TalkingData原子立方体借力Druid加速海量数据的统计分析

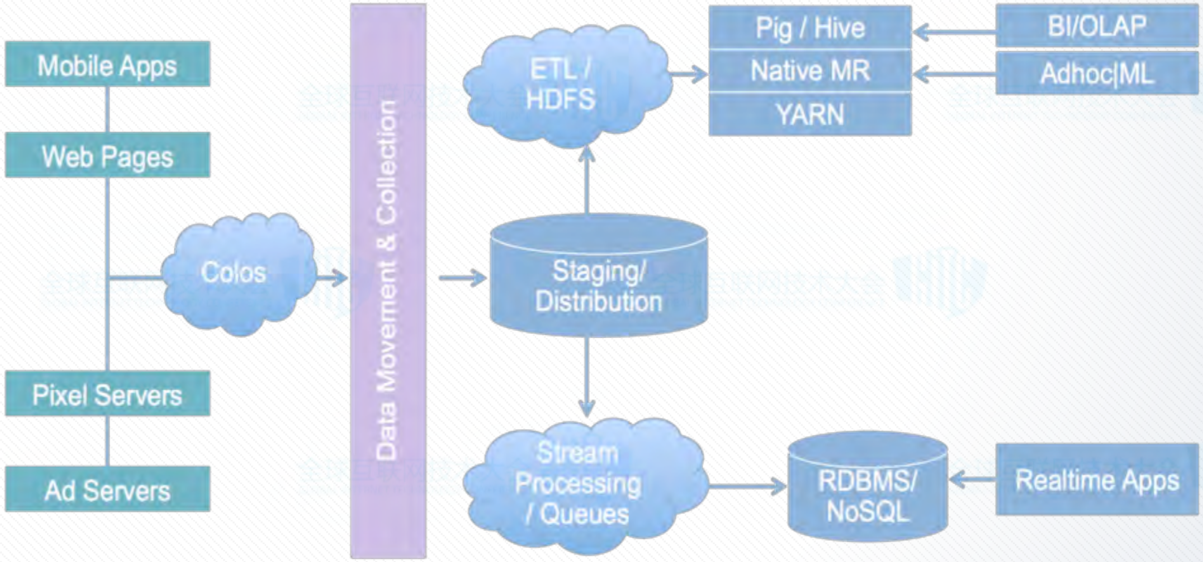
TalkingData
移动 · 数据 · 价值

▶ TalkingData’ s Massive Data statistical analysis business

- 支持20种主流移动平台
- 40万款智能移动系统
- 每天2.5亿活跃智能设备
- 每天处理34亿会话，400亿事件
- 每天14TB数据流分发
- 支持实时统计分析与查询，部分指标需要精准统计

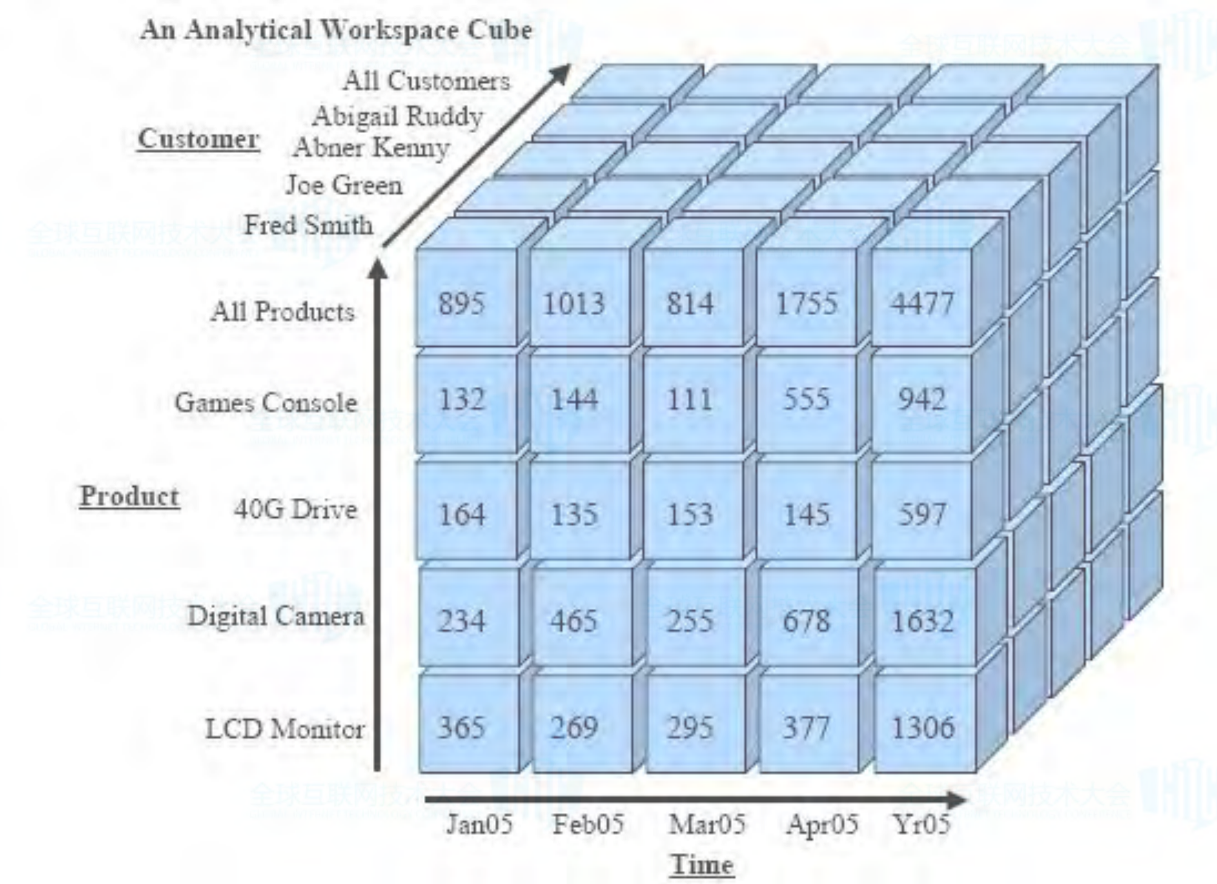
传统技术体系和缺陷

- 大数据量查询速度慢
- 大集群构建成本高昂
- 多维交叉计算能力低效
- 流式计算无法回朔



TD Atom Cube Definition

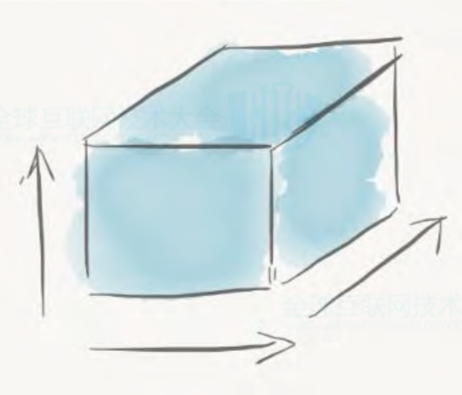
OLAP-Cube: multi-dimensional array of data

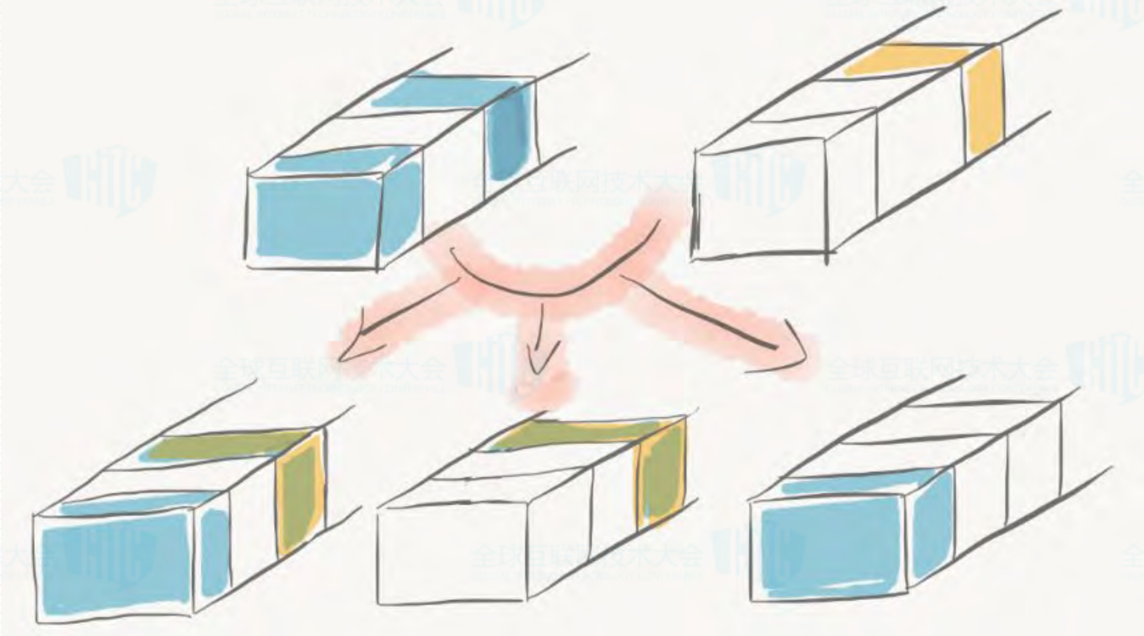


An Analytical Workspace Cube

Product	Jan05	Feb05	Mar05	Apr05	Yr05
All Products	895	1013	814	1755	4477
Games Console	132	144	111	555	942
40G Drive	164	135	153	145	597
Digital Camera	234	465	255	678	1632
LCD Monitor	365	269	295	377	1306

Atom OLAP Cube:





Calculation：

	∪	∩	⊕
交换	$A\cup B=B\cup A$	$A\cap B=B\cap A$	$A\oplus B=B\oplus A$
结合	$(A\cup B)\cup C=$ $A\cup(B\cup C)$	$(A\cap B)\cap C=$ $A\cap(B\cap C)$	$(A\oplus B)\oplus C=$ $A\oplus(B\oplus C)$
幂等	$A\cup A=A$	$A\cap A=A$	

TD atom cube

Fact table：

Time	Deviceid	Province	Mobile	App	Event
2015-12-22 1:00	036ca36f9c971906a97e2321ae0aeff8a	北京	Nexus S	全面枪战	充值
2015-12-22 1:30	06bd68dc66029f975c86d30e3e296d658	北京	Nexus S	全面枪战	充值
2015-12-22 2:00	02e2b5bac7ec1f9f993d48484b9fbf333	天津	iPhone 5s	滴滴打车	支付

id	Deviceid
0	036ca36f9c971906a97e2321ae0aeff8a
1	06bd68dc66029f975c86d30e3e296d658
2	02e2b5bac7ec1f9f993d48484b9fbf333

Time	id	Province	Mobile	App	Event
2015-12-22 1:00	0	北京	Nexus S	全面枪战	充值
2015-12-22 1:30	1	北京	Nexus S	全面枪战	充值
2015-12-22 2:00	2	天津	iPhone 5s	滴滴打车	支付

TD atom cube 原理

Fact table：

id	Deviceid
0	036ca36f9c971906a97e2321ae0aeff8a
1	06bd68dc66029f975c86d30e3e296d658
2	02e2b5bac7ec1f9f993d48484b9fbf333

Time	id	Province	Mobile	App	Event
2015-12-22 1:00	0	北京	Nexus S	全面枪战	充值
2015-12-22 1:30	1	北京	iPhone 5s	全面枪战	充值
2015-12-22 2:00	2	天津	Nexus S	滴滴打车	支付

TD atom cube：

Time	App	Bitmap
2015-12-22	全面枪战	0、1
2015-12-22	滴滴打车	2

Time	Dimension	Metric
Time	Provice	Bitmap
2015-12-22	北京	0、1
2015-12-22	天津	2

Time	Mobile	Bitmap
2015-12-22	Nexus S	0、2
2015-12-22	iPhone 5s	1

Time	Event	Bitmap
2015-12-22	充值	0、1
2015-12-22	支付	2

TD atom cube

TD atom cube：

Time	App	Bitmap
2015-12-22	全面枪战	0、1（bitmap1）
2015-12-22	滴滴打车	2（bitmap2）

Time	Province	Bitmap
2015-12-22	北京	0、1（bitmap5）
2015-12-22	天津	2（bitmap6）

Time	Mobile	Bitmap
2015-12-22	Nexus S	0、2（bitmap3）
2015-12-22	iPhone 5s	1（bitmap4）

Time	Event	Bitmap
2015-12-22	充值	0、1（bitmap7）
2015-12-22	支付	2（bitmap8）

基数计算：

SELECT Distinct(Device) Where App =全面枪战 and province=北京 and time= 2015-12-22

运算转变成：Bitmap1 and bitmap5

优势：

1. 存储减少
2. 计算快
3. 支持join(这个能够很好的解决留存类型的分析)

TD Atom Cube Implementation

TD atom cube：

Bitmap+ Concise=ConciseSet

ConciseSet ：https://github.com/metamx/extendedset

RoaringBitmap

TD atom cube 面临的问题

1. Storage

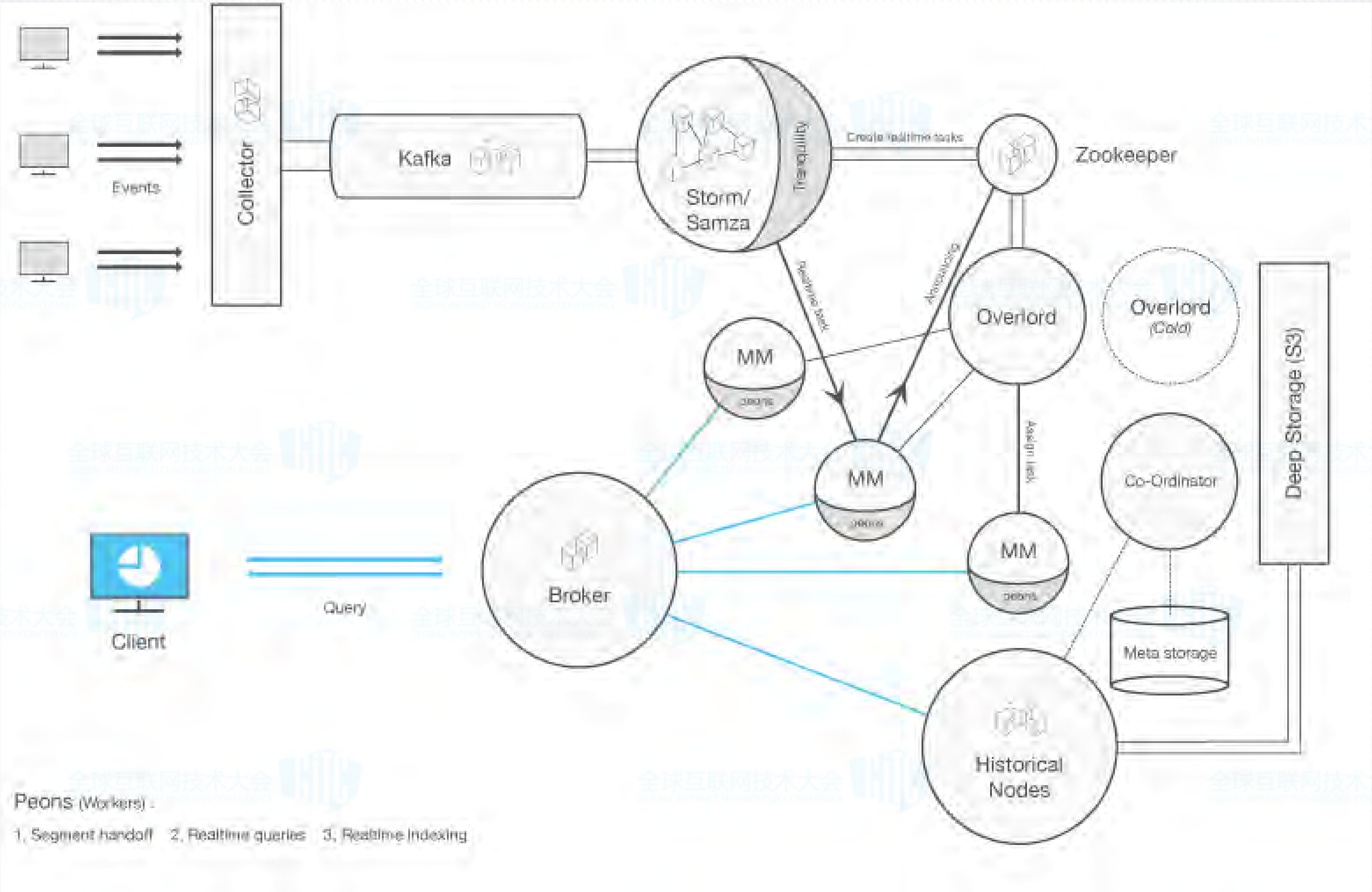
   Central Storage, IO exhaust.

2. Computation

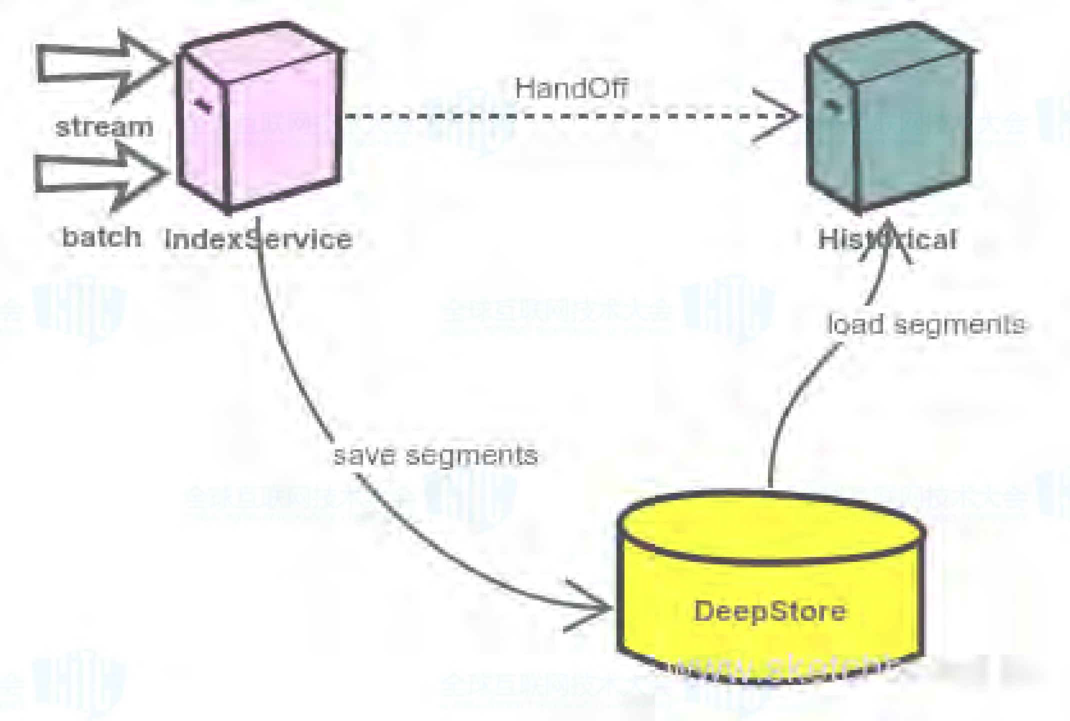
   OOM

druid

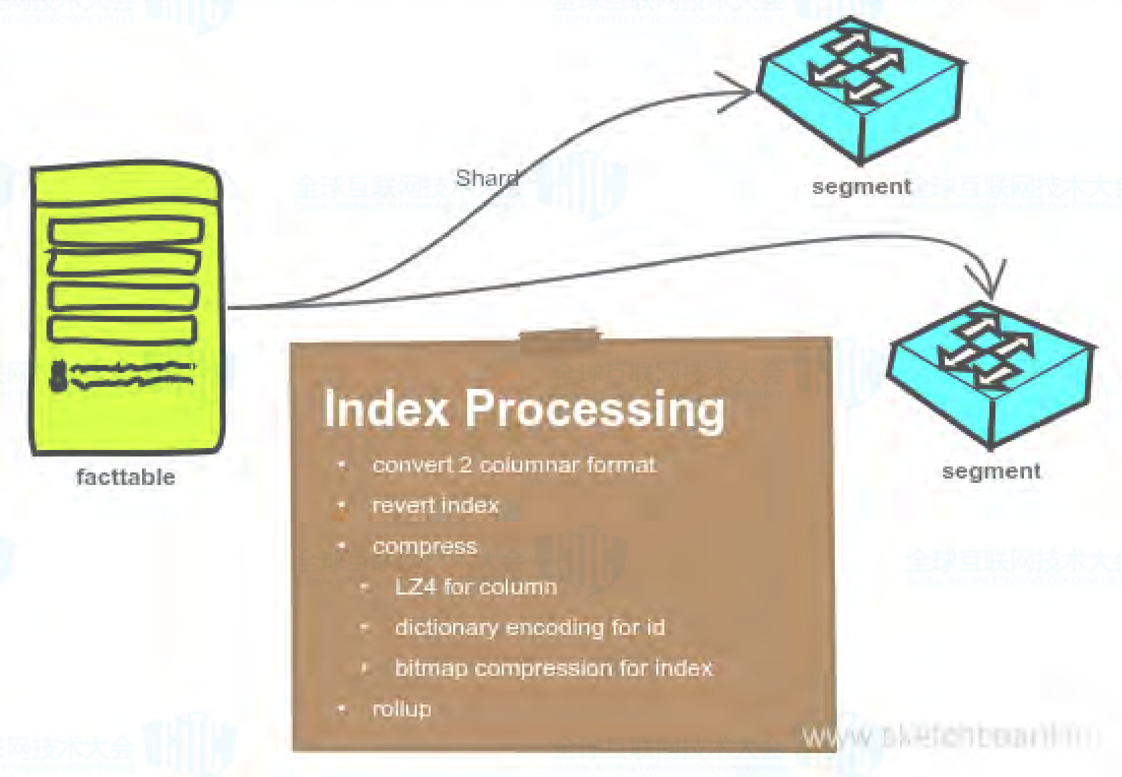
Druid 简介


Events
Collector
Kafka
Storm/
Samza
Tranquility
Create realtime tasks
Zookeeper
Realtime task
Announcing
Overlord
Overlord
(Cold)
MM
peons
MM
peons
MM
peons
Assign task
Deep Storage (S3)
Co-Ordinator
Meta storage
Client
Query
Broker
Historical
Nodes
Peons (Workers) :
1. Segment handoff
2. Realtime queries
3. Realtime indexing

Stream/Batch ingestion



Indexing

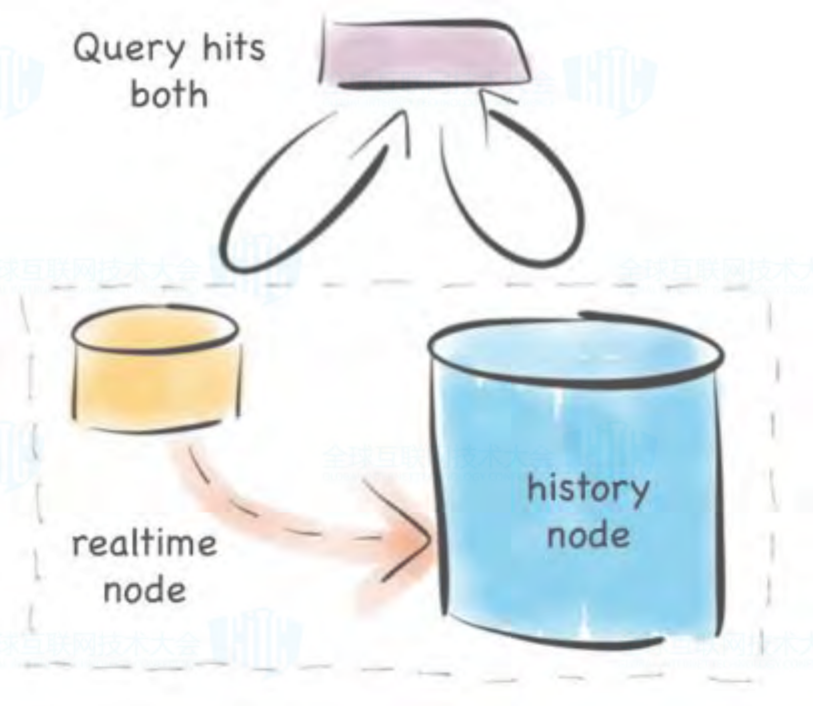


Index Processing

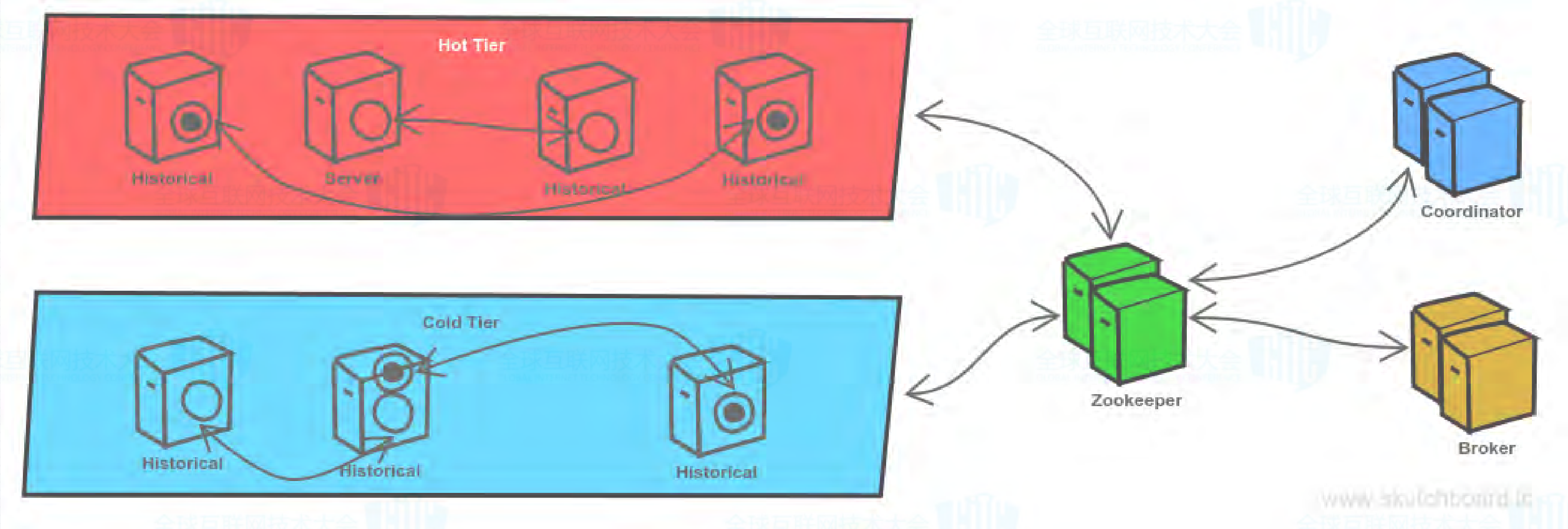
- convert 2 columnar format
- revert index
- compress
  - LZ4 for column
  - dictionary encoding for id
  - bitmap compression for index
- rollup

Query

- Group by
- Top N
- Timeseries
- Search
- Time boundary
- Metadata query



High Availability and Load Balance



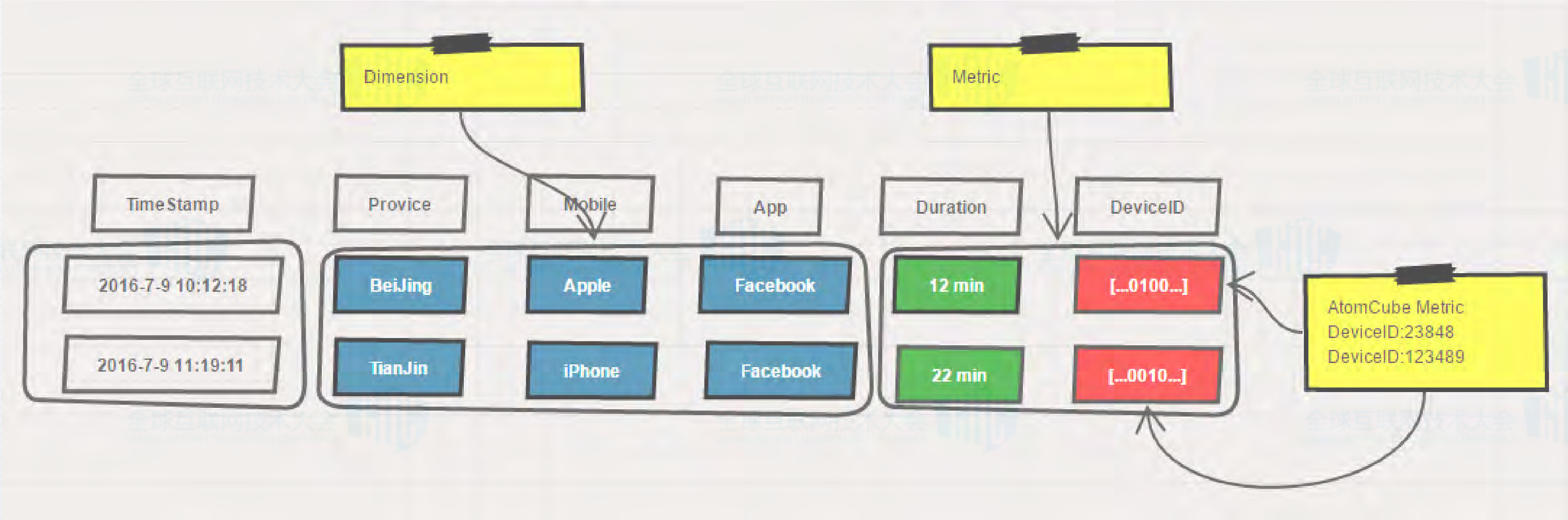
Druid's Limitation we concerned:

- Exactly cardinality calculate
  - Estimation approach by HyperUnique/Cardinality/Sketch estimat
- Join
  - limited support through “lookups”, replace one dimension value with another value

TD Atom Cube Integrate with Druid

Idea :

Makes an unique id as metric, saved in Atomcube(bitmap) for each row.



Dimension

Metric

TimeStamp	Provice	Mobile	App	Duration	DeviceID
2016-7-9 10:12:18	BeiJing	Apple	Facebook	12 min	[...0100...]
2016-7-9 11:19:11	TianJin	iPhone	Facebook	22 min	[...0010...]

AtomCube Metric
DeviceID:23848
DeviceID:123489

Main Points:

aggregation:

the atomcube naturally support.(UNION), so this feature can be applied in Rollup, QueryAggregation, and postAggregation processes.

Exactly cardinality:

the size of the atomcube after aggregation.

Join:

The atomcube in query result can do intersect if they have same means.

More:

Can do union and not operation

The Benefits:

- The huge bitmap stored separately and loaded in clustered nodes(Druid's historical).
- The computation can be accomplished in distributed environment (rollup, aggregation and historical) .
- Avoid Data skewed(Druid's load balance)

Nonintrusive

- Did not touch any existence Druid code.
- Just added an extension - druid-atomcube-0.9.0.jar
- Install
  - Put the druid-atomcube-0.9.0.jar under /druid-0.9.0/extensions/druid-atom-cube
  - add druid.extensions.loadList=["druid-atom-cube"] in common.runtime.properties file
  - startup all nodes

Implementation glance

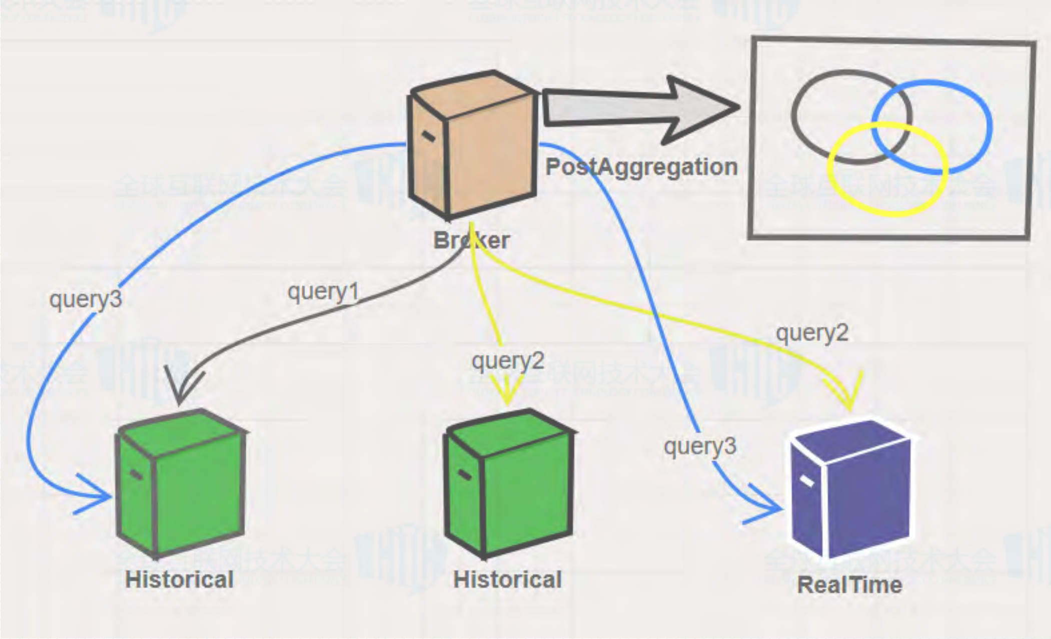
- new DruidModule
- Aggregator and AggregatorFactory
  deserialize, metric, aggregate
- Query
  - Defined new query url:/druid/v2/atomcube
  - Defined new query structure.
  - Parallel running multiple queries on difference tables.

Schema Definition: define atomcube metric

```
"dataSchema":{
  "dataSource":"wikiticker",
  "granularitySpec":{
    ......
  },
  "parser":{
    ......
  },
  "metricsSpec":[
    {
      "name":"count",
      "type":"count"
    },
    {
      "name":"user_unique",
      "type":"hyperUnique",
      "fieldName":"hyperUser"
    },
    {
      "name":"user_atomcube",
      "type":"atomCube",
      "fieldName":"uuid"
    }
  ]
}
```

Query: new structure

```
{
 "queryType": "atomCube",
 …
 "queries":  {
   "query1":{…}, // each query must include atomcube aggregation
   "query2":{…},
   "query3":{…},
 },
"postAggregations":[
 {
   "type": "atomCubeSet",
   "name": "test_set",
   "func": "INTERSECT",
   "fields": ["query1","query2","query3"]
 },
 {
   "type": "atomCubeSize",
   "name": "test_size",
   "field": "test_set"
 },
 {
   "type": "atomCubeRaw",
   "name": "test_raw",
   "format": "LIST",
   "field": "test_set"
 } ]
```



Query Result:

```
[ {
  "test_raw" : [ 177, 1411, 2086, 2580, 3237, 3708, 3855, 6031, 6868, 8167,
8668, 9647, 10138 ],
  "test_size" : 13,
  "test_set" :
"OjAAAAEAAAAAAAwAEAAAALEAgwUmCBQKpQx8Dg8PjxfUGucf3CGvJZ
on"
} ]
```

TalkingData

SQL-Like Query Engine

Based on Calcite Druid Adapter

Refactor the Rules and QueryNode to Support Atomcube Query with standard SQL, like:

sql1 = "select distinct "uuid" from "wiki2" where "namespace “ = 'Wikipedia'";

sql2 = "select distinct "uuid" from "wiki1" where "countryName" = 'France'";

sql = "select distinct count(*) from (" + sql1 + " union " + sql2 + ")";

TalkingData

THANKS!

官网 / www.talkingdata.net　　微信 / TalkingData

微博 / @TalkingData　　服务支持 / support@tendcloud.com