DTCC

第十四届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA

数智赋能 共筑未来

IT168.com ChinaUnix ITPUB

北京国际会议中心 | 2023/8/16-18

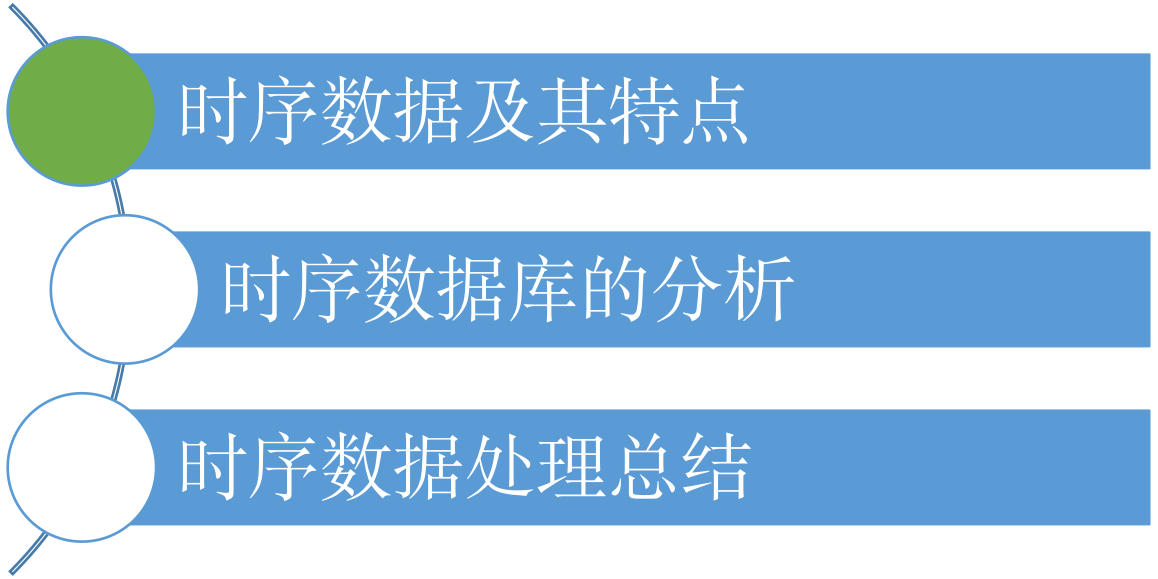
在物联网大数据背景下的时序数据库选型

TDengine 陈东明

个人简介

- 陈东明
- 创业者，曾任国美在线云平台中心总监，饿了么、百度架构师
- 主导开发饿了么KV数据库，负责百度即时通讯产品的架构设计
- 专注于大规模系统架构和基础架构领域和分布式技术研究
- 坚持技术文章创作和社区分享在个人博客blog.csdn.net/cadem和公众号’老陈聊分布式’
- 著有《分布式系统与一致性》一书

Agenda



什么是时序数据?

timestamp	CPU usage	host	zone	IDC
2022-05-05 00:22:21	21%	wfw.local	bj	yz

- Data point ;
- Tag – host, zone, IDC
- Serial -- host, zone, IDC的组合
- Timestamp
- Value – CPU usage

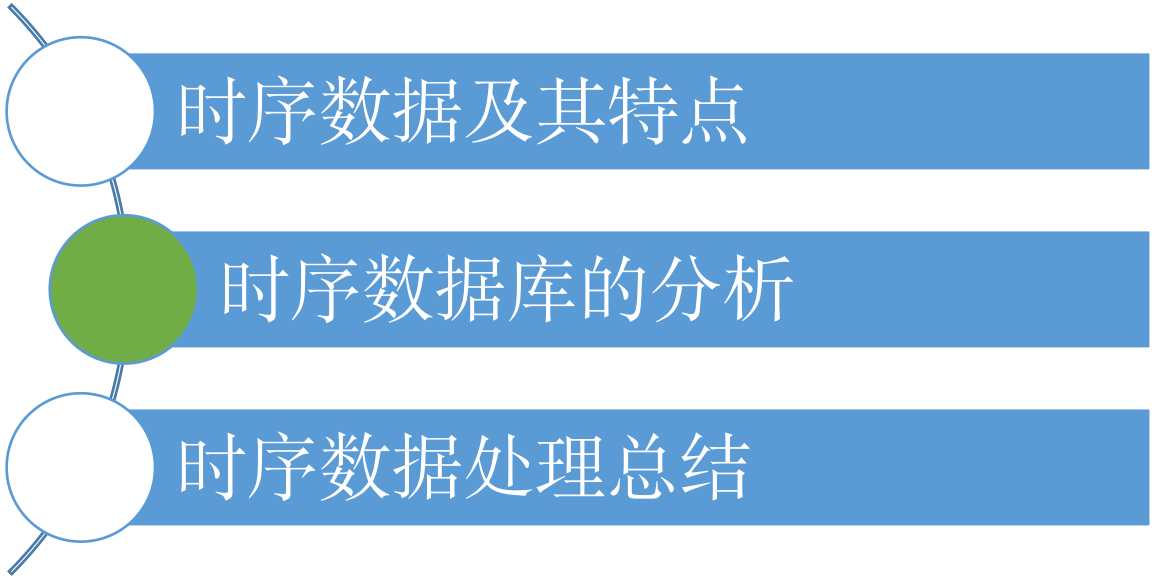
承载时序数据的困难

- 写入量大：并发写入量大
- 查询量大：单个数据价值低，往往是一次查询多条数据，或者进行聚合计算，单次查询的数据量大
- 删除量大：多进行批量删除
- 实时要求高：交互式查询，而非T+1式的大数据查询

时序数据的特点

- 数据较少更新/删除
- 数据多按日期删除
- 数据多指定时间段查询、聚合
- 数据多按tag维度筛选
- Tag高度重复
- 结构化
- 数据平稳
- 信息熵低

Agenda



InfluxDB数据模型

_time	_measurement	location	scientist	_field	_value
2019-08-18T00:00:00Z	census	klamath	anderson	bees	23
2019-08-18T00:00:00Z	census	portland	mullen	ants	30
2019-08-18T00:06:00Z	census	klamath	anderson	bees	28
2019-08-18T00:06:00Z	census	portland	mullen	ants	32

_measurement	tag set	_field
census	location=klamath,scientist=anderson	bees
census	location=portland,scientist=mullen	ants

InfluxDB Clustering

- Sharding
  - Shard group：按时间维度切分，一天一个shard group
  - Shard：
    - shard数量=node number/replica factor
    - shard := shardGroup.shards[fnv.New64a(key) % len(shardGroup.Shards)]
- Replica
  - Dynamo 风格
  - write consistency： one, quorum, all
  - hinted handoff

InfluxDB Storage

- WAL
- Cache
  - 按key分开
- TSM
  - Blocks部分： _time, _value，按列存储，按不同类型使用不同的压缩 算法
  - Index部分： key + 时间段 → blocks的位置
- Compaction
  - 多个小文件 → 大文件 → 按key拆开再合并

TDengine数据模型

Device ID	Timestamp	Collected Metrics			Tags	
		current	voltage	phase	location	groupid
d1001	1538548685000	10.3	219	0.31	California.SanFrancisco	2
d1002	1538548684000	10.2	220	0.23	California.SanFrancisco	3
d1003	1538548686500	11.5	221	0.35	California.LosAngeles	3
d1004	1538548685500	13.4	223	0.29	California.LosAngeles	2
d1001	1538548695000	12.6	218	0.33	California.SanFrancisco	2
d1004	1538548696600	11.8	221	0.28	California.LosAngeles	2

- 采集点：一般是一个设备，有唯一的标识
- 表：一个采集点一张表
- Tag：每一张表都有一个tag值的组合与之对应，这个表里所有point的tag都一样
- Serial：一张表相当于一个serial
- 超级表：同一类型表组成一个超级表，超级表中的所有表的结构都一样

TDengine Clustering

- Sharding
  - 时间维度（partition）： days参数控制，每个都是独立的一组文件
  - Tag维度（sharding）： 一致性hash(表名)->vgroup，vgroup的数量在创建数据库时指定
- Replica
  - Vgroup包含一组vnode
  - Vgroup内的vnode采用基于Raft的主从复制，数据强一致性，保证数据可靠性

TDengine Storage

- 先写入WAL和内存
  - 内存中按行存储，用skiplist做索引
  - 超过阈值后转成持久化，形成一个数据块
- 持久化
  - 按时间分组存储，每组包含文件：head, data
  - 按数据块存储在data中，按列存储，按不同类型压缩
  - Head是数据块的index，表名+时间段 → 在data文件中的位置

Timescale

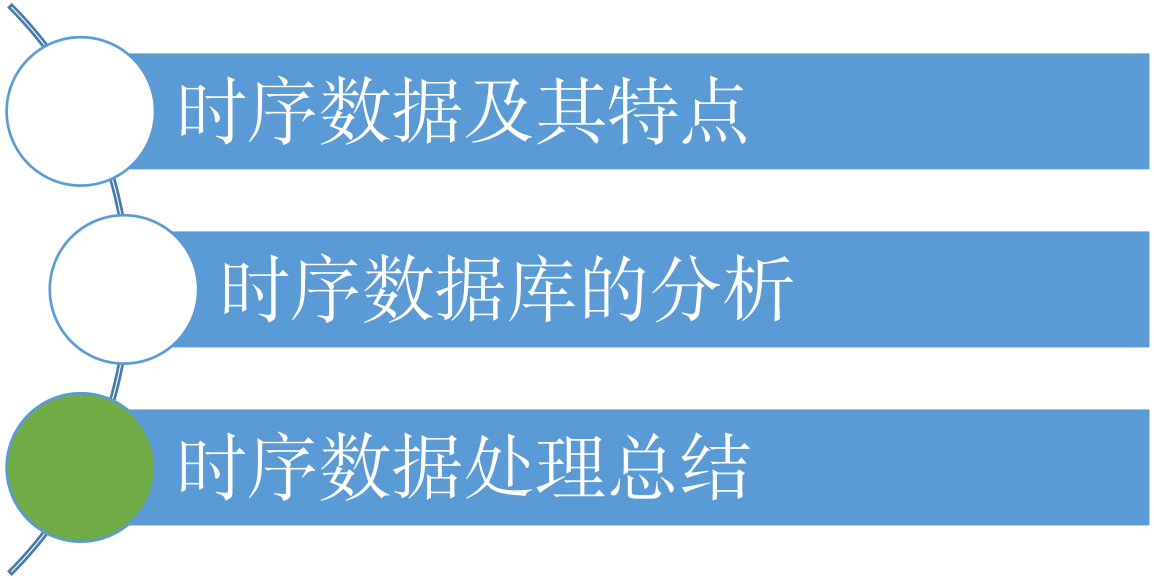
- 架构
  - PostgreDB
  - 以扩展方式
- 数据模型
  - Tables
  - Hypertables（第一个列为time列）

Timescale

- 按时间分表存储
- 按空间分到多个节点
  - space partitioning
  - distributed hypertables
- Compaction

time	device_id	cpu	disk_io	energy_consumption
[12:00:02, 12:00:01]	1	[88.2, 88.6]	[20, 25]	[0.8, 0.85]
[12:00:02, 12:00:01]	2	[300.5, 299.1]	[30, 40]	[0.9, 0.95]

Agenda



时序数据的特点

- 数据较少更新/删除
- 数据多按日期删除
- 数据多指定时间段查询、聚合
- 数据多按tag维度筛选
- Tag高度重复
- 结构化
- 数据平稳
- 信息熵低

按时间维度切分

按serial维度切分，减少tag存储，集群处理

列式存储，压缩

时序数据的特点

- 数据平稳
  - 时间间隔一般是有规律的，比如10s一个。
  - Value一般也是有一定幅度的波动，比较大的跳跃比较少，有的甚至没有波动

- RLE（run-length encoding）：360个重复的15
- Delta：前后2个值做差，去掉前面相同的，掐头
- Scaling：除以最大公约数（10的倍数），去掉后面0, 去尾
- Simple8b：把64位的整形切分多段，分别存入多个小整数
- XOR：针对浮点型数据，类似delta先后2个值做“差”，但用的是XOR

1538548685000	10.3	219	0.31
1538548684000	10.2	220	0.23
1538548686500	11.5	221	0.35
1538548685500	13.4	223	0.29
1538548695000	12.6	218	0.33

相关系统对比

- 关系型数据库
  - 无法高并发写入量
  - Tag高度重复，占用大量存储空间
  - 无时间类的聚合
- OLAP
  - 按时间切分
  - Druid也是按照时间来切分，所以也拿来做时序数据库来用
- KV数据库
  - 当storage
  - 缺少聚合查询能力
- 流式计算
  - 更丰富的基于时间的查询和聚合能力
  - 之前时序数据库往往要与流式计算系统联合使用，目前时序数据库有些已自己built-in流式计算能力

时序数据库的权衡

- 一致性
  - 强一致性 vs 弱一致性
  - 异步 vs 同步
  - 可用性上的差异
- 是否使用KV storage engine
  - 使用专有的符合时序数据特性的storage engine
  - influxdb:leveldb(LSM tree)->Boltdb(Btree)->TSM tree
  - Timescale使用的PostgreDB，本质上类似KV
- Schema vs schema-free

THANKS