
# DTCC

# 数，造，未，来

## 第十二届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2021

2021 年 10 月 18 日 - 20 日 | 北京国际会议中心

Spark
Hadoop
OLAP
HANA
MPP
Oracle
SQL

IT168.com ChinaUnix.net ITPUB

# 时序数据库终局破局开放探讨

姚延栋
yandong@ymatrix.cn
北京四维纵横数据技术有限公司

# About Me

姚延栋

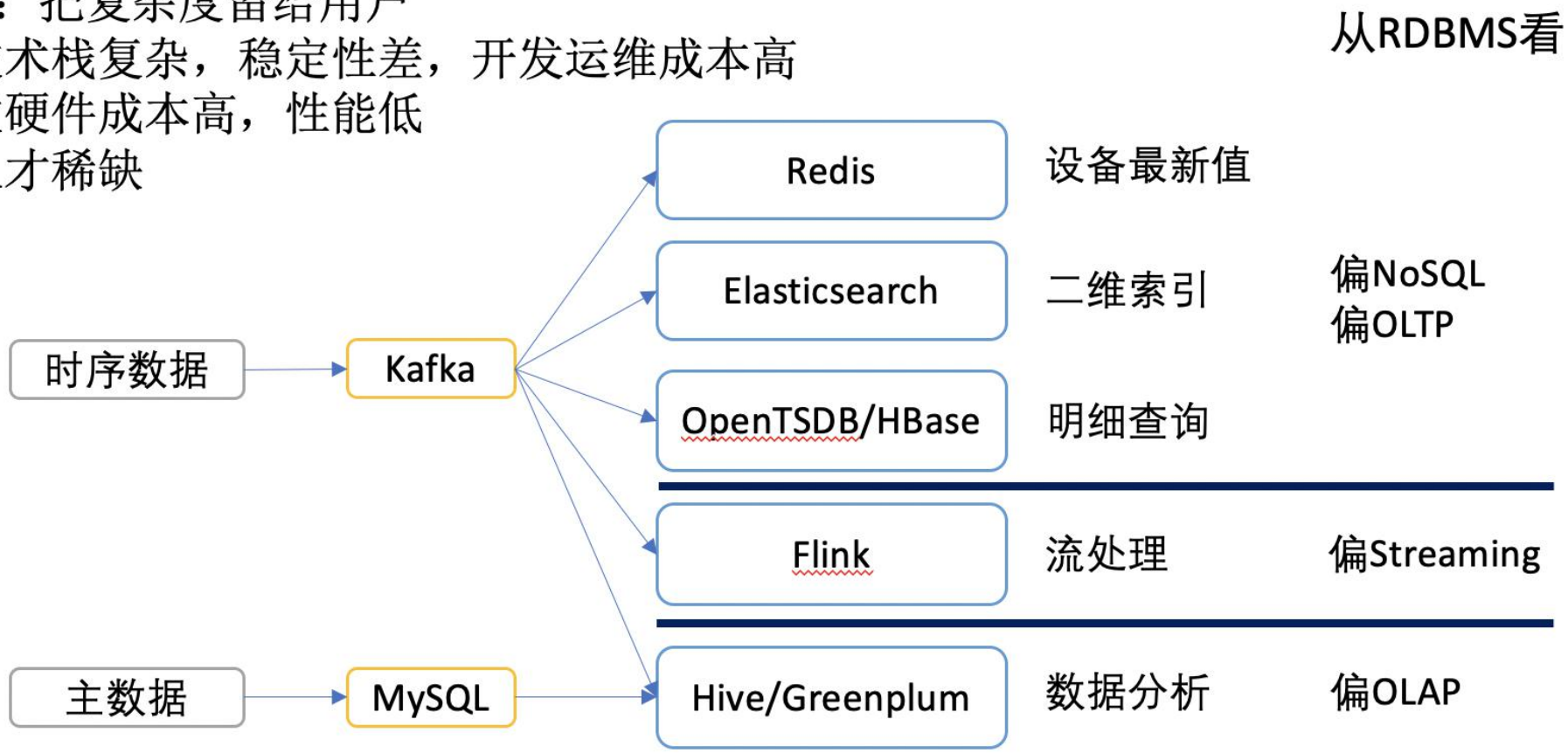
- 四维纵横（MatrixDB）创始人&CEO
- Greenplum 北京研发中心负责人（2010–2020）
- Greenplum 中文社区创始人
- PostgreSQL 中文社区常委
- 壹零贰肆数字基金会（NGO）联合创始人

# 当前时序架构问题：DIY技术栈，慢、复杂

慢：使用专用产品，看似更快，但端到端组合在一起会慢

复杂：把复杂度留给用户

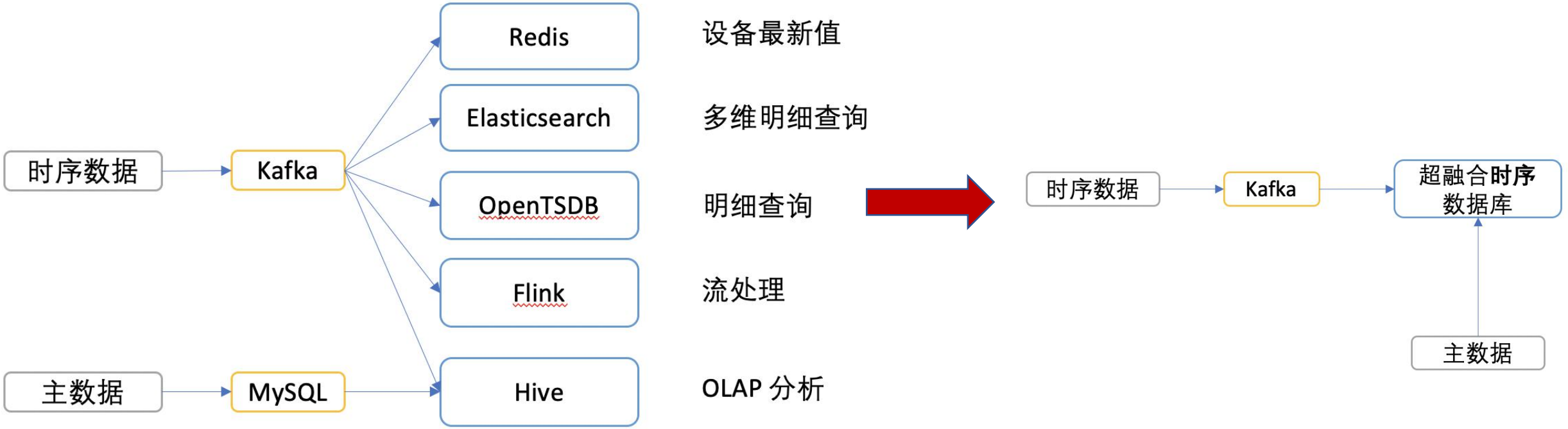
- 技术栈复杂，稳定性差，开发运维成本高
- 软硬件成本高，性能低
- 人才稀缺



# 下一代时序数据库架构：极速、极简

**奥卡姆剃刀原理：如无必要、勿增实体**
本质是谁简单、谁复杂?

超融合时序数据库解决复杂度问题
把**极简**、**极速**留给用户



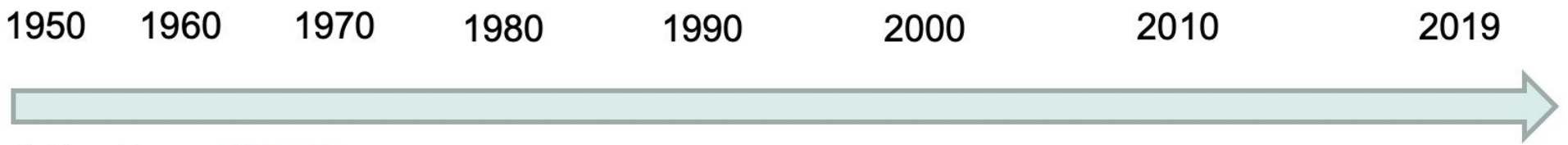
# Agenda

- 数据处理平台60年
- 主流时序数据库架构简析
- 时序数据库破局探讨

# 数据处理平台60年

- Pre–数据库：文件系统、主文件（flat文件格式）
- IDS、IMS：最早的数据库，解决数据共享和处理独立性问题
- 关系数据库（OLTP）：关系数据模型、SQL、set over record
- XML/对象数据库数据库：嵌套数据结构，内存结构更友好
- NoSQL：扩展性、高性能、高并发，解决互联网大数据挑战
- OLAP数据库：MPP、对分析场景复杂查询优化
- NewSQL、HTAP、Lakehouse、多模：不同层次、角度跨界融合
- 超融合数据库：多层次、多角度深度融合，分久必合

# 过去60年大约每十年一次技术PK




文件系统 vs. 数据库

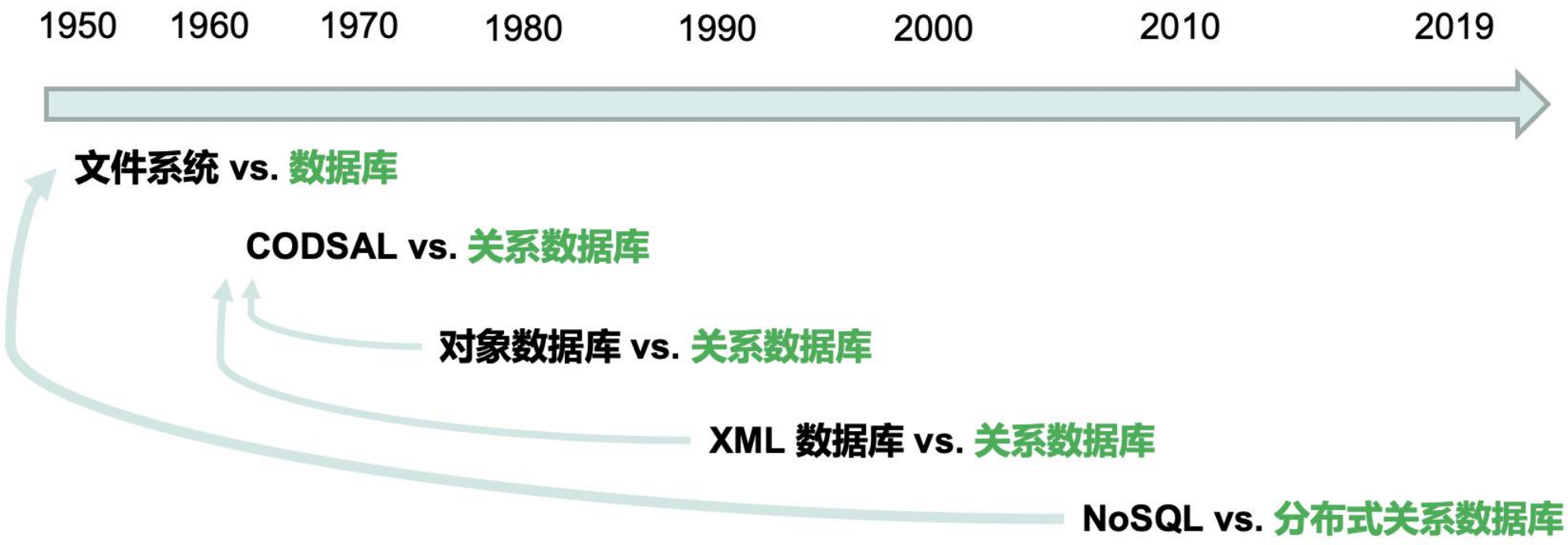
CODSAL vs. 关系数据库

对象数据库 vs. 关系数据库

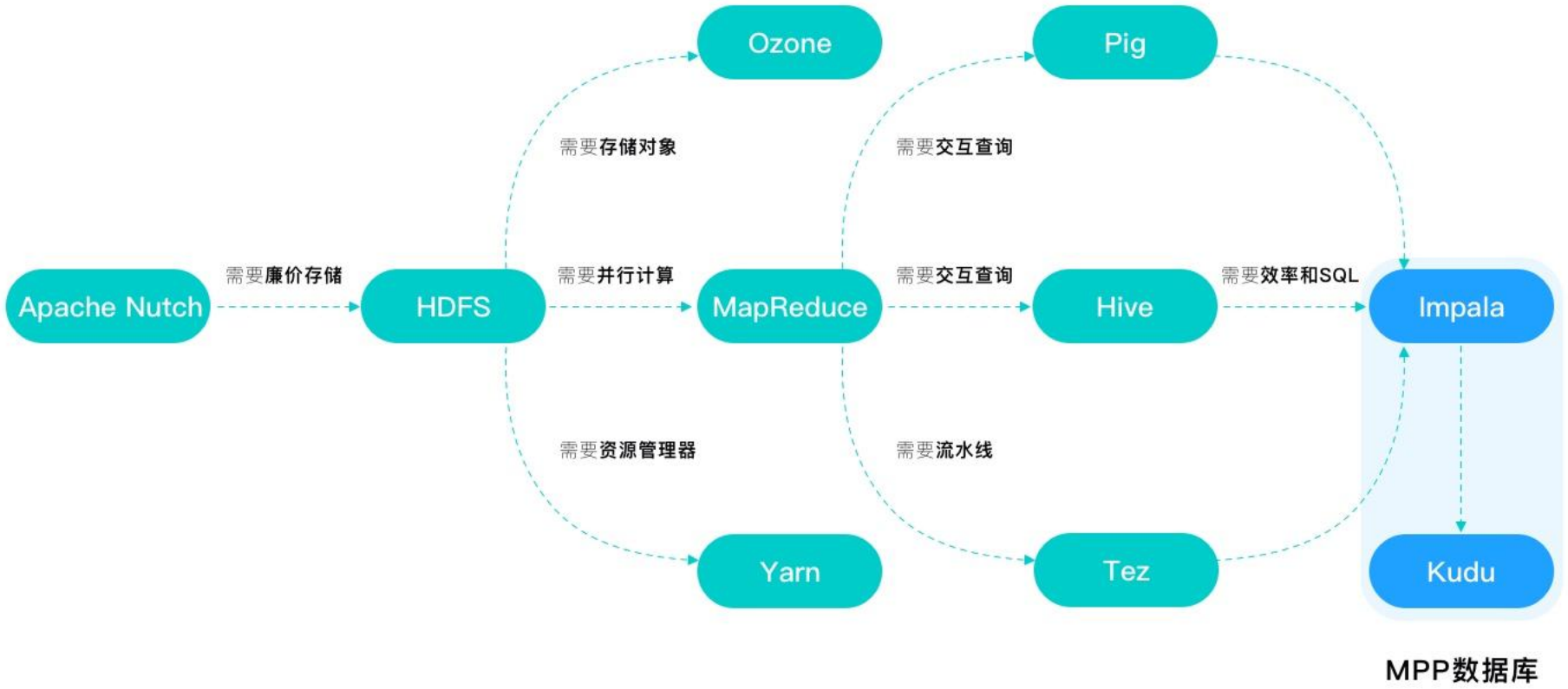
XML 数据库 vs. 关系数据库

NoSQL vs. 分布式关系数据库

# 历史总是在不断的重复

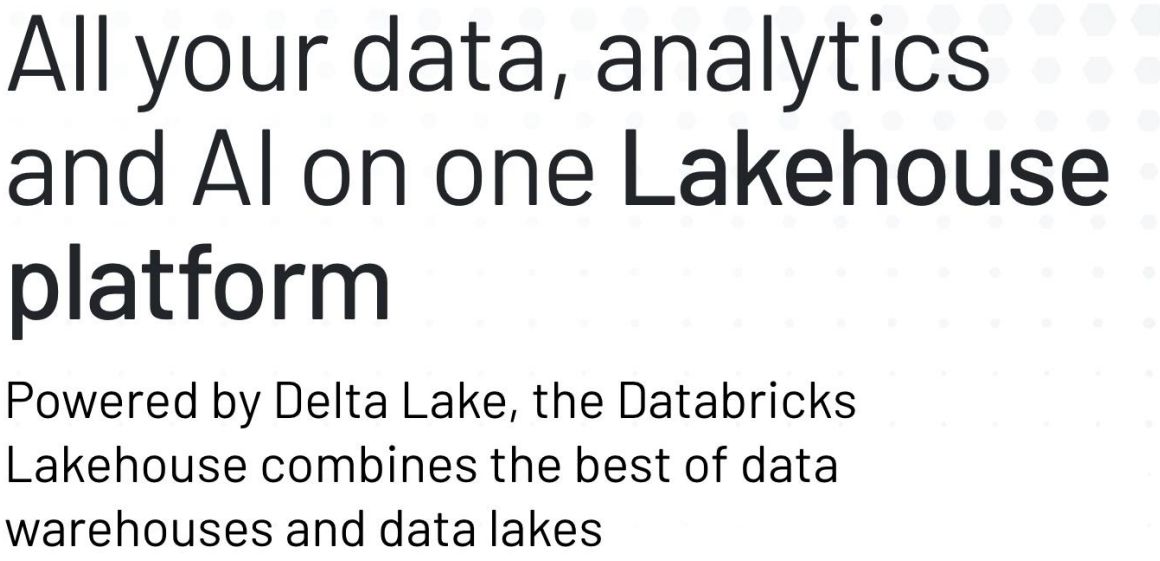

1950
1960
1970
1980
1990
2000
2010
2019
文件系统 vs. 数据库
CODSAL vs. 关系数据库
对象数据库 vs. 关系数据库
XML 数据库 vs. 关系数据库
NoSQL vs. 分布式关系数据库

# Hadoop 演进逻辑：从存储到计算



# Spark 演进逻辑：从计算到存储

- Spark RDD
- Spark SQL and DataFrames
- Spark Streaming
- Spark Graph
- Spark ML
- DeltaLake
- One data platform

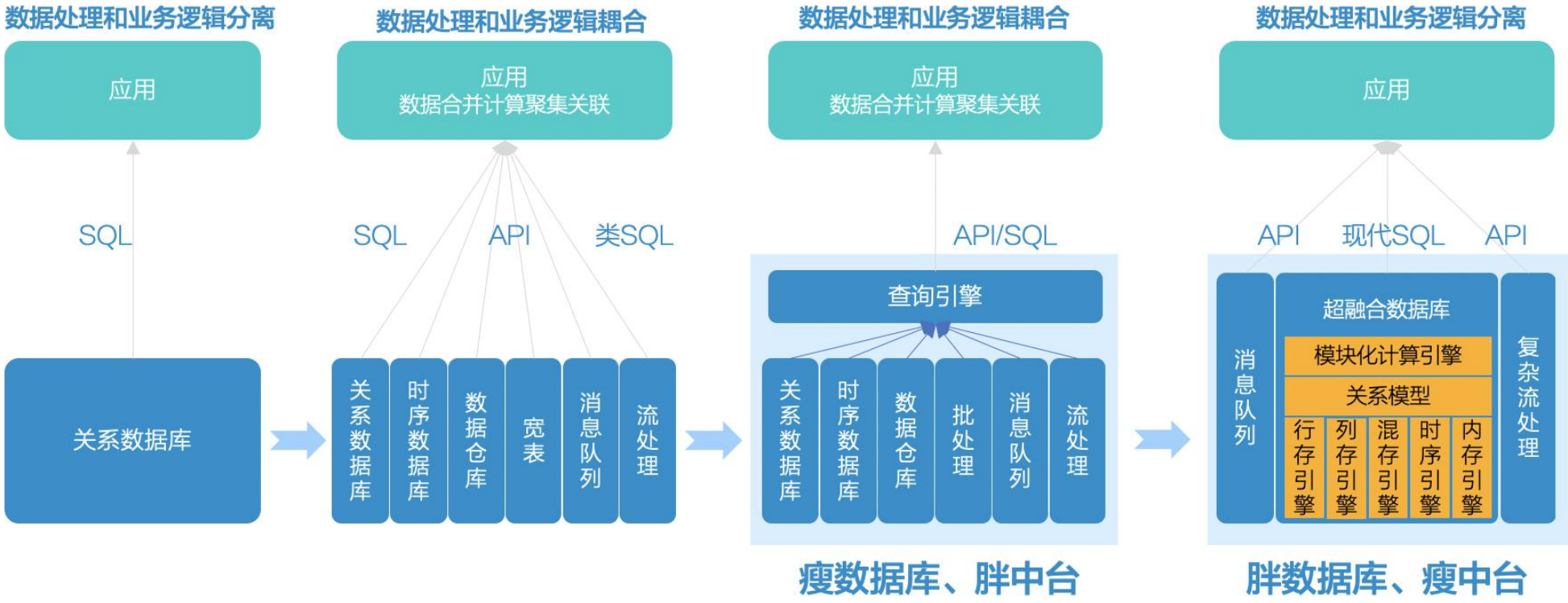


# Snowflake 演进逻辑：从雪花到雪球

- 存算分离、存算同时演进
- 雪花到雪球，然后到更大的雪球
- One data platform

# 数据库演进：处理复杂度的游戏

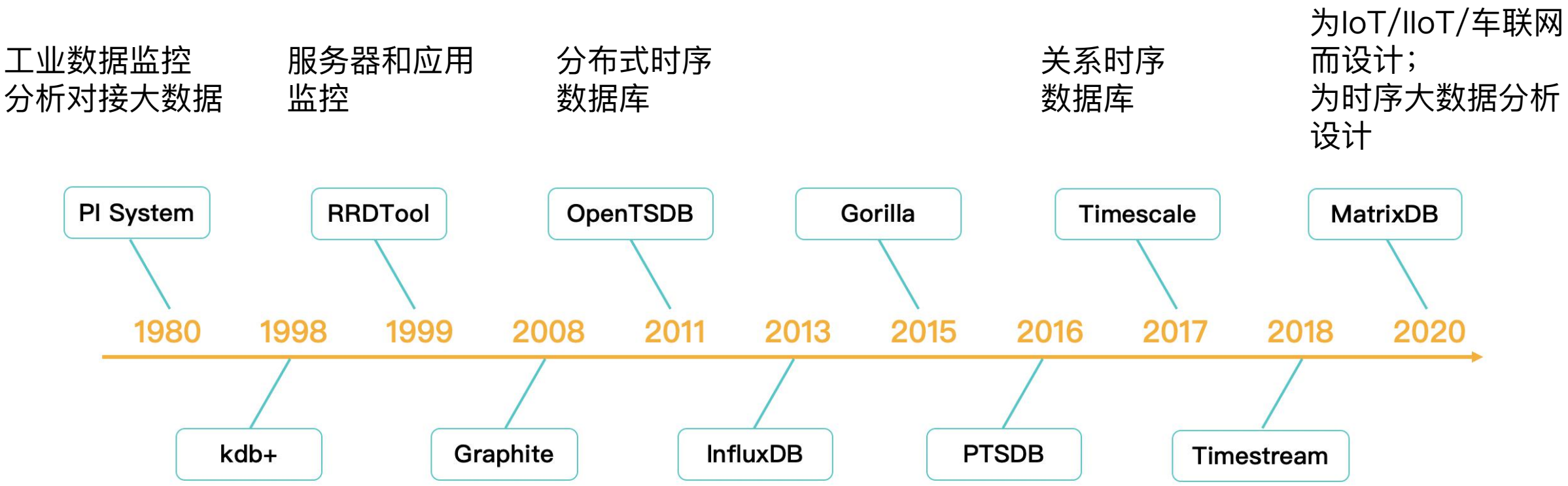
核心是需求推动之下，选择解决哪些复杂度问题，留哪些复杂度问题给用户；解决的怎么样



# Agenda

- 数据处理平台60年
- 主流时序数据库架构简析
- 时序数据库破局探讨

# 时序数据库：80年代出现，近期繁荣



# 经典时序数据库架构

- OSISoft PI System：
  - PI archiver：自研时序引擎
  - 微软 SQLServer：内部集成了关系数据库处理关系数据
- KDB+：列式时序数据库，closed source
- RRDTool：
  - 以固定时间间隔采集数据、round-robin方式存储数据
  - 最新数据覆盖老数据
- Graphite：
  - 数据引擎是Whisper，类似RRD，数据库大小固定，
  - 支持降采样

# 经典时序数据库架构（续）

- OpenTSDB：
  - 基于HBase，第一款分布式时序数据库，
  - 针对时序KV做了优化，譬如一小时时间点散列为列族的不同列
- InfluxDB：
  - 不断迭代存储引擎：LevelDB/RocksDB → TSM+TSI → Parquet+Arrow
  - 下一代产品 Iox 主打分析能力和扩展性
- TimescaleDB：
  - 基于PostgreSQL，hack存储层，1000+ 行 → 1行
- MatrixDB：
  - 基于Greenplum（PostgreSQL 12内核），
  - 自研专为时序场景优化的存储引擎
  - 支持关系数据、时序数据、GIS数据
  - 支持监控类小查询和大 OLAP 分析，支持JOIN，支持完善SQL标准

# 时序13条：行业第一眼中的下一代时序数据库

| InfluxDB iox功能目标 |
| --- |
| No limits on cardinality. Write any kind of event data and don't worry about what a tag or field is. |
| Best-in-class performance on analytics queries in addition to well-served metrics queries. |
| Separate compute from storage and tiered data storage. The DB should use cheaper object storage as its long-term durable store. |
| Operator control over memory usage. The operator should be able to define how much memory is used for each of buffering, caching, and query processing. |
| Operator-controlled replication. The operator should be able to set fine-grained replication rules on each server. |
| Operator-controlled partitioning. The operator should be able to define how data is split up amongst many servers and on a per-server basis. |
| Operator control over topology including the ability to break up and decouple server tasks for write buffering and subscriptions, query processing, and sorting and indexing for long term storage. |
| Designed to run in an ephemeral containerized environment. That is, to run with no locally attached storage. |
| Bulk data export and import. |
| Fine-grained subscriptions for some or all of the data. |
| Broader ecosystem compatibility. Aim to use and embrace emerging standards in the data and analytics ecosystem. |
| Run at the edge and in the datacenter. Federated by design. |
| Embeddable scripting for in-process computation. |

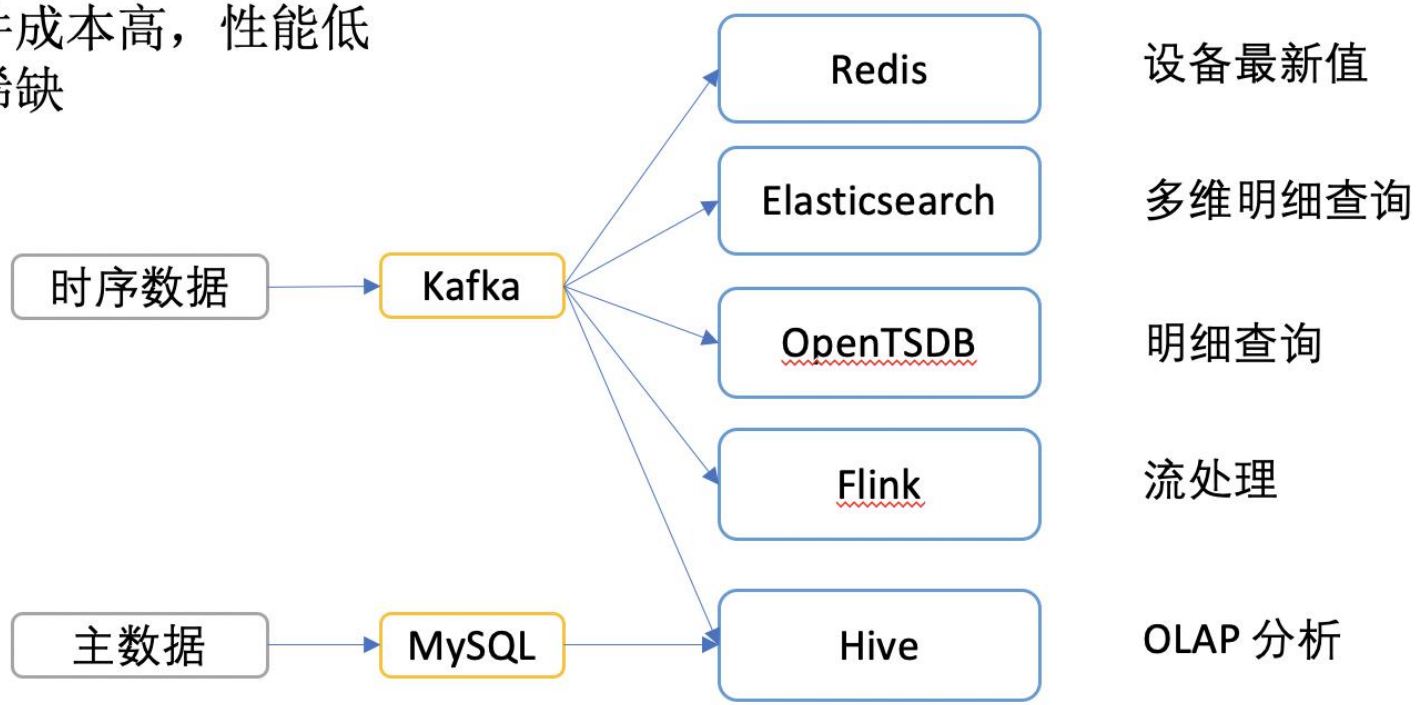
# Agenda

- 数据处理平台60年
- 主流时序数据库架构简析
- 时序数据库破局探讨

# 当前时序架构：DIY技术栈，慢、复杂

慢：使用专用产品，看似更快，但端到端组合在一起会慢
复杂：把复杂度留给用户

- 技术栈复杂，稳定性差，开发运维成本高
- 软硬件成本高，性能低
- 人才稀缺

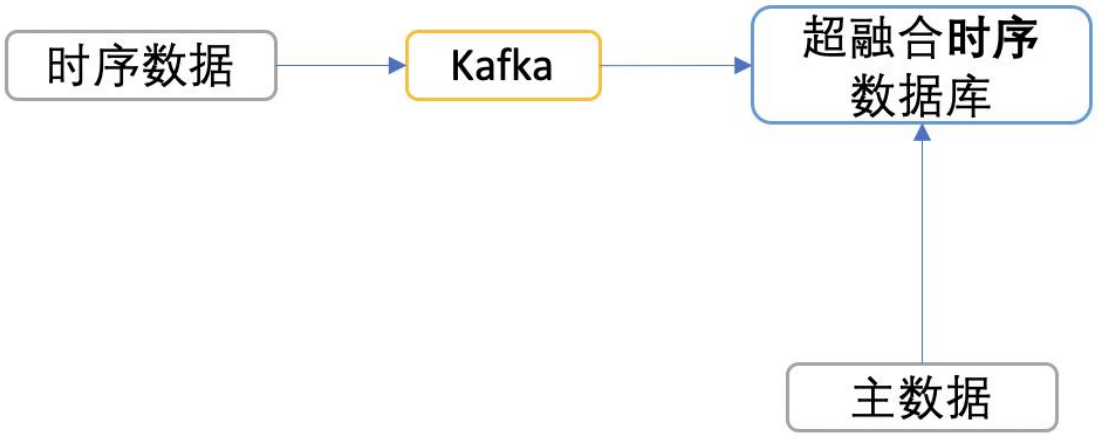


# 时序数据库应该怎么样：快、简单

**奥卡姆剃刀原理：“如无必要、勿增实体”**

- 本质是谁简单，谁复杂？能否承受这种复杂？

超融合数据库：让用户更简单




设备最新值

多维查询

明细查询

OLAP 分析

机器学习

流处理

| | |
|---|---|
| 这 | E8BF99 |
| 么 | E4B988 |
| 拽 | E68BBD |
| ？ | EFBC9F |


把事情变复杂很简单，把事情变简单很复杂

# 回归场景

## 写

- 平稳高效实时（95%+）
- 写入模式突出
- 延迟写入
- 分批写入，自动合并
- 异频写入
- 更新删除
- 动态增删指标

## 读

- 点查/最新值
- 明细查询
- 聚集查询
- 多维查询
- 峰值检测、移动平均值、跳变检测、线性回归等
- 高级分析（eg：故障模型）

# B+树/LSM树：关系数据库不能做时序?

- B+树为读而优化，LSM树为写而优化
- 时序场景95–99%为写操作，所以大多数时序数据库使用类LSM树方式
- 然后基于B+树的 MatrixDB 写入性能是基于LSM/TSM的 InfluxDB 的几十倍，是国内友商的十几倍（仅仅是比较一下，不是说B+树比LSM好，也不是说B+树是最好的方案）
- Why?

PG中文社区第三方评测：http://www.postgres.cn/v2/news/viewone/1/697

# 认知误区

B+树为读而优化

为了提升写性能：

- 分页随机IO造成写性能低
- WAL顺序写
- Buffer尽量避免随机IO
- 分区避免B+树太大
- 时序数据顺序特征利用

LSM树为写而优化

为了提升读性能：

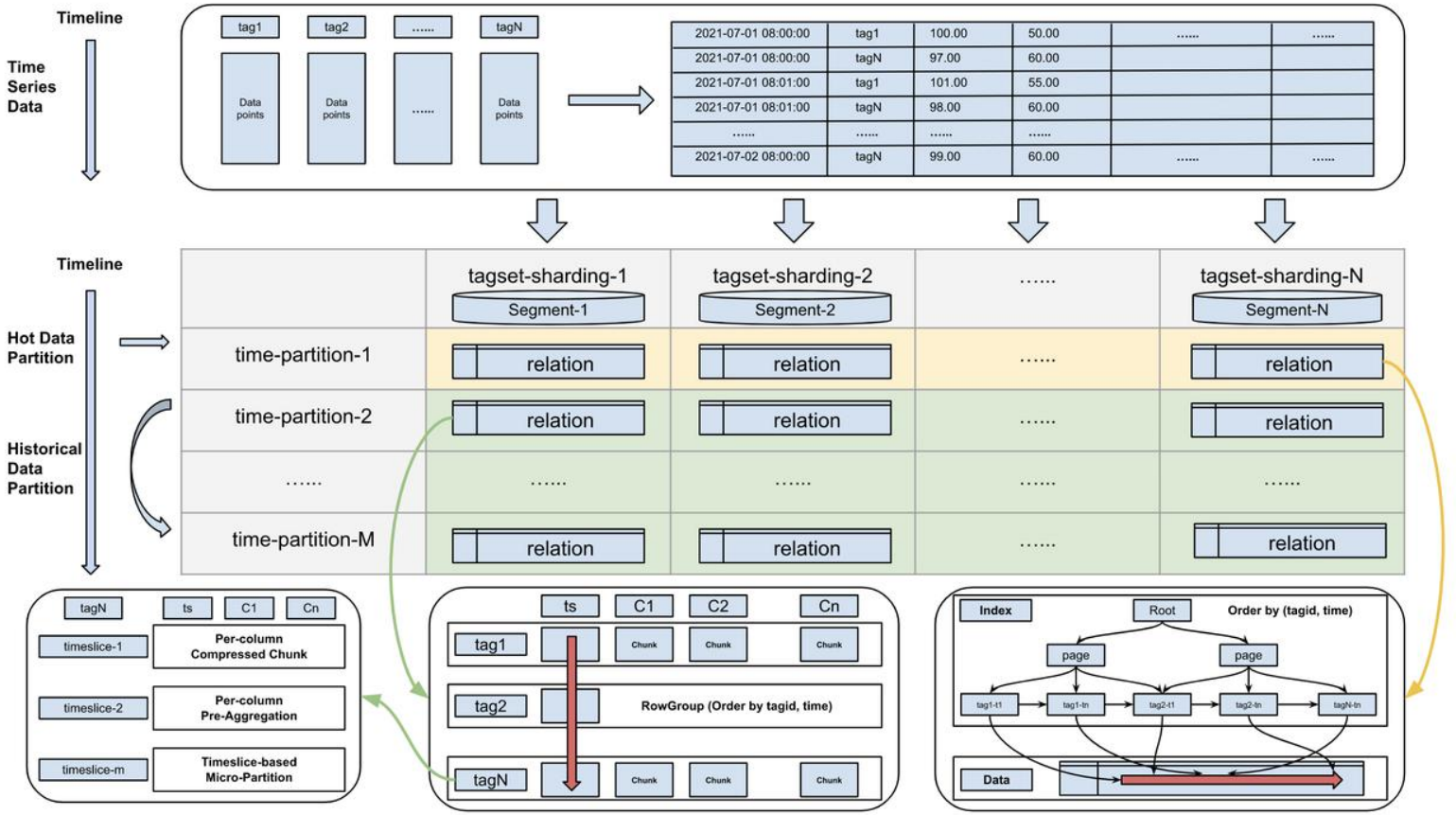
- 内建B+树加速读性能
- 内建Bloomfilter加速读
- 创建倒排索引加速查询
- WAL 避免丢数据

# 超融合架构：把极简、极速留给客户

# 从开发运维角度看超融合时序数据库

# 超融合时序数据库内核：以MatrixDB为例

设备属性信息
存在 heap 表




设备时序数据
根据时间分区；
热数据存在 heap 表
温冷数据存 mars 表；
线性扩展

# 数据存储格式：行列混存，以MatrixDB为例

两个设备，10:01–10:04分，温度、湿度时间序列

| 设备id | 时间戳 | 温度 | 湿度 |
|---|---|---|---|
| 10001 | 2021/03/11 10:01 | 37.1 | 80.1 |
| 10001 | 2021/03/11 10:02 | 36.5 | 60.7 |
| 10001 | 2021/03/11 10:03 | 37.2 | 80.0 |
| 10001 | 2021/03/11 10:04 | 36.4 | 60.4 |
| 10001 | 2021/03/11 10:03 | 37.0 | 80.3 |
| 10001 | 2021/03/11 10:04 | 36.8 | 60.8 |
| 10002 | 2021/03/11 10:01 | 36.8 | 50.3 |
| 10002 | 2021/03/11 10:02 | 36.9 | 56.6 |
| 10002 | 2021/03/11 10:03 | 36.9 | 50.2 |
| 10002 | 2021/03/11 10:04 | 36.8 | 56.8 |
| 10002 | 2021/03/11 10:03 | 36.1 | 50.2 |
| 10002 | 2021/03/11 10:04 | 37.0 | 56.3 |

Partition内分块，一个设备数据存储在一个块内

# 时序数据写入

各种姿势

- 顺序上报
- 乱序上报
- 延迟上报
- 异频上报
- 上报信号/指标动态增减
- 更新或删除
- 自动降采样、持续聚集

场景特色

- 高性能：95%以上操作是插入，性能敏感
- 平稳持续：没有明显的高低峰
- 实时写入：秒级、分钟级，少量小时级或天级
- 数据正确性：不错、不重、不丢，
  1）Garbage In、Garbage Out；
  2）聚焦业务，而不是数据处理

# 时序数据存储

存储效率/压缩比，时序数据10：1左右，范式型数据压缩比高达几十倍

- 列式编码压缩
- 行式压缩
- 块压缩
- 多种编码压缩算法，针对数据类型优化，包括delta-delta、gorilla、RLE、lz4、zstd
- 多种压缩级别，达到压缩比和压缩速度的平衡

冷热分级存储

- 热数据、温数据、冷数据分层，最小化存储开销
- 自动分区管理，无需人工干预

# 多样性数据类型，不仅仅是时序

- 基本数据类型：字符串、数字、日期/时间、布尔类型
- 复合数据类型：数组、IP地址等
- KV 数据类型，兼顾效率和schemaless的灵活性
- 空间数据类型（和函数）：点、线、多边形等
- XML/JSON 半结构化类型
- 非结构化数据类型：文本
- 自定义数据类型

# 查询快、并发高、大小查询

查询

- 单表：点查、明细、聚集、多维查询
- 多表：JOIN，支持10+表关联
- 高级分析：子查询、窗口函数、Cube、CTE等
- ML模型训练和推理

查询类型

- OLTP 型查询，TPCB 单机30万tps-60万tps
- OLAP 型查询：TPCH，比Greenplum快3倍
- 时序类型查询：first, last, 窗口查询等
- 空间数据类型查询：范围查询、相交查询等

# 数据库内建分析，比 Spark 快 10 倍

数据库内建查询，计算贴金数据，高效简洁

- Python/R/Java：应用Python/R/Java代码原地处理库内海量数据，可以重用大量函数库，譬如Pandas、Numpy等
- Tensorflow：使用主流AI库对数据库内部海量数据原地训练和分析
- SQL 机器学习：监督学习、无监督学习、统计分析、图计算等

# 完善的开发工具和生态，开箱即用

- 支持各种流行IDE，包括IntelliJ、DataGrip、Dbeaver、Navicat 等
- BI和可视化：Grafana、Tableau、永洪、帆软、SAS、Cognos、Zabbix 等
- ETL/CDC：Informatica、Talend、DSG、DataPipeline、HRV 等
- IoT 协议：MQTT、OPC-UA、OPC-DA、MODBUS等
- IoT 平台：Node-Red、ThingsBoard

# 企业级稳定性、完备性，省心放心

- 图形化部署；线性扩展，可以单机部署，也可以集群化部署
- 监控、报警以及可视化
- 在线扩容，不停机，不停业务
- 分布式备份恢复
- 资源管理：CPU、内存、并发等
- 分钟级升级

# 时序数据库破局

时序形态不是数据库，而是一种数据类型，终局是超融合时序数据库

## 第一代

时序形态：数据库

需要关系型数据库配合
偏监控，仅支持小查询

influxdata

## 第二代

SQL关系数据库

时序形态：数据类型

需要OLAP、Streaming方案配合
偏监控，仅支持小查询

Timescale

## 第三代

超融合时序数据库

时序形态：数据类型

OLTP+OLAP+Streaming
监控分析并重

智能化重视分析，一库搞定

MATRIX

# 谢谢

更多资讯请访问 http://ymatrix.cn

小M助手


THANKS
时序
开源
自研
分布式
文档
Spark
HANA
SQL
OLAP
Oracle
Hadoop
MPP
DB2