High-Performance Distributed DBMS for Analytics

Victor Tarnavsky, Head of Analytic Systems Department
ClickHouse

Story
Yandex

› One of the largest internet companies in Europe
› Over 5000 employees
› Top-1 Search in Russia
› More than 50 different b2c and b2b products
› Big Data, Machine Learning
Yandex.Metrica is 2nd largest web analytics tool in the world

› 30+ billions of events daily
› Millions of websites
› 100+ thousands of analysts every day

We need fast and feature-rich database capable to handle our clients data
Yandex.Metrica

Traffic source

Sessions

- Direct traffic: 49.3%
- Link traffic: 24.5%
- Internal traffic: 18.3%
- Search engine traffic: 7.67%
- Social network traffic: 0.099%
- Other: 0.038%

2016.09.19 10:30, Monday
Internal traffic
Sessions: 374
Before ClickHouse

2008-2011: MySQL
  › Slow and tricky

2010-2015: Metrage, custom aggregated data storage
  › Fast and realtime
  › Aggregated data can't be filtered
Requirements

- Fast. Really fast
- Data processing in real time
- Capable of storing petabytes of data
- Fault-tolerance in terms of datacenters
- Flexible query language
Nothing? Oh. Well...
The main ideas behind ClickHouse

› SQL
› Linearly scalable
› Focused on fast query execution
› Realtime
› Column-oriented
ClickHouse

Today
ClickHouse today

› Open-source, Apache 2.0
› 100+ companies outside Yandex
› Strong community
› Active development

yandex / ClickHouse  Watch  181  Unstar  2,293  Fork  285
Opensource

› Apache 2.0
› 53 committers on main repo
› Committers from CloudFlare, booking.com
› 100+ companies using ClickHouse
Community

› 700+ people in Telegram chats, active every day
› 102 side projects on GitHub: drivers, clients, interfaces etc.
› Tabix: web interface over ClickHouse
› Integrations:
  Grafana
  Redash
  Apache Zeppelin
  Superset
  Power BI
ClickHouse

Features
Scalable

› Petabytes of data
› Cross-datacenter
› High availability
› Data compression
Metrica Cluster

- 20+ trillions of rows
- 3 Pb
- 450 Servers
- 6 Datacenters
- Up to 2 terabytes per second on query processing
Querying

› SQL dialect + extensions
› Additional features: approximate functions, URI functions and more
› Arrays, nested data types
› Distributed out of the box
› Pluggable external key-value dictionaries
# Weekly traffic and audience

SELECT
    count() as visits,
    sum(PageViews) as hits,
    uniq(UserID) as users
FROM visits_all
WHERE StartDate > today() - 7
# Using dictionary for regions

SELECT
    count() as visits,
    regionToName(regionToCountry(RegionID), 'en') as country
FROM visits_all
WHERE StartDate > today() - 7
GROUP BY country
ORDER BY visits DESC
LIMIT 10
Performance

- Sub-second query latency
- >100x faster than Hadoop,
  >100x faster than typical DBMS
- Up to a few billion rows/second per single node
- Up to 2 terabytes per second on clustered setup of 400 nodes
## Relative query processing time (lower is better):

<table>
<thead>
<tr>
<th>Database</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClickHouse</td>
<td>1.00</td>
</tr>
<tr>
<td>(1.1.53960)</td>
<td></td>
</tr>
<tr>
<td>Hive</td>
<td>168.19</td>
</tr>
<tr>
<td>(0.11, ORC File)</td>
<td></td>
</tr>
<tr>
<td>MySQL</td>
<td>511.57</td>
</tr>
<tr>
<td>(5.5.32, MyISAM)</td>
<td></td>
</tr>
</tbody>
</table>

## Relative query processing time (lower is better):

<table>
<thead>
<tr>
<th>Database</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClickHouse</td>
<td>1.00</td>
</tr>
<tr>
<td>(1.1.53960)</td>
<td></td>
</tr>
<tr>
<td>Vertica</td>
<td>3.38</td>
</tr>
<tr>
<td>(7.1.1)</td>
<td></td>
</tr>
<tr>
<td>InfiniDB</td>
<td>20.91</td>
</tr>
<tr>
<td>(Enterprise 3.6.23)</td>
<td></td>
</tr>
<tr>
<td>MonetDB</td>
<td>21.51</td>
</tr>
</tbody>
</table>

More info: [https://clickhouse.yandex/benchmark.html](https://clickhouse.yandex/benchmark.html)
ClickHouse vs. Spark and MariaDB ColumnStore

More benchmarks

› 1.1 Billion Taxi Rides on ClickHouse by Mark Litwintschik
 http://tech.marksblogg.com/billion-nyc-taxi-clickhouse.html

› ClickHouse vs. other Open-source Databases by Percona
 http://bit.ly/2pf9aF
Interfaces

› Console client
› HTTP
› JDBC Driver, ODBC Driver in beta
› Clients for:  
  Python, PHP, Go, Node.js  
  Perl, Ruby, R, C++  
  .NET, Scala, Julia
ClickHouse

Inside
Why is it so fast?

Code

› Vectorized query execution
› Low-level optimisations and specialisations
› Every piece of code is tested in terms of performance
Why is it so fast?

Data

› Column-oriented
› Merge Tree
  minimal number of seeks
› All processing as close to data as possible
Why is it so fast?

Features

› Sampling
› Approximate functions
› Performance tuning even on a request level
Scalability and fault-tolerance

- Hours of downtime on Metrica cluster for years
  - Cross-datacenter
  - Asynchronous replication
  - Eventual consistency
Cluster Scheme

Datacenter

— Shard
ClickHouse: wrong cases

- Not an OLTP
- Not a key-value store
- Not a document store
- Do not modify your data (you don't need that)
ClickHouse: best practices

- A few wide tables with a lot of columns
- Structured data
- QPS is relatively low but data usage per request is high
- Huge amounts of data incoming
- Petabytes of data
Typical use cases

- Adv networks data
- RTB
- Web/App analytics
- Ecommerce
- Telecom logs
- Online games
- Sensor data
- Monitoring
- Messengers
Unusual applications

› Blockchain search and analytics engine
  https://blockchair.com/

› Evolutionary genetics and genomics (analyzing BLAST database)
  https://github.com/msestak/FindOrigin

› Nuclear research: CERN's LHCb experiment
Case: server log analysis

Common first case for new ClickHouse users.
Estimated time: few hours

› Insert access logs into ClickHouse
› Analyze incidents with instant queries
› Monitoring reports: error rates, response timings and more
Case: in-house analytics database

Build your own data warehouse and dig your data in seconds.

› Take your Hadoop or other 'not so fast' storage and want to do things faster
› Copy all your data to ClickHouse
› Build internal dashboards/metrics
› Do realtime analysis of your business process
Wrap up
ClickHouse briefly

- Open-source column-oriented DBMS
- Linearly scalable
- Blazingly fast
- SQL dialect with extensions
- Strong community
ClickHouse

How to start?
› Try our tutorial: https://clickhouse.yandex/tutorial.html
› Ask anything: clickhouse-feedback@yandex-team.com
› GitHub: https://github.com/yandex/ClickHouse
› Telegram: https://t.me/clickhouse_en
› More info: https://clickhouse.yandex
Contacts

Viktor Tarnavskiy

Head of Analytic Products Department in Yandex

jkee@yandex-team.ru

clickhouse-feedback@yandex-team.ru