Altinity WEBINAR

ADDING FAST ANALYTICS TO MySQL APPLICATIONS WITH CLICKHOUSE

with Robert Hodges

Introduction to Presenter

Robert Hodges - Altinity CEO

30+ years on DBMS plus virtualization and security.

ClickHouse is DBMS #20

Altinity

www.altinity.com

Leading software and services provider for ClickHouse

Major committer and community sponsor in US and Western Europe

Introduction to ClickHouse

SQL optimized for analytics

Runs on bare metal to cloud

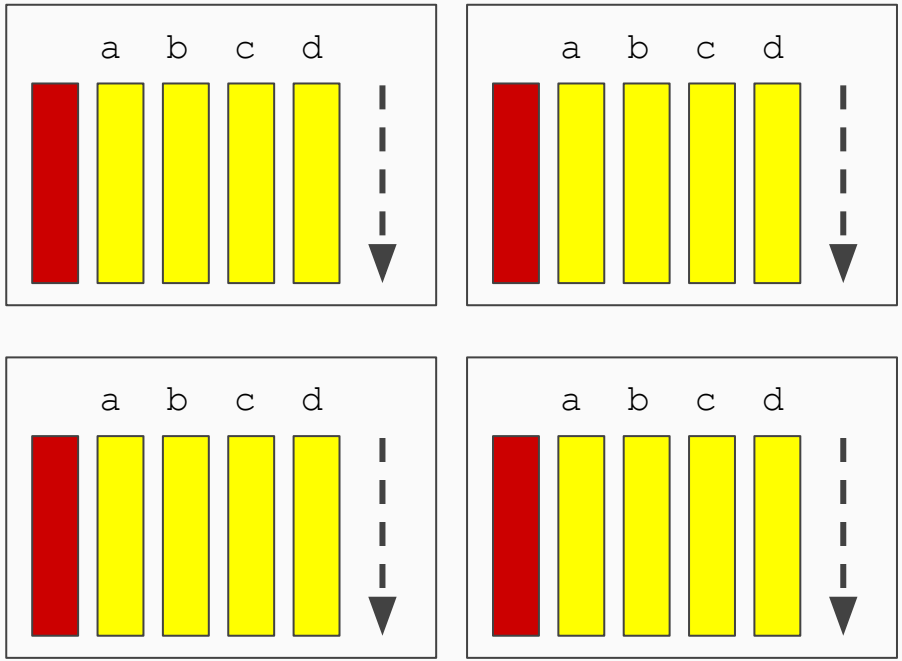
Stores data in columns

Parallel and vectorized execution

Scales to many petabytes

Is Open source (Apache 2.0)

Is WAY fast on analytic queries



Introduction to MySQL

Full SQL implementation

Runs on bare metal to cloud

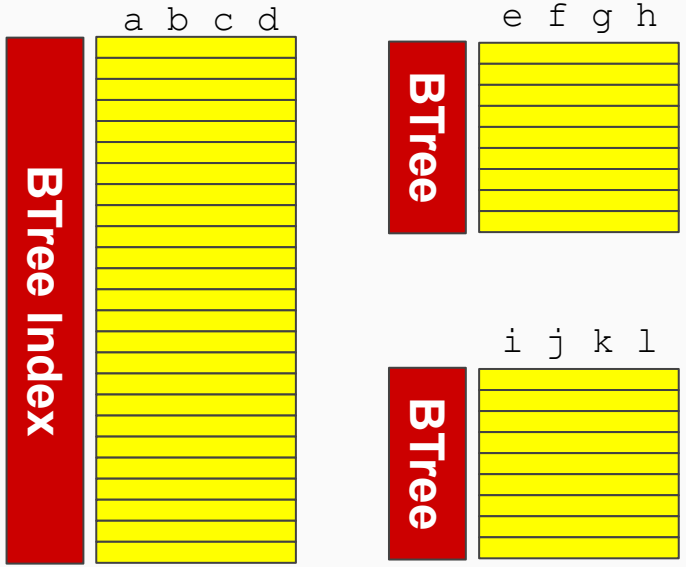
Stores data in **rows**

Single-threaded, concurrent query

Scales to **high transaction loads**

Is Open source (**GPL V2**)

Is WAY fast on **updates & point queries**



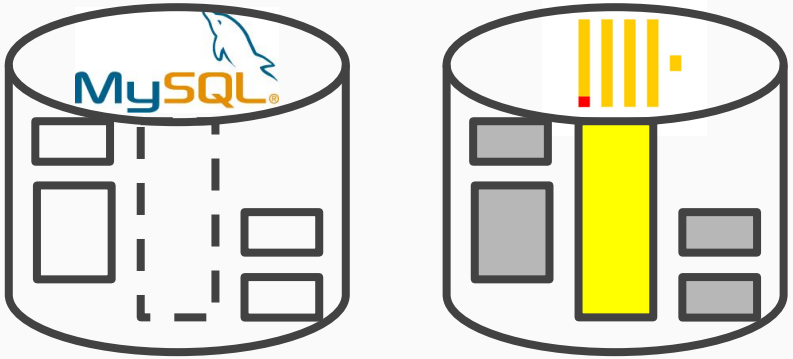
MySQL Trade-Offs

Queries on large MySQL tables are resource-intensive and inefficient....

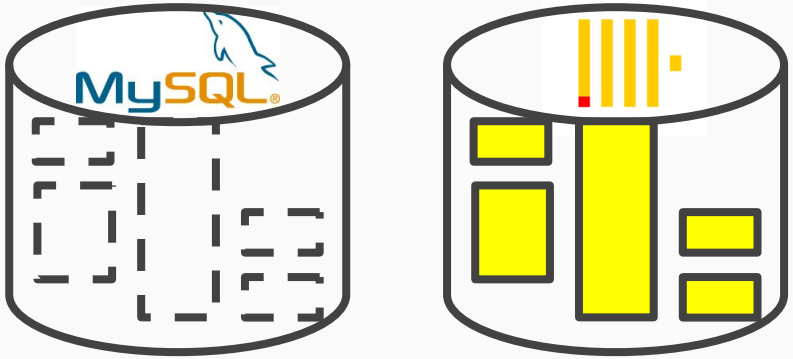
- Enormous I/O load due to row organization
- Careful indexing required
- Compression of limited value
- Parallel query limited/unavailable
- Highly dependent on buffer pool size

For rows > 100M MySQL analytic results are very slow

Options for ClickHouse query acceleration



Move big tables, keep dimensions in MySQL



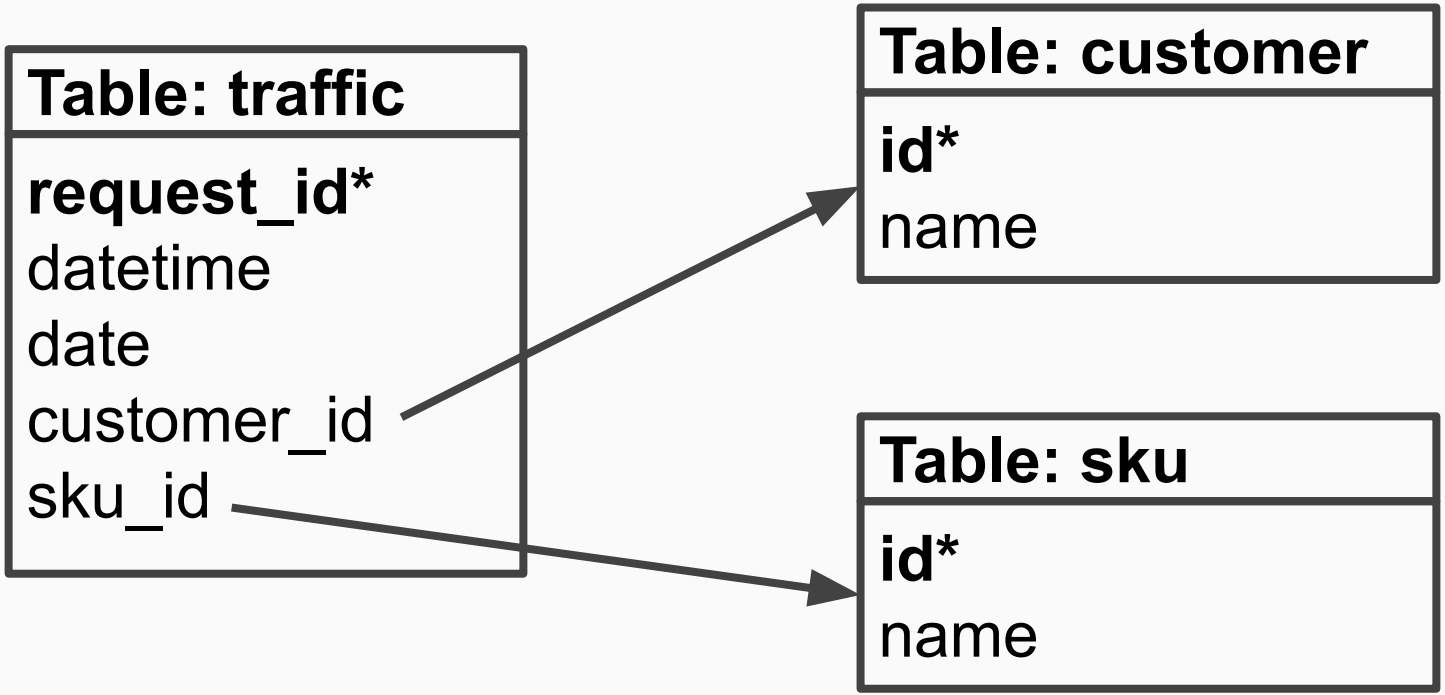
Full migration to ClickHouse

Other combinations are possible...

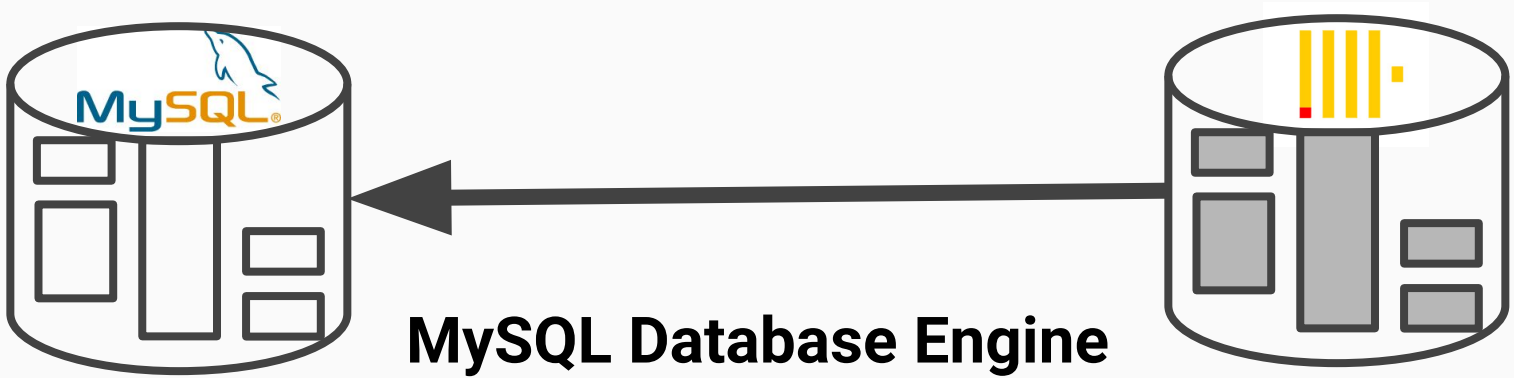
Accessing MySQL Tables from ClickHouse

MySQL sample tables

Database: repl



Accessing MySQL data from ClickHouse



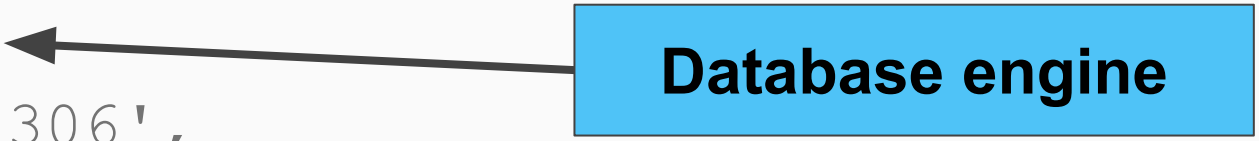
MySQL Database Engine
MySQL Table Function
MySQL Table Engine
MySQL Dictionary

Selecting data from tables in MySQL

```
-- Select data from all tables.
SELECT
  t.datetime, t.date, t.request_id,
  c.name customer, s.name sku
FROM traffic t
  JOIN customer c ON t.customer_id = c.id
    JOIN sku s ON t.sku_id = s.id
LIMIT 10;
```

Access a MySQL database from ClickHouse

```
CREATE DATABASE mysql_repl
ENGINE=MySQL(
   '127.0.0.1:3306',
   'repl',
   'root',
   'secret')

use mysql_repl
show tables
```

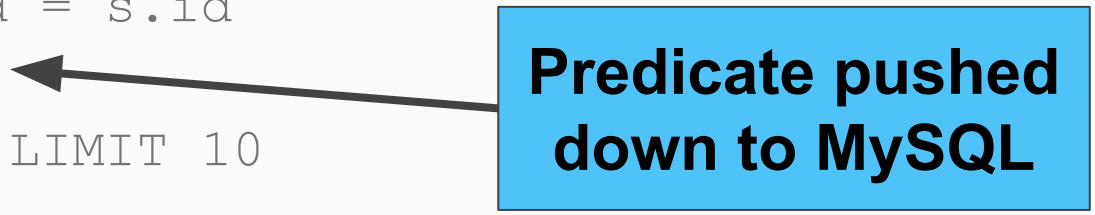


Navigating MySQL tables from ClickHouse

Demo

Selecting data from MySQL

```
SELECT
  t.datetime, t.date, t.request_id,
  t.name customer, s.name sku
FROM (
  SELECT t.* FROM traffic t
  JOIN customer c ON t.customer_id = c.id) AS t
JOIN sku s ON t.sku_id = s.id
WHERE customer_id = 5
ORDER BY t.request_id LIMIT 10
```



ClickHouse performance beats MySQL!

Query	MySQL processing time (seconds)	MySQL-> ClickHouse processing time (seconds)	ClickHouse processing time (seconds)
groupby(data1)	3.22	2.685	0.071
groupby(data2)	4.01	2.692	0.177
groupby(data3)	212.82	5.236	0.779
groupby(data1) +uniq(data5)	183.56	12.944	1.725
groupby(data1) +uniq(data5) Approximate	-	6.026	0.285

Transferring data from MySQL Engine

```
-- Create a ClickHouse table from MySQL.
CREATE TABLE traffic as repl.traffic
ENGINE = MergeTree
PARTITION BY toYYYYMM(datetime)
ORDER BY (customer_id, date)

-- Pull in MySQL data.
INSERT INTO traffic SELECT *
FROM mysql_repl.traffic

SELECT count(*) FROM traffic
```

Accessing data using MySQL table function

```
SELECT t.datetime, t.date, t.request_id, c.name customer
FROM traffic t
JOIN
  mysql('127.0.0.1:3306', 'repl', 'customer', 'root',
'secret') c
ON t.customer_id = c.id
WHERE t.customer_id = 5
ORDER BY t.request_id LIMIT 10
```

Predicate pushdown works from base table

Accessing data using MySQL table engine

```
CREATE TABLE mysql_customer (
  id Int32,
  name String
)
ENGINE = MySQL(127.0.0.1:3306', 'repl', 'customer', 'root',
'secret')

SELECT t.datetime, t.date, t.request_id, c.name customer
FROM traffic t
JOIN mysql_customer c ON t.customer_id = c.id
ORDER BY t.request_id LIMIT 10
```

Access a MySQL table using a Dictionary

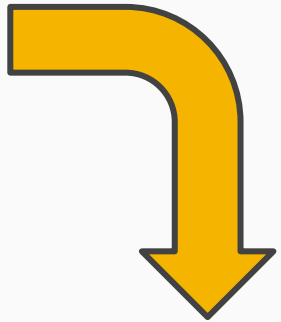
```
<yandex>
  <dictionary><name>mysql_sku</name>
    <source> <mysql>
       <host>localhost</host> <port>3306</port><user>root</user>
       <password>********</password><db>repl</db> <table>sku</table>
    </mysql> </source>
    <layout> <hashed/> </layout>
    <structure>
        <id> <name>id</name> </id>
        <attribute>
            <name>name</name> <type>String</type> <null_value></null_value>
        </attribute>
    </structure>
    <lifetime>0</lifetime>
  </dictionary>
</yandex>
```

Select local, remote, and dictionary data

```
SELECT
    t.datetime,
    t.date,
    t.request_id,
    c.name AS customer,
    dictGetOrDefault('mysql_sku', 'name',
                     toUInt64(sku_id), 'NOT FOUND') AS sku
FROM traffic AS t
INNER JOIN mysql_customer AS c ON t.customer_id = c.id
ORDER BY t.request_id ASC
LIMIT 10
```

Figuring out what's happening on MySQL

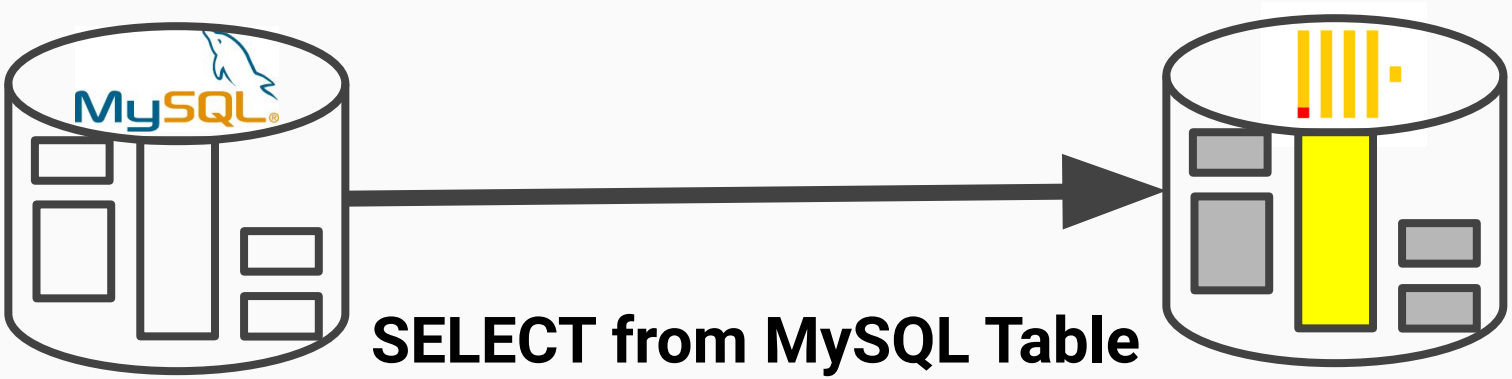
```
-- Enable MySQL query log
set global general_log=1;
```



```
(MySQL query log)
16 Connect   root@localhost on repl using TCP/IP
16 Query  SET NAMES utf8
16 Query  SELECT `id`, `name` FROM `repl`.`sku` WHERE `id` = 3
```

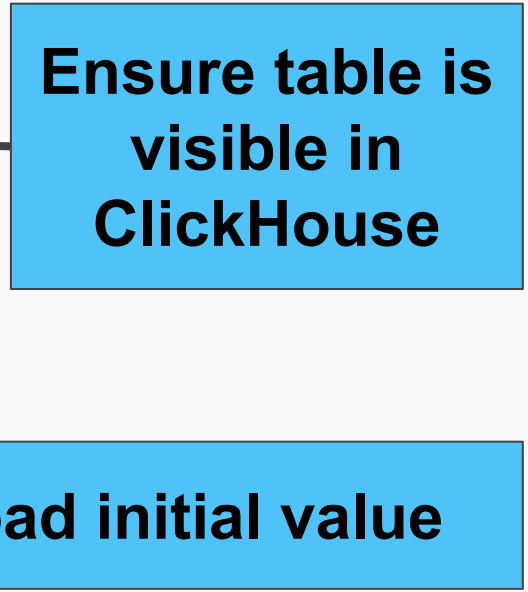
Automatic Propagation of Changes

Replicate MySQL Data to ClickHouse



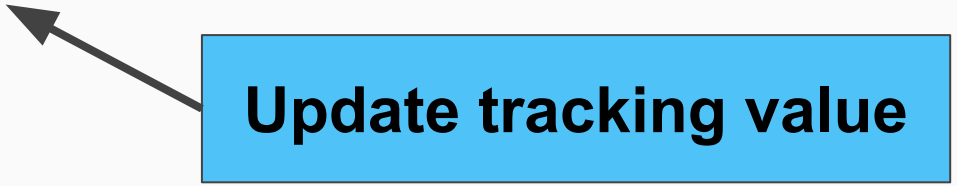
SELECT Method: Setup

```
-- Create a tracking table on MySQL side.
CREATE TABLE last_request_id (
  id bigint
);
INSERT INTO last_request_id VALUES (-1);
```



SELECT Method: Change Propagation

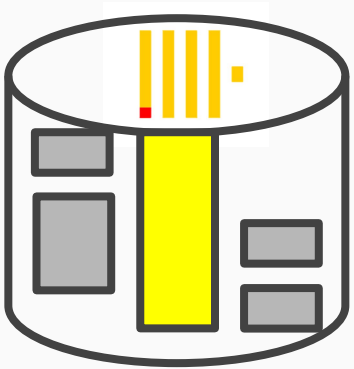
```
INSERT INTO traffic SELECT *
FROM mysql_repl.traffic
WHERE request_id >
(
    SELECT max(id)
    FROM mysql_repl.last_request_id
)

INSERT INTO mysql_repl.last_request_id SELECT max(request_id)
FROM traffic
```



MySQL Replication Method: Setup

(1) Ensure MySQL table(s) have primary keys

(2) Enable row replication



(3) Install and run clickhouse-mysql

my.cnf:

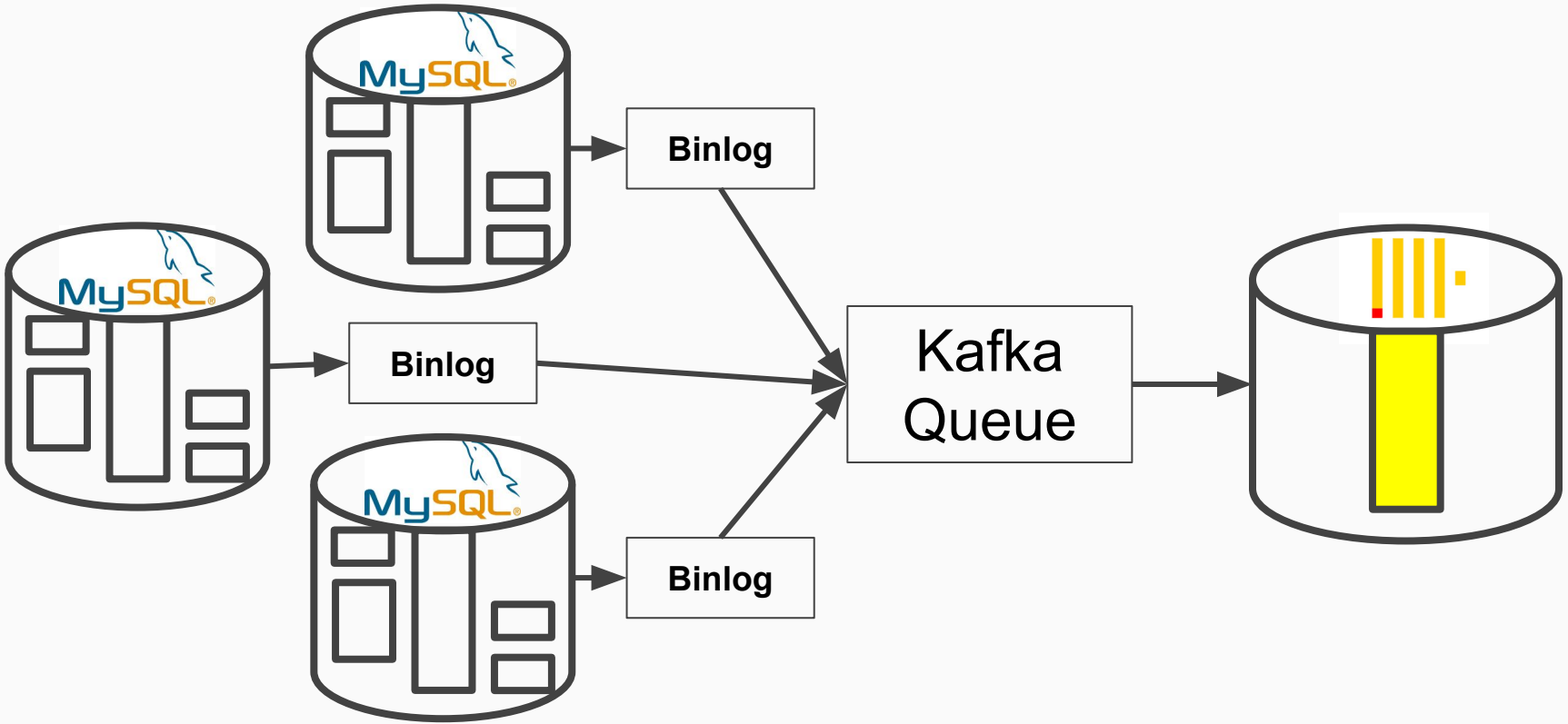
```
server-id  = 1
log_bin  = /var/log/mysql/mysql-bin.log
expire_logs_days  = 10
max_binlog_size   = 100M
binlog-format = row
```

https://github.com/Altinity/clickhouse-mysql-data-reader

MySQL Replication: Change Propagation

Demo

Kafka Method: Discussion



Food for thought

- ClickHouse works best on wide tables
- Consider triggers to add dimension info to base rows on MySQL
- Replication methods are complex to operate
- Use Kafka when many MySQL instances generate data
- Best approach is to migrate large tables completely to MySQL
  - No replication to manage
  - MySQL runs faster and requires fewer resources

ClickHouse MySQL Client Support

ClickHouse supports MySQL clients??!

```
<?xml version="1.0"?>
<yandex>
    ...
    <!-- Enable MySQL wire protocol. -->
    <mysql_port>33306</mysql_port>
    ...
</yandex>
```

Here’s the proof!

```
mysql -h127.0.0.1 -P33306 -udefault --password=''
...
mysql> use mysql_repl
mysql> SELECT
    ->   t.datetime, t.date, t.request_id,
    ->   t.name customer, s.name sku
    -> FROM (
    ->   SELECT t.* FROM traffic t
    ->   JOIN customer c ON t.customer_id = c.id) AS t
    -> JOIN sku s ON t.sku_id = s.id
    -> WHERE customer_id = 5
    -> ORDER BY t.request_id LIMIT 10;
...
```

Wrap-up

Key Takeaways

- ClickHouse has multiple ways to access data in MySQL
- Use replication to pull changes, if you have to
- ClickHouse supports MySQL Protocol so clients can connect directly
- Keep approach as simple as possible for maximum joy

ClickHouse can query MySQL data faster than MySQL!*

***Your mileage may vary**

Thank you!

Special Offer:

Contact us for a
1-hour consultation!

Contacts:
info@altinity.com

Visit us at:
https://www.altinity.com

ClickHouse-MySQL:
https://github.com/Altinity/clickhouse-mysql-data-reader

Free Consultation:
https://blog.altinity.com/offer