

How to analyze trillions data with ClickHouse

虎牙信息 HUYA

李本旺 sundy-li

Github: <http://github.com/sundy-li>

Mail: libenw1992@gmail.com

APM

覆盖全公司上报数据

- 主播端 带宽, 卡顿, PCU
- 客户端 卡顿, API 请求, 信令通道
- TAF 调用
- 基础监控 Open-Falcon
- CDN,P2P,UDB ...

3000亿 +
日均增量

400+
业务指标



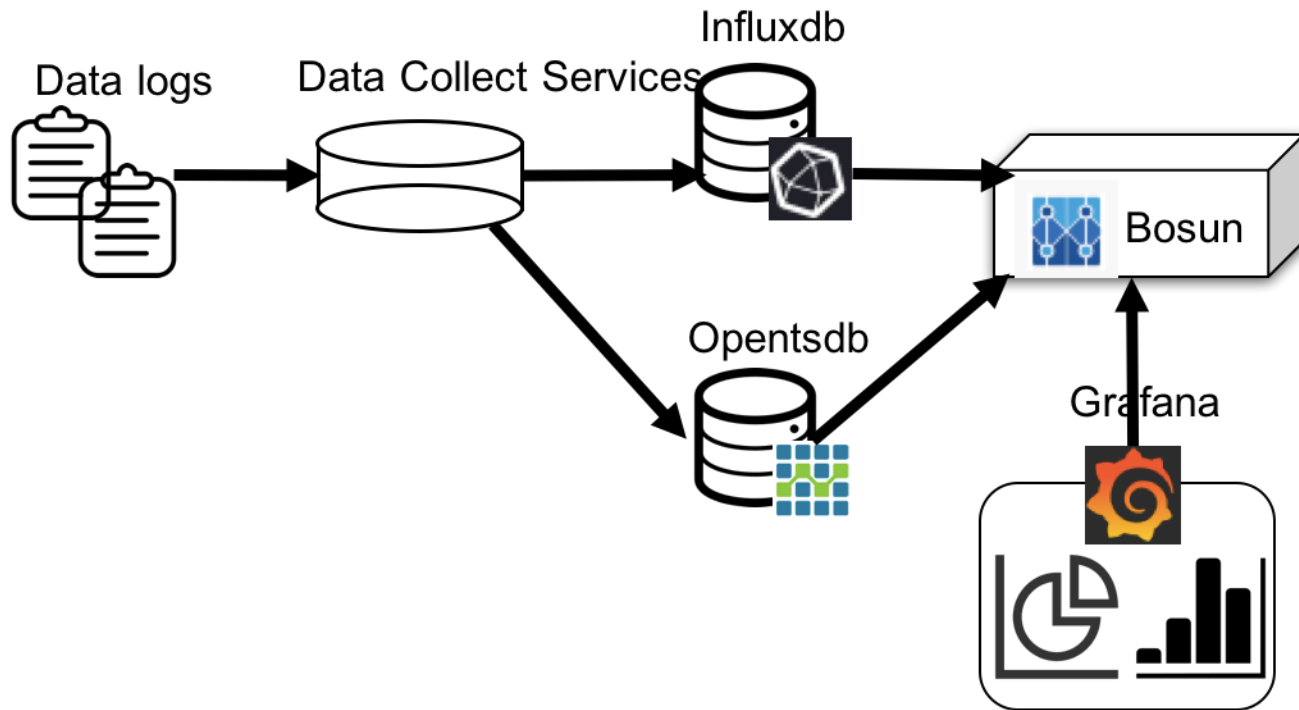
HUYA

实时监控
秒级响应

500w +
每秒新增

3个集群, 近百节点部署
覆盖海内外服务

Previous architecture



Features:

- Real-time pre-calculation
- Opentsdb Seriously dependent on hbase
- Opentsdb poor aggregation ability
- Influxdb is not clustered
- Not friendly with OLAP
- Poor performance
- Not flexible
- ...

Why ClickHouse

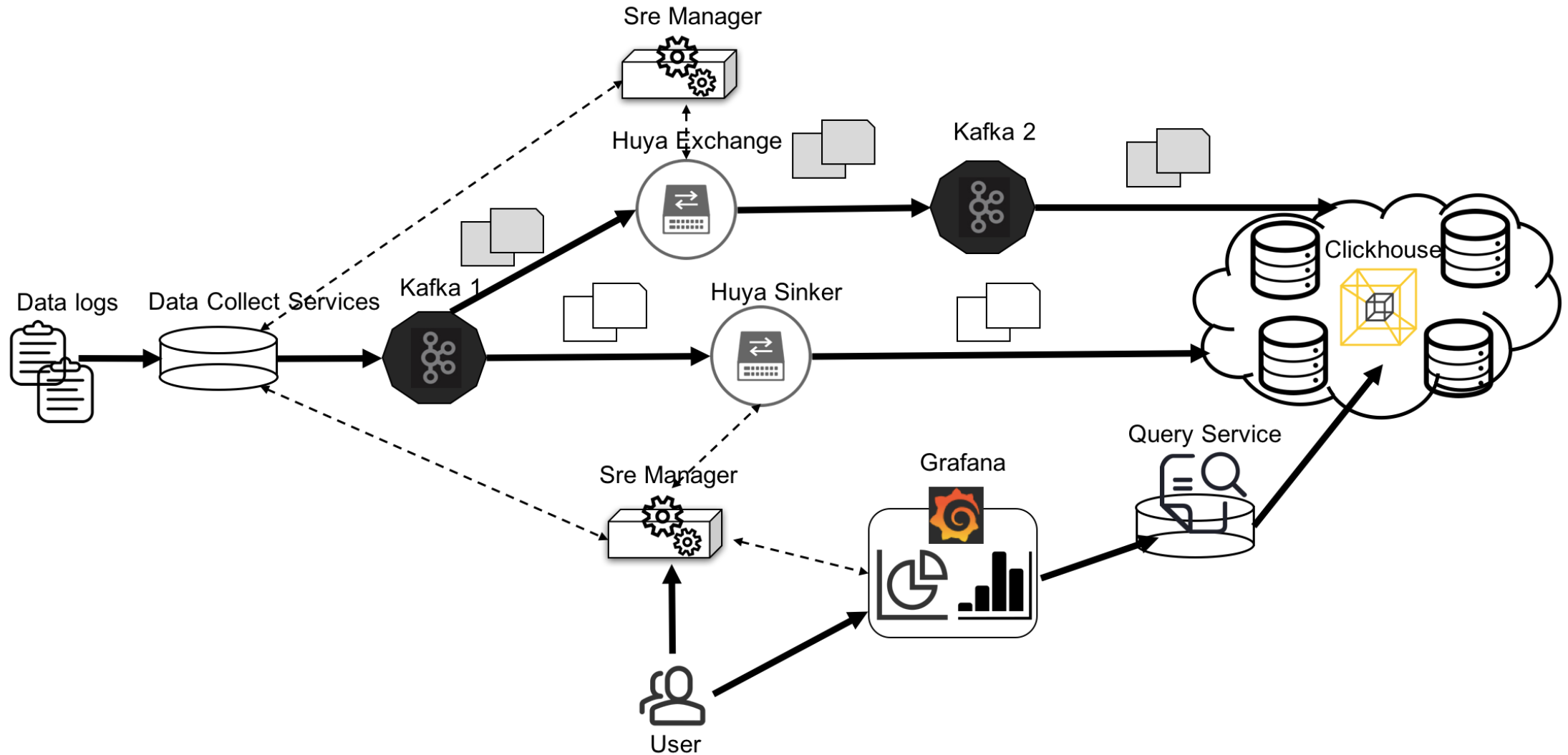
- Column Oriented
- Super Fast
- OLAP
- SQL supported
- Linearly Scalable
- Simple

ClickHouse as time-series database

Get the average value group by line, interval 20s

```
SELECT
    intDiv(its, 20) * 20 AS _timestamp,
    sum(value) / sum(_cnt) AS `卡比`,
    line AS `线路`
FROM view.
WHERE
    (day >= toDate(1540278823)) AND (day <= toDate(1540279123))
AND
    (its >= 1540278823) AND (its <= 1540279123) AND (line IN (1, 3, 5)) AND (platform = 'adr')
GROUP BY
    line,
    _timestamp
ORDER BY _timestamp ASC
LIMIT 50000
FORMAT JSONCompact
```

Current architecture



Best practice start

Realtime ingestion

- Kafka Engine(×)

Not Recommended

Because

- Error handling
- Kafka partition && consumer auto-rebalance
- ClickHouse is shared-nothing architecture
- Bad control of data sharding

Kafka SETTINGS

```
kafka_broker_list = 'localhost:9092',  
kafka_topic_list = 'topic1,topic2',  
kafka_group_name = 'group1',  
kafka_format = 'JSONEachRow',  
kafka_row_delimiter = '\n',  
kafka_schema = '',  
kafka_num_consumers = 2
```

Realtime ingestion

- How ?

huya_sinker

- Golang implemented
- Auto rebalance, easy to deploy as a container
- Dynamic awareness tables and columns configuration (etcd)
- Batch insert (interval, batch_size)
- Custom parser support
- Message JSON format
- Use [Go tcp client](#), use [gjson](#) parser
- Random write Strategy
- On parallel
- High rate 10 Million rows/s

Realtime ingestion

- How to avoid data loss
 - Back pressure
 - Connection auto rebuild
 - Write backup data to backup kafka
 - Alarm monitor
 - ...

Realtime ingestion

Write to single shard table or distributed table ?

Single Shard

Consider about

- I. Number of Sinkers processors
- II. Number of Connections
- III. Number of Tables
- IV. Number of shards

- I. Good control of read or write split
- II. Good control data transfer

Query

TSDB styles Grafana plugin

▼ A	Metric	video.video_no_picture_ratio	Downsample	\$downsample ⓘ	TimeShift	shift time ⓘ	Query	Show Help ▶									
Fields ⓘ																	
sum(value)/sum(_cnt) AS `卡比`			✎ ✕ ↓														
line AS `线路`			✎ ✕ ↑														
Field		v	Alias	field_alias		add field ✕											
Filters ⓘ	anchored	version	line in (\$line) and platform = 'adr'														
Group by ⓘ	line ✎	value															
Having ⓘ	line ✎	_prov															
sum(_cnt)>\$threshold																	
DimsDesc ⓘ	anchoruid	line	coderate	h265	p2p	time	mini	platform	version	success	retcode	experiment	consistenhashinfo	_isp	_prov	its	day
FieldsDesc ⓘ	value																

Query

- Rewrite Query
 - SQL Parser
 - Custom function support
 - Custom query statement

Query

- Rewrite Query

```
SELECT  intDiv(its, 20) * 20 AS _timestamp, sum(value) / sum(_cnt) as v, dict(device_id) as device_name
FROM metric_name WHERE (its >= 1540178823) AND (its <= 1540279123) AND platform = * and line in (1,2,3)
GROUP BY line, _timestamp
```

```
SQLParser =>      time granularity : 20 seconds
                  time start to end: [1540178823, 1540279123]
                  replace `*` filter: platform = *
                  matched_columns : [its, value, _cnt, device_id, platform, line]
```

```
SELECT  intDiv(its, 20) * 20 AS _timestamp, sum(value) / sum(_cnt) as v, dictGetString('device', 'name', toUInt64(device_id)) as device_name
FROM real_table WHERE (its >= 1540178823) AND (its <= 1540279123) and line in (1,2,3)
GROUP BY line, _timestamp
```

Query

- Custom query statement

```
ck_tags(metric_name, sVersion, now()-300, platform in ($platfom) ,100)
```

Parser =>

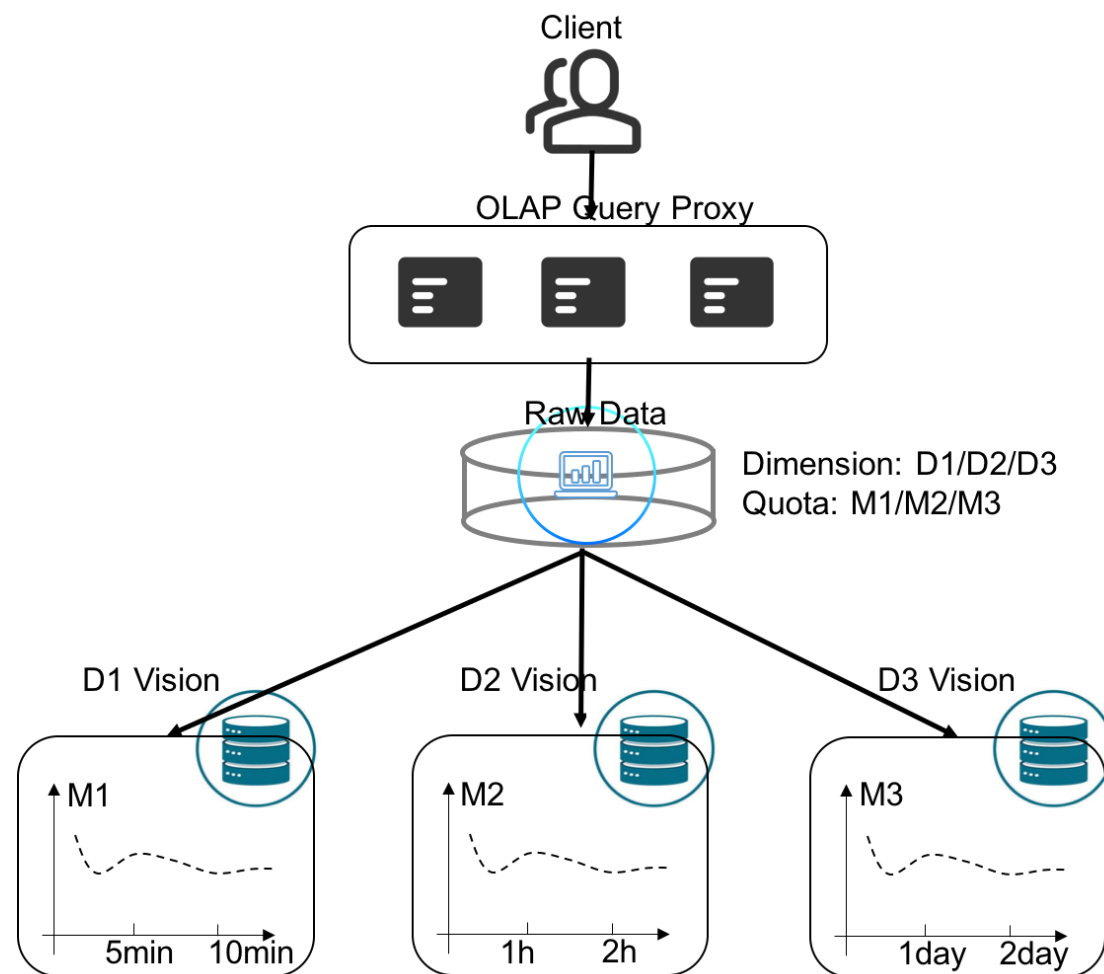
```
select sVersion from (select sVersion, count() as c  
  from real_table where its >= now()-300 AND platform in ('android', 'ios') group by sVersion order by c limit 100)
```

Query

- Query rewrite to view

Matching Rules:

```
time granularity : 20 seconds  
time start to end: [1540178823, 1540279123]  
columns : [its, value, _cnt, device_id, platform, line]  
...
```



Data Monitor

Flink CEP ?
Flink SQL ?



ClickHouse

查询

英文名:

v_metric_app_monitor

中文名:

视频应用资源查询

目标ck集群:

id:1 name:production

查询SQL:

select appname,ip,max(connections_num) as connections_num
from appname='vhuya-transcode' and day = today() AND
group by appname,ip having connections_num >500

系统会自动生成 view 语句，只写 query 即可。校验

结果 view:

v_metric_app_monitor

按 "v_业务名称" 或 "v_业务名称_窗口时间" 命名，如 "v_video_1h"

执行查询间隔:

20000

单位 ms, 1 second = 1000 millionseconds

Source Table => SQL => View

Interval Query => Rule Parser => Alarm

appname	ip	connections_num
vhuya-transcode	58.192.168.100	6
vhuya-transcode	58.192.168.101	6
vhuya-transcode	58.192.168.102	6
vhuya-transcode	58.192.168.103	6
vhuya-transcode	58.192.168.104	6
vhuya-transcode	58.192.168.105	6
vhuya-transcode	221.192.168.106	6
vhuya-transcode	58.192.168.107	6
vhuya-transcode	58.192.168.108	10
vhuya-transcode	58.192.168.109	6
vhuya-transcode	58.192.168.110	6

```
[
  {"appname":"vhuya-transcode","ip":"...", "connections_num":6},
  ...
]
```

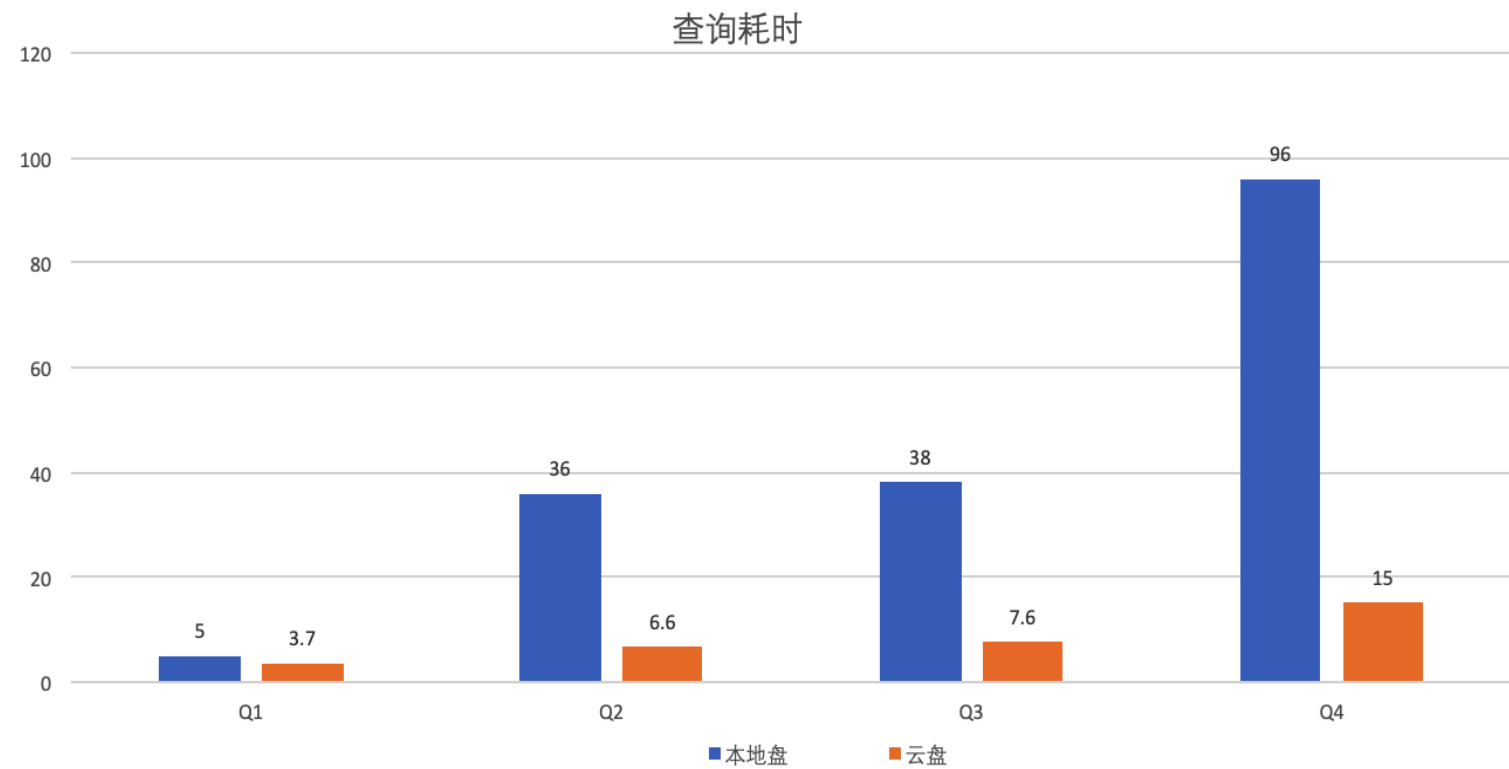
ckview="v_metric_app_monitor" AND connections_num > 500

Cloud

- IO

- Sequence Read
- Sequence Write

IOPS Cloud >> local



More about cloud

- Distributed file systems (Considering)
 - Moosefs
 - Aws efs(Pinterest Goku)
 - Aliyun nas
- Data store separation (Working on it)
 - 90% query data is about last month
 - SSD for hot data, SATA for history data.

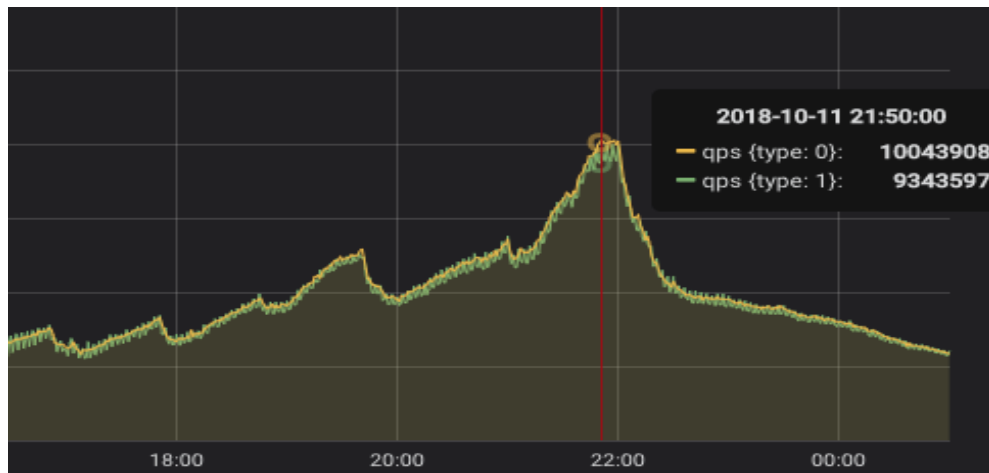
Operations

- Deploy platform
- Ansible
- Golang Template

```
<logs>
  {{range $index, $shards := .In.shards}}
    <shard>
      <weight>1</weight>
      <internal_replication>false</internal_replication>
      {{range $index2, $shard := $shards}}
        <replica>
          <host>{{$shard.ip}}</host>
          <port>{{$shard.port}}</port>
          <user>{{$shard.user}}</user>
          <password>{{$shard.pwd}}</password>
        </replica>
      {{end}}
    </shard>
  {{end}}
</logs>
```

Performance

- Data insert up to 10 Million rows/second, 300 Billion rows per day.
- 18 trillion total rows
- Query average costs < 1s (yet still contains little long queries)



```
SELECT sum(rows)
FROM cluster('logs', 'system', 'parts')
WHERE active = 1
```

```
sum(rows)
18345366782968
```

1 rows in set. Elapsed: 1.610 sec. Processed 535.27 thousand rows, 1

To community

- Tools

- JDBC <https://github.com/housepower/ClickHouse-Native-JDBC>
 - Clickhouse_sinker https://github.com/housepower/clickhouse_sinker

- Features

- windowFunnel
 - retention

Thank You

Hiring, we want you!

大数据，算法
监控 质量分析平台开发工程师
欢迎各路人才加入

