th>faster // fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query raw data</td>
<td>Materialized views</td>
</tr>
<tr>
<td></td>
<td>JSON files</td>
</tr>
</tbody>
</table>
The solution
Infrastructure to handle 200-300QPS

Nginx → Varnish → API → Varnish → Clickhouse

- Just https
- Load balancing
- Grace mode
- Python
- Data import
- API Endpoints
- Load balancing
- Health checks
- Database engine
  (+ zookeeper)
The solution: first problem, dealing with import batches

Kafka?

Data is sent in batches

Append only “landing” table leveraging Clickhouse super fast data import (also multimaster)

Generate all the views based on that “landing” table
The solution: dealing with import data
Upserts - clickhouse is not the best one here
30 minutes window to upsert
Solution: real time + historic tables
2 MV (for RT and historic) to cover 90% the API endpoints
Simple, fast to generate MV
The solution: some small but important details

Deal with replication lag

Materialized views must fit in memory

Leverage caching and indexing as much as possible
The solution: endpoints, aka queries to Clickhouse

Objective: 1 core - 150ms (q95)

Materialize data taking into account the (estimated) call distribution

Data driven optimization: a simple spreadsheet
WITH ( 
  select split_date from split_table 
) as split_date

select from historic where date < split_date 
union all

WITH ( 
  select split_date from split_table 
) as split_date

select from RT where date >= split_date
Black Friday day[s]
650B rows ingested
12T rows queried
50QPS median, 300QPS peak
600ms q95 response time by the end of the BF :( 
Thank you!

https://blog.tinybird.co/2020/12/21/how-we-setup-real-time-analytics-service-to-process-12-trillion-rows-during-black-friday/

https://tinybird.co

@javisantana