

—— World Of Tech 2017 ——

全球架构与运维技术峰会

2017年4月14日-15日 北京富力万丽酒店

ARCHITECTURE



出品人及主持人：

李鹏涛

京东商城
青龙研发部高级总监

大数据应用创新

云客服实时分析架构演进

从NOSQL到时序数据库



千慕

阿里巴巴
资深研发工程师

分享主题：

客服SAAS实时分析架构演进
-从NoSQL到时序数据库

关于我

李灼灵（花名：千慕）

阿里巴巴集团-商家事业部-基础技术

2014 基于Docker的PAAS云平台 (6000+ 容器)

2015 AliAPM (apm.aliyun.com , 1200+活跃应用 , 类似听云)

2016 做垂直业务的商业变现（例如：客服SAAS、千牛付费问答）

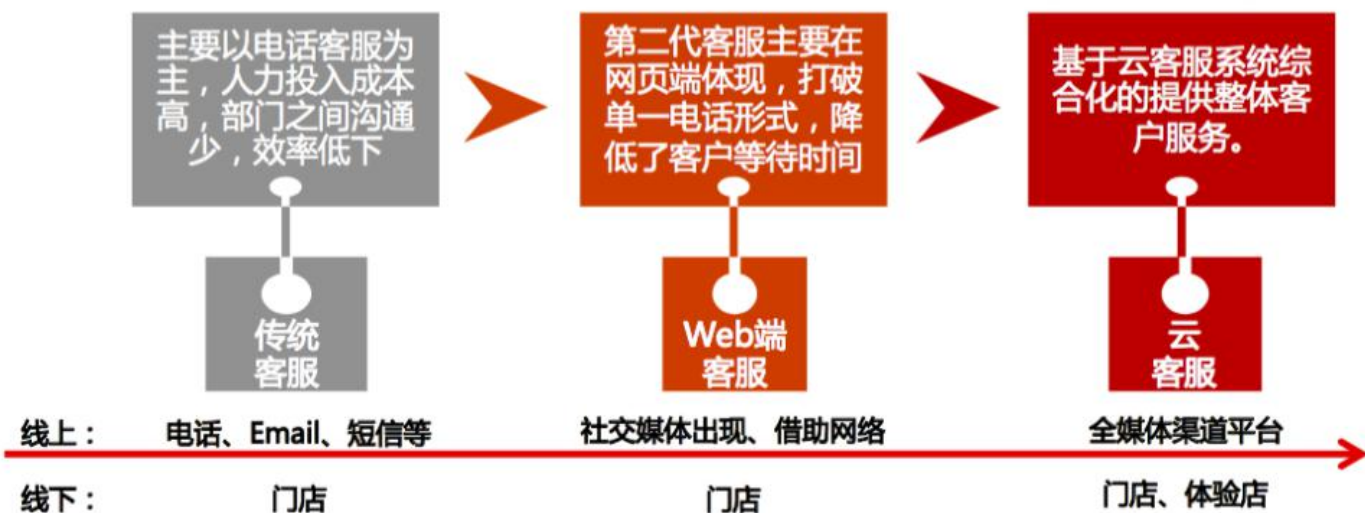
邮箱：zhuoling.lzl@alibaba-inc.com

目录

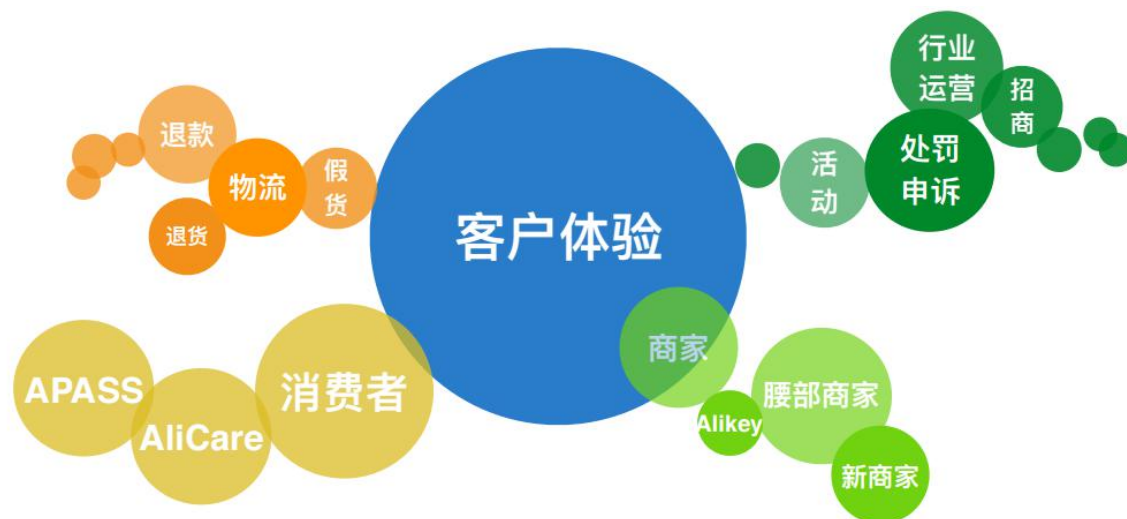
- 01 客服行业背景
- 02 云客服架构
- 03 实时分析应用\架构\常见问题
- 04 架构演进-从NOSQL到时序数据库

客服行业背景

国内客服体系发展历程

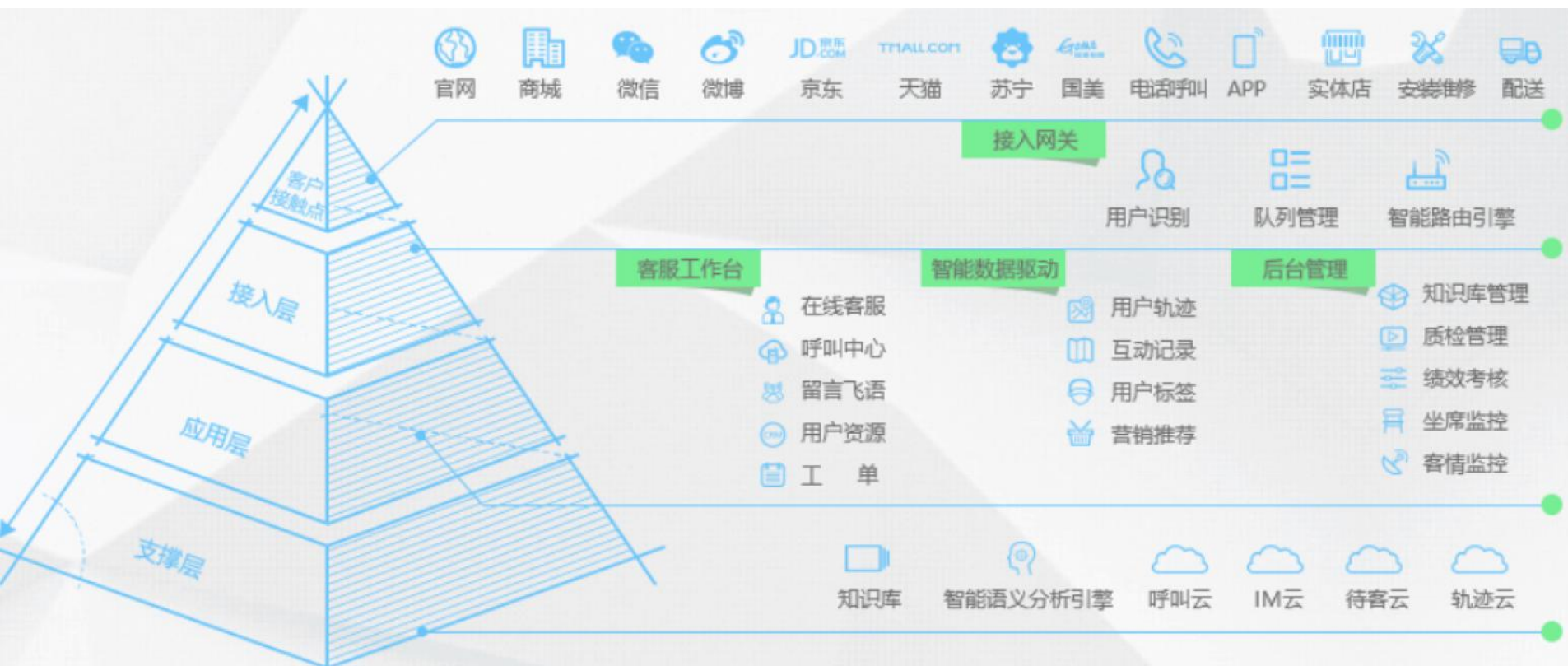


客服职能转型



云客服架构

云客服主要架构



渠道整合 + 数据驱动

云客服实时分析场景

花呗

十二期

免息

出现次数
2216

花呗分十二期免息吗

不是十二期免息吗怎么还收费

请问，十二期免息是怎样的呢

查看更多

热点问题分析

售前客服01组

繁忙

+ 添加人员

2,821

组接待人数

12

人均接待

230 / 5

在线 / 挂起

账号名	接待数	响应	转化率	客件数	操作
森a57	21	24s	37%	2.7	挂起 换组
森a136	4	57s	41%	2.1	挂起 换组
森a112	10	21s	35%	1.1	挂起 换组
森a1	11	25s	33%	2.6	挂起 换组

实时接待

安暖 售前

检测消息数：123,234

命中消息数：336

命中规则数：3

开始语 是否主动及时招呼客户 2

结束语 是否与顾客礼貌道别 3

文字运用和耐心度 是否礼貌用户 23

说明力 是否了解用户偏好 3

说明力 是否提炼商品卖点 3

说明力 是否准确响应顾客需求打消顾客疑虑 7

销售技巧 是否有针对性地关联销售 22

催拍付 卡单意识 是否及时进行催拍 3

业务遵守 是否按照公司SOP流程操作 22

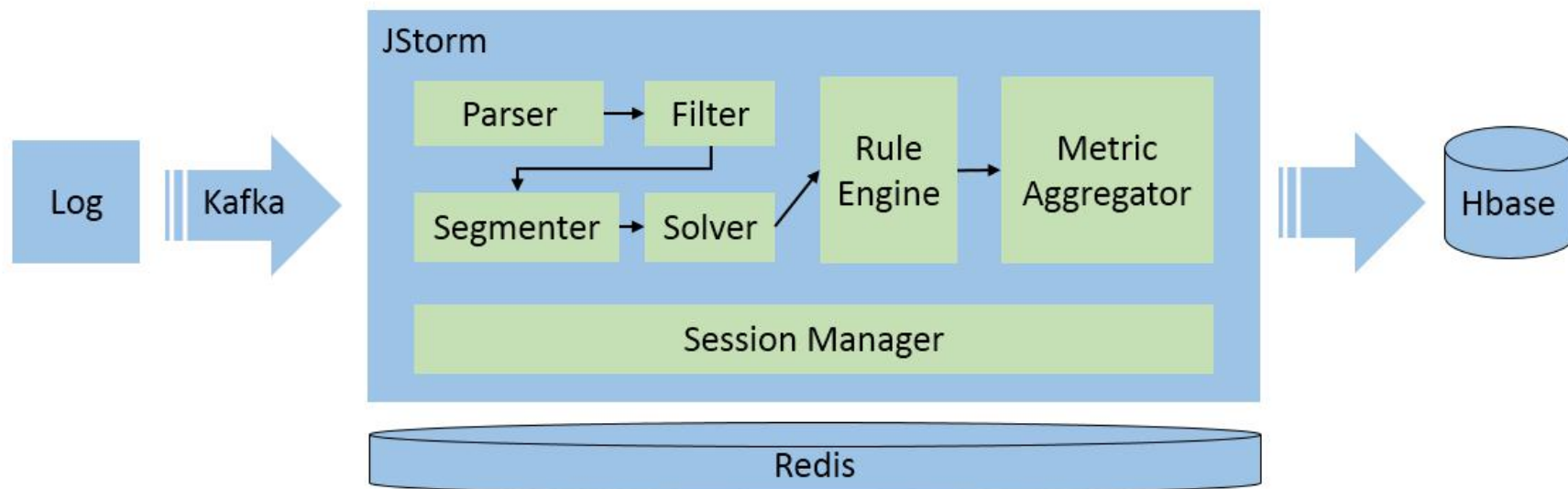
催拍付 卡单意识 是否及时进行催付 3

查看详情

实时质检

实时分析架构

实时分析架构



数据采集

聊天
浏览轨迹
交互记录
...

数据通道

MQ: Kafaka
API

实时计算引擎（Storm、Spark）

Segmenter: 分词器
Solver: 解答器，基于知识图谱
Rule Engine: 是否合规、达到话术要求等
Metric aggregator: 聚合器

存储层

Nosql: Hbase
Cache: Redis

数据存储结构

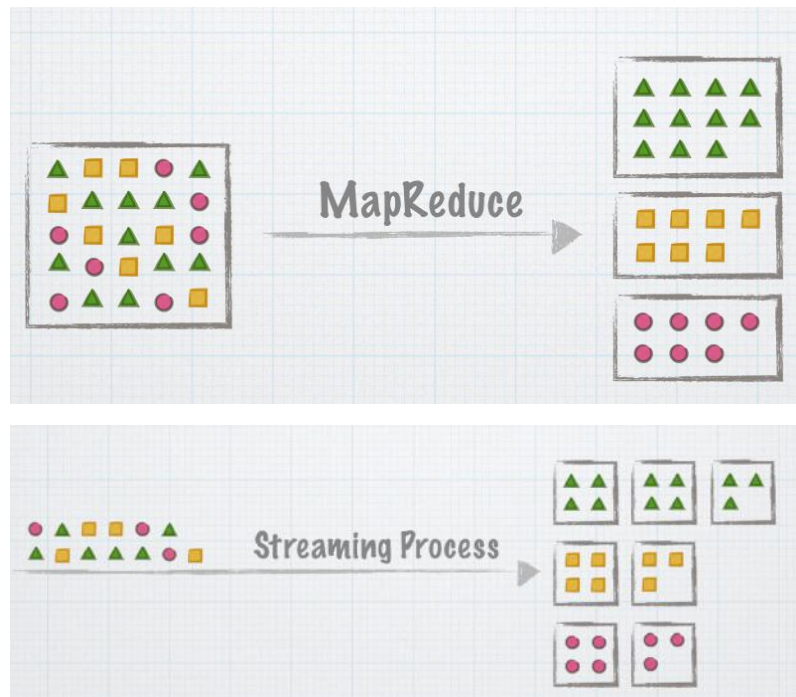
Rowkey	Value
聊天指标,2017-3-12 19:03,买家:张三,卖家:李四, 指标:响应时间	10000

Rowkey = metric name + timestamp + tags的组合。

实时聚合常见问题

数据聚合问题1 – 数据倾斜(热点)问题

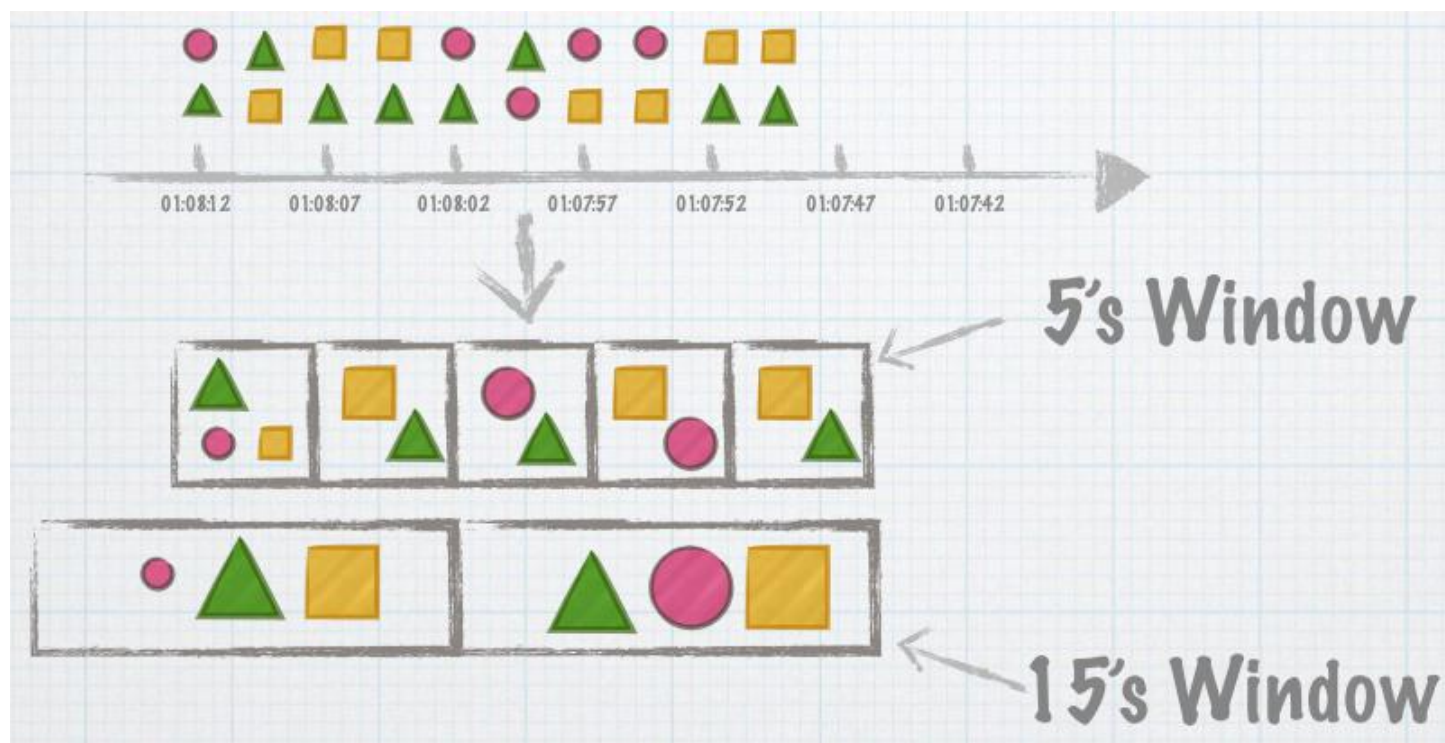
- 导致堆积、OOM、实时性下降
- 尽可能细粒度hash
- 特殊key处理
- 二段Merge
- ...



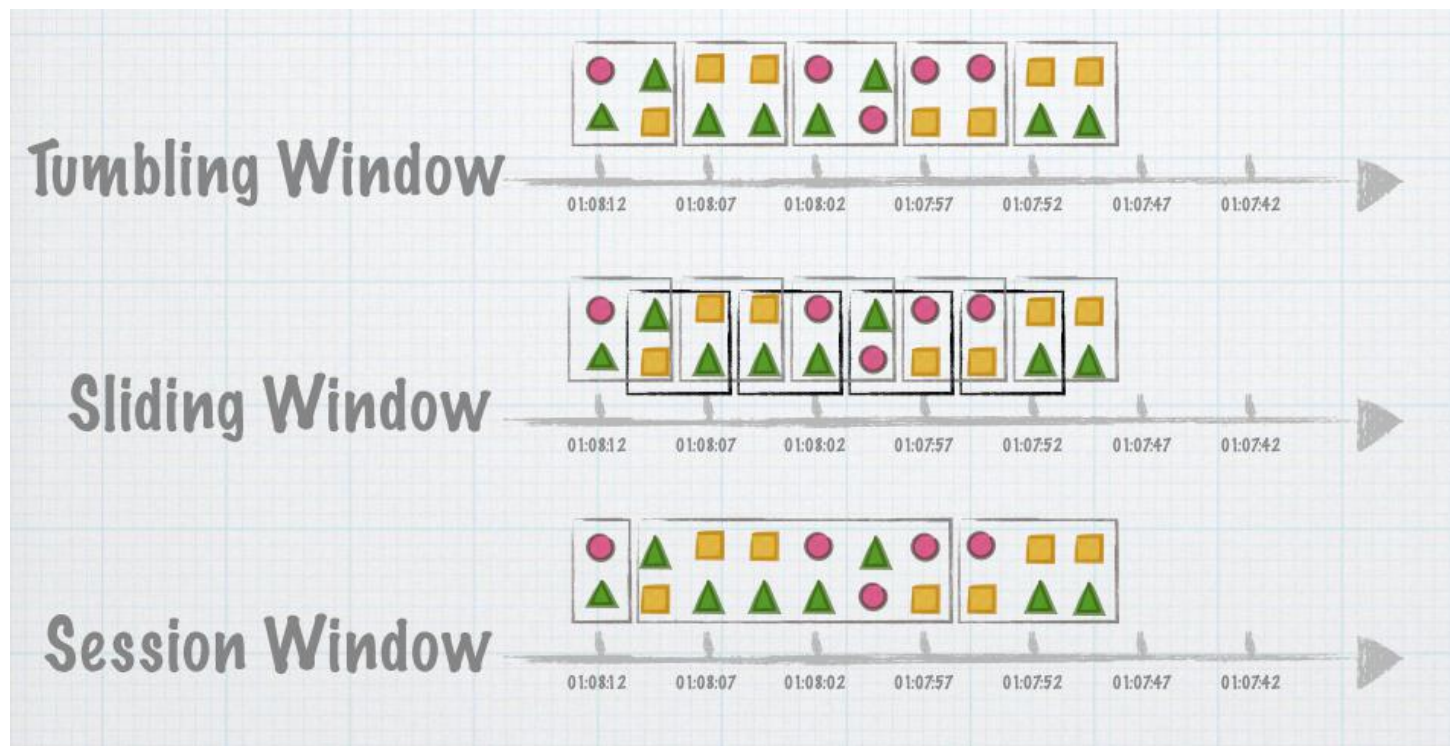
绿色△多，导致数据倾斜

数据聚合问题2 – 窗口切分 (Windowing)

窗口切分(windowing): 把流式的数据按精度要求 (5s, 15s等) 进行划分。



常见计算窗口



1. 难点

- 数据到达乱序
- 窗口切分的判断
- 大量数据缓存

2. 处理难度: Tumbling(固定) < Sliding(滑动) < Session (会话)

数据聚合问题3 – 海量时间线 (Series)

时间线(Series): 由多个维度值确定的最细粒度的聚合项。



三种图案形成三条时间线

数据聚合问题3 – 海量时间线 (Series)

当维度变多时，时间线个数会指数上涨！

随之，内存占用升高，存储成本升高，查询速度下降。

Rowkey	Value
聊天指标,2017-3-12 19:03,买家:张三,卖家:李四, 指标:响应时间	10000

时间线举例

例如：

买家：10w

卖家：1w

指标：10个

时间线个数： $10w * 1w * 10 = 100 \text{ 亿} !$

总结&思考

技术复杂

- 数据聚合问题1 – 数据倾斜(热点)问题
- 数据聚合问题2 – 窗口切分 (Windowing)
- 数据聚合问题3 – 海量时间线 (Series)

成本高

- 存储成本一年几百万
- 实时质检80万通电话，要200万的硬件成本

技术复杂，成本高，是否有优化方案？

时序数据库(TSDB)是什么

TSDB：专门存储按时间顺序变化（即时间序列化）的数据，支持原始数据查询和实时聚合，支持数据压缩，适合海量数据处理。

- InfluxDB

短小精悍，社区很活跃。但集群方案收费，适合小规模用户。

- OpenTSDB



基于Hbase的成熟的TSDB方案，被很多大公司使用。

- KairosDB --- Google

- Gorilla --- Facebook

- Informix --- IBM

...

Rank			DBMS	Database Model	Score		
May 2016	Apr 2016	May 2015			May 2016	Apr 2016	May 2015
1.	1.	1.	InfluxDB	Time Series DBMS	4.18	+0.23	+2.65
2.	2.		RRDtool	Time Series DBMS	2.58	+0.05	
3.	3.		Graphite	Time Series DBMS	1.64	+0.08	
4.	4.		OpenTSDB	Time Series DBMS	1.43	+0.02	
5.	5.	↓ 2.	Kdb+ 	Multi-model 	1.22	+0.02	+0.35
6.	6.		Druid	Time Series DBMS	0.27	+0.04	
7.	7.		KairosDB	Time Series DBMS	0.23	+0.04	
8.	8.		Prometheus	Time Series DBMS	0.14	-0.01	
9.	↑ 10.		Riak TS	Time Series DBMS	0.10	+0.08	
10.	↓ 9.		Axibase	Time Series DBMS	0.08	-0.04	

方案对比

基于NOSQL预计算方案	基于时序数据库（TSDB）的实时聚合方案
需自主实现聚合逻辑	查询时聚合，不再需要考虑时间窗口问题
查询维度固定、计算方法固定，扩展困难	支持任意维度、时间段，多种聚合计算
空间换时间，存储成本高	存储优化明显
受时间窗口限制，往往只支持到分钟级聚合	支持秒级聚合，更真实还原某些场景
有状态，要有checkpoint，故障恢复、发布的处理逻辑复杂	只需记录MQ的offset
查询速度快	因为查询时聚合，查询速度较慢



OpenTSDB存储优化原理-优化前

Rowkey = metric name + timestamp + tags的组合。

数据重复明显

rowkey	value
sys.cpu.user,1465920000{cpu=0,host= 10.101.168.111}	50
sys.cpu.user,1465920000{cpu=0,host= 10.101.168.112}	50
sys.cpu.user,1465920000{cpu=1,host= 10.101.168.112}	50
sys.cpu.user,1465920001{cpu=0,host= 10.101.168.111}	50
sys.cpu.user,1465920001{cpu=0,host= 10.101.168.112}	50
sys.cpu.user,1465920001{cpu=1,host= 10.101.168.112}	50

图片来自@木洛的分享

OpenTSDB存储优化原理

- 为每个metric、tag key和tag value都分配一个UID，缩短row key。
- 将同一小时的数据存到不同的列中，减少key-value数。
- 使用偏移量时间戳，进一步减少列名占用空间。

UID映射

metric	UID
sys.cpu.user	{0,0,1}

tagkey	UID
cpu	{0,0,1}
host	{0,0,2}

tagvalue	UID
0	{0,0,1}
1	{0,0,2}
10.101.168.111	{0,0,3}
10.101.168.112	{0,0,4}

数据表

rowkey	+0	+1	+2	+3
{0,0,1},1465920000[{0,0,1}={0,0,1},{0,0,2}={0,0,3}]	50	50	50	50
{0,0,1},1465920000[{0,0,1}={0,0,1},{0,0,2}={0,0,4}]	50	50	50	50
{0,0,1},1465920000[{0,0,1}={0,0,2},{0,0,2}={0,0,4}]	50	50	50	50

图片来自@木洛的分享

OpenTSDB查询优化经验

使用端优化



- 合理拆分Metric，减少时间线个数
- 注意Tag的顺序
- 并发查优化



服务器优化

- 预集合
- 结果缓存
- 降精度



Thank you !

数据采集问题

1. 低消耗 实时

- 批量发送
- 日志不落地
- 客户端合并
- 客户端过滤

2. 使用MQ的正确姿势

- 消息体大小
- 批量还是单条消费
- 消费轨迹
- 消费重置



高可用问题

1. 故障恢复
 - 记录MQ的消费偏移量，恢复时重跑数据
 - Check Point机制
2. 节奏控制
 - 多粒度的流控
 - 反压机制 (Backpressure)
3. 外部接口强依赖
 - 元信息本地缓存 + 分布式缓存
 - 合理的重试 + 备用方案



TSDB市场

19 systems in ranking, November 2016

Rank			DBMS	Database Model	Score		
Nov 2016	Oct 2016	Nov 2015			Nov 2016	Oct 2016	Nov 2015
1.	1.	↑ 2.	InfluxDB	Time Series DBMS	5.60	+0.28	+2.83
2.	2.	↓ 1.	RRDtool	Time Series DBMS	2.47	-0.01	-0.84
3.	3.	3.	Graphite	Time Series DBMS	1.91	+0.01	+0.42
4.	4.	↑ 5.	OpenTSDB	Time Series DBMS	1.45	-0.02	+0.09
5.	5.	↓ 4.	Kdb+	Multi-model	1.17	-0.04	-0.19
6.	6.	↑ 8.	Druid	Time Series DBMS	0.63	+0.03	+0.50
7.	7.	7.	Prometheus	Time Series DBMS	0.31	+0.03	+0.18
8.	8.	↓ 6.	KairosDB	Time Series DBMS	0.27	+0.00	+0.13

InfluxDB: 集群方案收费, 适合小规模用户。

RRDtool: 老牌的文件型时间序列数据库, 适合嵌入场合。

Graphite: 从采集到展示的全套方案, 聚焦在功能, 开箱即用。

OpenTSDB: 成熟的TSDB实现, 被很多大公司采用, 大部分基于HBase的时间序列应用都有跟其类似的设计。

Druid: 一个基于时间的OLAP列存数据库, 长处在于AD-Hoc的聚合/分析。

Prometheus: go编写, 和go环境结合紧密, 随着go语言而流行。

HITSDB: 最基础的算法是无状态的, 可以很方便的以集群方式部署。

HITSDB: 高性能的分布式时间序列数据库, 适合搭建服务。

HITSDB: 专注于存储和计算, 聚焦在性能。

HITSDB: 兼容OpenTSDB协议, 提供双存储引擎针对不同的应用场景, 增加了很多性能调优和集群灾备功能。

HITSDB: 底层存储以分布式k-v的方式构建, 长处在于高性能的写入和快速读取时间线。

HITSDB: 通用的HTTP/Json协议, 有丰富的开源软件生态支持, 为Java提供专门的SDK。

