# 时序型数据库预研报告

1. 背景
   1. 目的

调研时序型数据库的目的是随着IOT下物联设备产生的的海量数据，传统的关系型数据库在存储效率、查询效率、空间利用率上体现出了巨大瓶颈，因此时序型数据库在这种条件下应运而生，专为解决关系型数据库在这些方面的痛点。

* 1. 简介

时序数据是随时间不断产生的一系列数据，简单来说，就是带时间戳的数据。时序数据库 (Time Series Database，TSDB) 是优化用于摄取、处理和存储带时间标签（按照时间的顺序变化，即时间序列化）数据的数据库，带时间标签的数据也称为时间序列数据。 虽然其他数据库也可以在数据规模较小时一定程度上处理时间序列数据，但TSDB可以更有效地处理随时间推移的数据摄取、压缩和聚合。简而言之，时序数据库是专门用于存储和处理时间序列数据的数据库，支持时序数据高效读写、高压缩存储、插值和聚合等功能。

* 1. 特点
* 产生频率快

以常见数据采集、发送数据为例，分为高低优先级数据，比如5秒采集一个点的数据，则每天一个单采集点会产生60/5\*60\*24 =17280条数据

* 严重依赖采集时间

设备数据每一条数据要求对应唯一时间

* 测点多、信息量大，但结构相对简单

常规的实时监测系统均有成千上万的监测点，监测点每秒钟都产生数据，每天产生大量的数据

* 抵达的数据几乎总是作为新条目被记录
* 数据通常按照时间顺序抵达
* 时间是一个主坐标轴（既可以是规则的时间间隔，也可以是不规则的）
* 基本不修改已有数据

1. 主流时序型数据库介绍

参考互联网上的数据，首先是来自知名数据库排名网站：<https://db-engines.com/en/ranking/time+series+dbms>，然后本论文其他数据参考了互联网上公共的文档及数据。

* 1. 当前时序型数据库前三十名排名

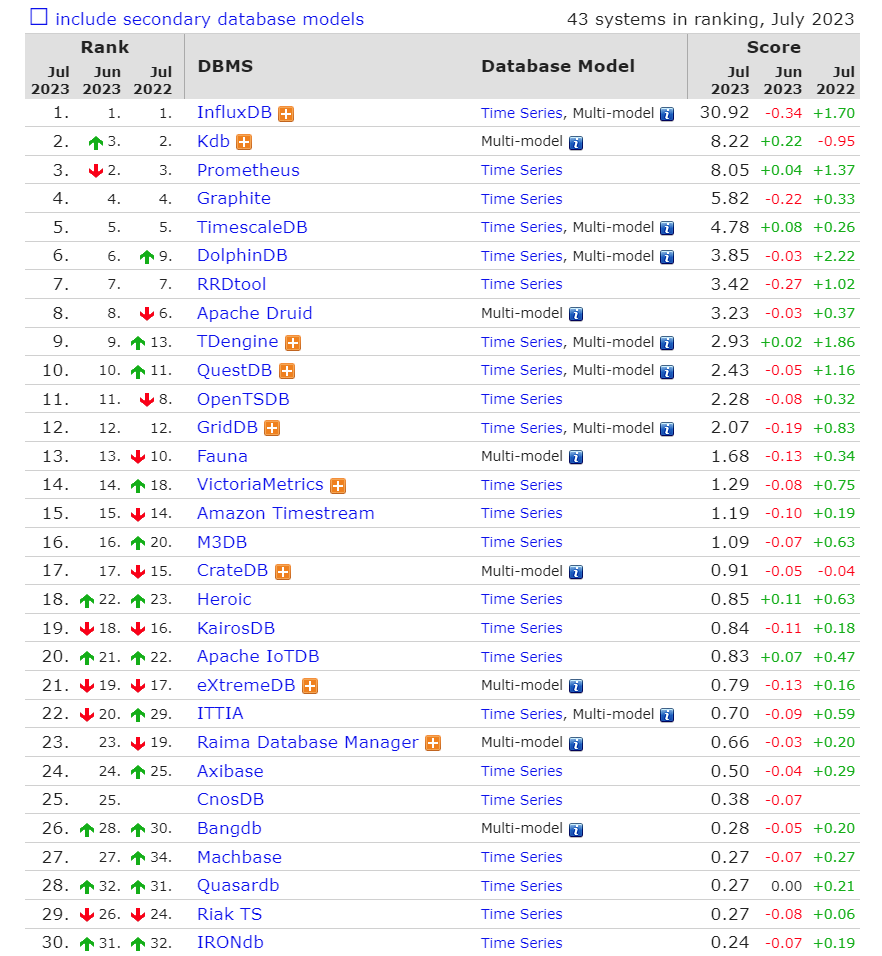


图2.1 时序型数据库排名

正如上图所示，时序型数据库有老牌的龙头数据库，也有迸发着勃勃生机的新兴后起者。毋庸置疑的一点是老牌的时序型数据库在经过这么多年的锤炼，在稳定性和生态方面肯定更完善以及更有优势，而新生代的数据库在性能及其他可观测的表现上，相较于老牌的数据库也凸显出了绝对的优势，但是也正因为它们的年轻，对它们的试炼还相对较少，以及缺少产品周边的生态和完备的文档说明这是它们目前面临的最大问题。所以在选择时，我们需要根据自己的实际情况，做出抉择。

* 1. 预研包含的数据库说明

此次预研进入待选池的时序型数据库包含：InfluxDB、TDengine、Apache IoTDB：

* + 1. InfluxDB待选理由

InfluxDB是当下使用最广泛的时序型数据库，且与第二名有着非常大的断层，突出了InfluxDB的垄断地位。又因为其使用广泛，经过了许多大厂、极端环境下的考验，说明了InfluxDB还是相对稳定的，且社区活跃度高，有着许多成熟的案例。

* + 1. TDengine待选理由

优秀的新生国产时序型数据库黑马，属于后起之秀。在短短的两年时间里，排名和使用范围正在逐年以极强的势头增长，可以预测在不久的将来，时序型数据库必有它的一席之地。更重要的原因是，在各方面的评测对比中，它的表现都是以极大的优势优于老牌的InfluxDB数据库。

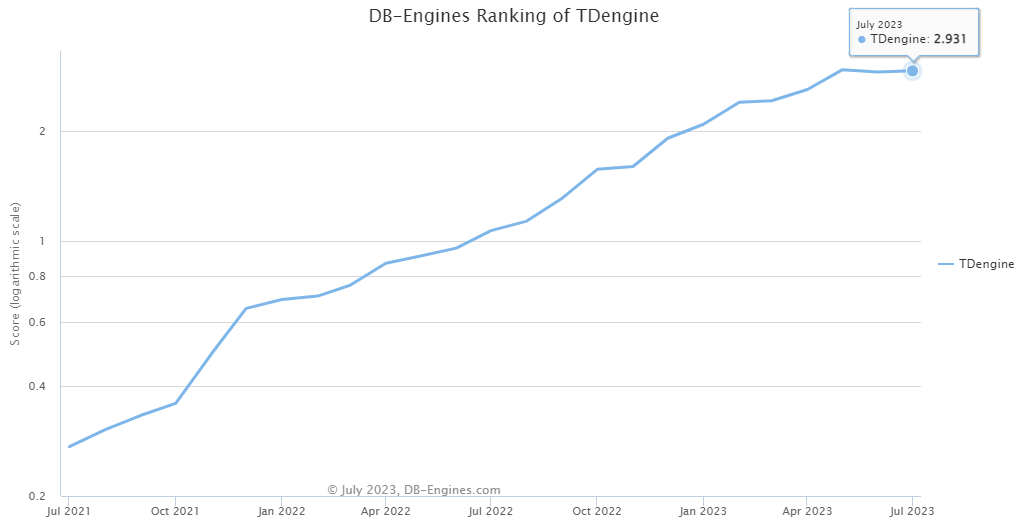


图2.2 TDengine的成长趋势

* + 1. Apache IoTDB待选理由

IoTDB也是国产开源项目，后成为了Apache的孵化项目，它是专为物联网打造的时序数据库，提供数据采集、存储、分析的功能。IoTDB 提供端云一体化的解决方案，在云端，提供高性能的数据读写以及丰富的查询能力，针对物联网场景定制高效的目录组织结构，并与 Apache Hadoop、Spark、Flink 等大数据系统无缝打通；在边缘端，提供轻量化的 TsFile 管理能力，端上的数据写到本地 TsFile，并提供一定的基础查询能力，同时支持将 TsFile 数据同步到云端。

* 1. 预研未包含的时序型数据库说明

当然，此次在DB-Engines排名中位于前列的一些时序型数据库未考虑在待选中，在这列举出相关的原因：

* Kdb：这款时序型数据库也是老牌数据库，业界给它的标签是久负盛名、最快、最贵、最小 的时序数据库。拥有着许多大厂的实际使用案例，但是因为它需要使用特定的K/Q脚本语言开发，接入的难度稍微有点大，而且是属于商用型的数据库；
* Prometheus：它的使用场景更多的是构建一套监控大盘的告警系统；
* Graphite：开发语言仅支持JavaScript和Python，服务器只支持Linux和Unix，使用场景也是用于监控场景的居多；
* TimescaleDB：基于postgresql的一个扩展，对大批量数据插入和复杂查询做了优化，但是底层仍然是postgresql，故继承了postgresql的所有特性，timescaledb数据存储原理也是使用postgresql分区表来实现，导致了它在数据存储上并没有体现出时序数据库的优势；
* DolphinDB：国产数据库，商用版本，开发了自己的编程语言，更适用于金融领域；
* Apache Druid：不支持Java开发语言和Windows操作系统；
* QuestDB：不支持更新、支持 designated timestamp列（带时间戳的列）在设定时间内，设定行数限制内的乱序数据写入，极大的限制应用场景；
* OpenTSDB：基于Hbase的时间序列数据库，其基于Hbase的分布式列存储特性实现了数据高可用、高性能写的特性，受限于Hbase，其所需存储空间较大，压缩不足，且依赖整套的HBase和 ZooKeeper服务；
  1. InfluxDB简介

InfluxDB 是用Go语言编写的一个开源分布式时序、事件和指标数据库，无需外部依赖。用于存储和分析时间序列数据的开源数据库。适合存储设备性能、日志、物联网传感器等带时间戳的数据,其设计目标是实现分布式和水平伸缩扩展。

InfluxDB 包括用于存储和查询数据，在后台处理ETL或监视和警报目的，用户仪表板以及可视化和探索数据等的API。

InfluxDB是一个由InfluxData开发的开源时序型数据。它由Go写成，着力于高性能地查询与存储时序型数据。InfluxDB被广泛应用于存储系统的监控数据，IoT行业的实时数据等场景。

* + 1. InfluxDB优点
* 专为时间序列数据编写的自定义高性能数据存储。 TSM引擎允许高摄取速度和数据压缩
* 完全用 Go 语言编写。 它编译成单个二进制文件，没有外部依赖项
* 简单，高性能的写入和查询HTTP API
* 插件支持其他数据提取协议，如Graphite，collectd和OpenTSDB
* 专为类似SQL的查询语言量身定制，可轻松查询聚合数据
* 标签允许对系列进行索引以实现快速有效的查询
* 保留策略有效地自动使过时数据过期
* 连续查询自动计算聚合数据，以提高频繁查询的效率
  + 1. InfluxDB缺点
* InfluxDB的开源版本只支持一个节点
* 开源版本没有集群功能，集群版本需要收费
* 1.x和2.x版本存在前后版本兼容问题
* 存储引擎在变化
* 2.x版本需要额外学习Flux语言
  + 1. InfluxDB的客户

InfluxData拥有超过1400名付费客户，其中客户包括MuleSoft，IBM，PayPal，Siemens，Tesla。

* 1. TDengine简介

TDengine 是一款开源、高性能、分布式、支持 SQL 的时序数据库，其时序数据库核心代码包括集群功能全部开源，同时 TDengine 还带有内建的缓存、流式计算、数据订阅等系统功能，能大幅减少研发和运维的复杂度。

* + 1. TDengine优点：
* 支持水平扩展，有开源的分布式集群解决方案；
* 性能测试中读写性能远高于InfluxDB，压缩率高；
* 采用标准 SQL 做查询语言（不完全兼容），并且采用关系数据库模型，学习成本低；
  + 1. TDengine缺点：
* 比较复杂的sql功能还不支持
* 单条插入性能很低，必须成批写入，增加了系统开发和维护的复杂度与运营成本
* 大厂实践少
  + 1. TDengine的使用案例
       1. TDengine 助力摇光实现技术改造

作为一个大而全的数据库，OpenTSDB 稍显笨重，伴随着业务需求的不断迭代及数据量的不断上涨，其局限性日益凸显，系统的架构升级和改造工作日渐迫切。

2021 年我们在对 TDengine 有了充分的了解后，决定将至数摇光从时序数据 OpenTSDB 迁移到 TDengine，并基于 TDengine 的特性对摇光进行彻底性的改造。目前改造工作已经全部完成，改造后有大约 80% 左右的指标模型放到了 TDengine 中，20% 左右的主数据或维表数据存放在 MySQL 数据库中。相较于改造前的 80% 指标模型存放在 MySQL 中，20% 指标数据存放在 OpenTSDB 数据库中，结果刚好进行了颠倒，服务器资源使用情况也有所下降。应用整体的页面影响速度显著提高，数据模型及数据指标上也可以更加地灵活多变。

* + - 1. TDengine 在大疆车载智能驾驶云端平台上的应用

由于当前的智能驾驶业务还是新的业务场景，所以大疆车载在选型上的历史负担相对较轻。在 Database 选型要求上，从业务需求出发，主要聚焦在两点：首先，结合当下的业务场景，需要满足单台车辆的高频消息上报频率；其次，支持在数据量大的时候，通过聚合函数，或选择函数来快速筛选出需要的数据。

此外，对数据库要求支持集群部署的同时，也要求更低的查询语句编写上手难度；而且需支持单表千万量级，在海量数据并发场景下，需要有较高的统计报表能力和较好的查询 SQL 效率；最后通过数据压缩、运维成本和并发能力上的考量，最终选定 TDengine 来存储海量数据。

综合来看，TDengine 满足需求的主要原因如下：

* 国产、开源的时序数据库（Time Series Database）
* 开源版支持分布式集群，方便扩展
* 列式存储，数据压缩比率高，读写性能优秀
* 一个设备一张表，对应我们一个车辆一张表，模型契合
* 超级表对于分组聚合查询的强大支持能力
  + 1. TDengine与InfluxDB性能对比

数据来源TDengine公司自测性能对比结果：<https://blog.csdn.net/taos_data/article/details/130215614?spm=1001.2101.3001.6650.2&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-2-130215614-blog-125291983.235%5Ev38%5Epc_relevant_default_base&depth_1-utm_source=distribute.>

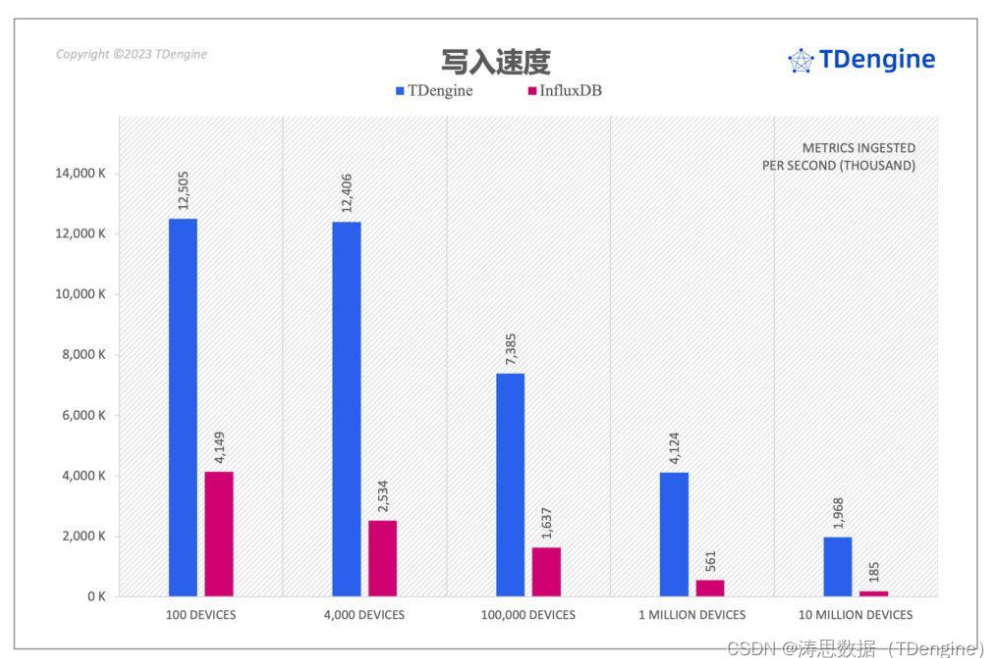
* + - 1. 不同场景下写入性能对比  
         

图2.3 TDengine与InfluxDB的写入速度对比

总体而言，在测试的五个场景中，TDengine 写入性能均优于 InfluxDB。TDengine 在场景五中写入性能是 InfluxDB 的 10.63 倍，在差距最小的场景一中也有 3.01 倍。

* + - 1. 4,000 devices × 10 metrics查询性能对比

由于部分类型（分类标准参见 TimescaleDB vs. InfluxDB 对比报告）单次查询响应时间非常短，为了更加准确地测量每个查询场景的较为稳定的响应时间，我们将单个查询运行次数提升到 5000 次，然后使用 TSBS 自动统计并输出结果，最后结果是 5000 次查询的算数平均值，使用并发客户端 Workers 数量为 8。提供场景二（4000 设备）的查询性能对比结果：

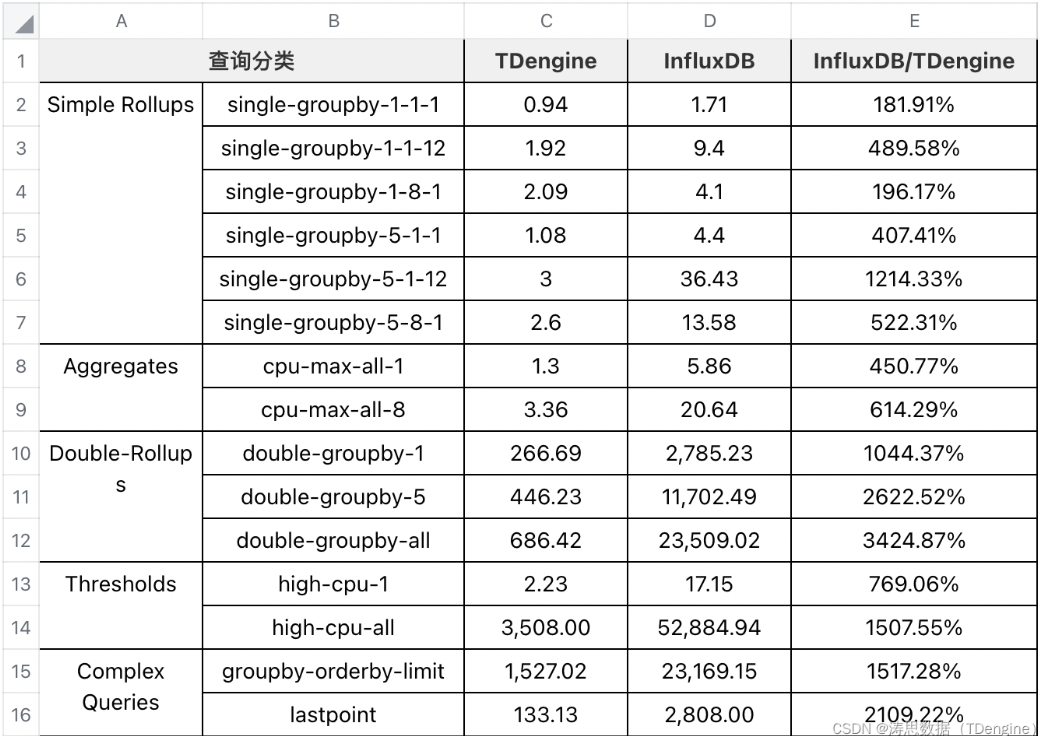


图2.4 TDengine与InfluxDB的各类查询速度对比

* + - 1. TDengine与InfluxDB的磁盘空间占比

在前面三个场景中，InfluxDB 落盘后数据文件规模与 TDengine 非常接近，但是在大数据规模的场景四和场景五中，InfluxDB 落盘后文件占用的磁盘空间显著超过了 TDengine。下图比较了 TDengine 和 InfluxDB 在不同场景下的磁盘空间占用情况：

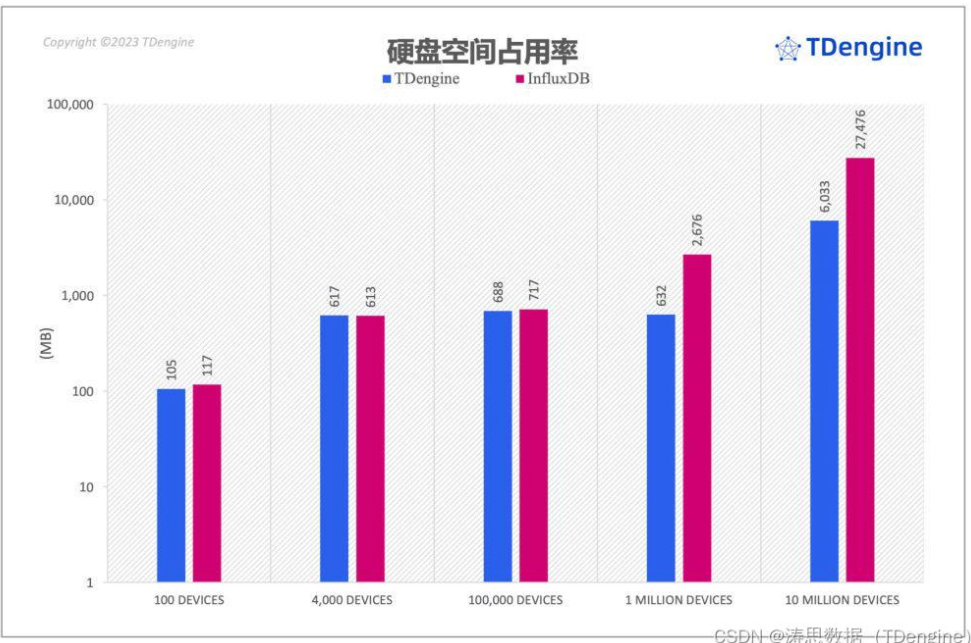


图2.5 TDengine与InfluxDB的磁盘空间占用对比

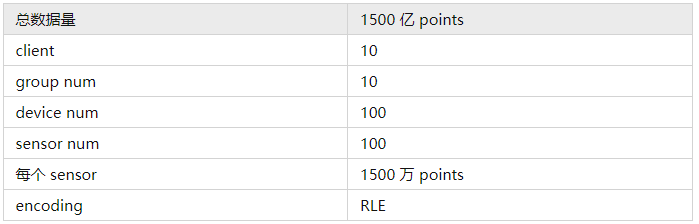
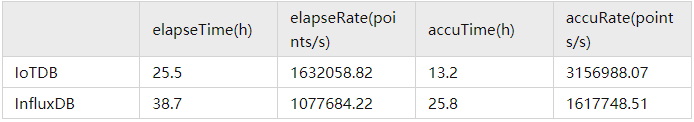
从上图可以看到，在前面三个场景中，InfluxDB 落盘后数据文件规模与 TDengine 非常接近（在场景二中，TDengine 的数据落盘规模比 InfluxDB 大了 1%）。但是在场景四和场景五中，InfluxDB 落盘后文件占用的磁盘空间分别是 TDengine 的 4.2 倍和 4.5 倍。

* 1. Apache IoTDB简介

Apache IoTDB 是用Java语言编写的, 是专为物联网时序数据打造的数据库，提供数据采集、存储、分析的功能。IoTDB 提供端云一体化的解决方案，在云端，提供高性能的数据读写以及丰富的查询能力，针对物联网场景定制高效的目录组织结构，并与 Apache Hadoop、Spark、Flink 等大数据系统无缝打通；在边缘端，提供轻量化的 TsFile 管理能力，端上的数据写到本地 TsFile，并提供一定的基础查询能力，同时支持将 TsFile 数据同步到云端。

* + 1. Apache IoTDB优点
* 国产项目，现在国家大力推行自主研发和国产化。IoTDB是清华自研时间序列数据库，Apache 孵化项目，2014年项目启动，在2018年11月18号 IoTDB 正式进入 Apache 孵化器，成为中国高校首个进入 Apache 孵化器的项目
* 为用户提供数据收集、存储和分析等特定的服务
* 轻量级的结构、高性能和可用的特性，以及与Hadoop和Spark生态的无缝集成，满足了工业IoTDB领域中海量数据存储、高吞吐量数据写入和复杂数据分析的需求
* 灵活的部署策略，IoTDB为用户提供了一个在云平台或终端设备上的一键安装工具，以及一个连接云平台和终端上的数据的数据同步工具
* 硬件成本低，IoTDB可以达到很高的磁盘存储压缩比（优于1：10无损压缩）
* 高效的目录结构，IoTDB支持智能网络设备对复杂时间序列数据结构的高效组织，同类设备对时间序列数据的组织，海量复杂时间序列数据目录的模糊搜索策略
* 高吞吐量读写，IoTDB支持数以百万计的低功耗设备的强连接数据访问、高速数据读写，适用于上述智能网络设备和混合设备
* 丰富的查询语义，IoTDB支持跨设备和测量的时间序列数据的时间对齐、时间序列字段的计算(频域转换)和时间维度的丰富聚合函数支持。
* 学习成本非常低，IoTDB支持类似sql的语言、JDBC标准API和易于使用的导入/导出工具。
* 与先进的开放源码生态系统的无缝集成，IoTDB支持分析生态系统，如Hadoop、Spark和可视化工具(如Grafana)。
* 跨平台部署，仅依靠 JDK/JRE
  + 1. Apache IoTDB缺点
* 目前只有单节点版本,不过集群版本马上要发布了
* IoTDB TsFile 的结构，目前仅有 Java 版本，资源占用方面对边缘轻量级设备不友好，限制了其在端/设备侧的应用。
* 存储上支持使用 HDFS 或 本地盘，通过使用 HDFS 来存储可保证存储层高可用，但计算层没有进一步的高可用保障。
  + 1. Apache IoTDB的客户

全球客户超过200家，包括博世、西门子、Komat'su、阿里巴巴等。

* + 1. InfluxDB和IotDB的性能测试  
       数据来源公共网页：<https://zhuanlan.zhihu.com/p/375572815>
       1. 测试配置  
          
       2. 测试结果  
          

测试共生成 1500 亿个 points，InfluxDB 总耗时 38.7 小时，IoTDB 总耗时 25.5 小时。测试 过程中，InfluxDB 和 IoTDB 都保持了平稳的插入速度，中途没有出现异常。

* + - 1. 压缩性能  
         

InfluxDB 占磁盘空间为 IoTDB 的 3.5 倍。

1. 预研时序型数据库的各参数对比

基于预研的几大类时序型数据库针对各项能力进行比对

|  |  |  |  |
| --- | --- | --- | --- |
| 时序型数据库类别 支持的能力 | InfluxDB | TDengine | Apache IoTDB |
| 排名 | 1 | 9 | 20 |
| 支持语法 | HTTP-API, | HTTP-API,JDBC | JDBC,Native-API |
| 事务 | 不支持 | 不支持 | 不支持 |
| 集群 | 付费版 | 支持 | 不支持（特定版本） |
| SQL | Flux | Standard SQL | SQL-like |
| 兼容EMQ-X | 支持桥接 | 支持桥接 | 不支持 |

1. 选型参考标准
   1. 时间序列数据库选型参考标准
      1. Data model

时间序列数据模型一般有两种，一种无schema，具有多tag的模型，还有一种name、timestamp、value型。前者适合多值模式，对复杂业务模型更适合。后者更适合单维数据模型。

* + 1. Query language

目前大部分TSDB都支持基于HTTP的SQL-like查询。

* + 1. Reliability

可用性主要体现在系统的稳定高可用上，以及数据的高可用存储上。一个优秀的系统，应该有一个优雅而高可用的架构设计，简约而稳定。

* + 1. Performance

性能是我们必须考虑的因素。当我们开始考虑更细分领域的数据存储时，除了数据模型的需求之外，很大的原因都是通用的数据库系统在性能上无法满足我们的需求。大部分时间序列库倾向写多读少场景，用户需要平衡自身的需求。下面会有一份各库的性能对比，大家可以做一个参考。

* + 1. Ecosystem

生态是我们选择一个开源组件必须认真考虑的问题。一个生态优秀的系统，使用的人多了，未被发现的坑也将少了。另外在使用中遇到问题，求助于社区，往往可以得到一些比较好的解决方案。另外好的生态，其周边边界系统将十分成熟，这让我们在对接其他系统时会有更多成熟的方案。

* + 1. Operational management

易于运维，易于操作。

* + 1. Company and support

一个系统其背后的支持公司也是比较重要的。背后有一个强大的公司或组织，这在项目可用性保证和后期维护更新上都会有较大的体验。

* 1. 比较推荐的选型
* 小而精，性能高，数据量较小(亿级): InfluxDB
* 简单，数据量不大（千万级），有联合查询、关系型数据库基础：timescales
* 数据量较大，大数据服务基础，分布式集群需求： opentsdb、KairosDB
* 分布式集群需求，OLAP实时在线分析，资源较充足：druid
* 性能极致追求，数据冷热差异大：Beringei
* 兼顾检索加载，分布式聚合计算： elsaticsearch

如果你兼具索引和时间序列的需求。那么Druid和Elasticsearch是最好的选择。其性能都不差，同时满足检索和时间序列的特性，并且都是高可用容错架构。

1. 调研总结

根据调研结果，列入待选名单的时序型数据库有：InfluxDB，TDengine和Apache IoTDB，InfluxDB是老牌龙头时序数据库，后两者都是国产的新星，且在时序数据库常用的操作场景下，如数据插入、数据查询，以及数据存储方面都有更好的表现。

就目前看来，各个时序数据库都有当下的优点和缺点，并结合当下公司的实际需求以及结合上述的参考标准，对待选的时序型数据库优缺点做个总结。

* + 1. 选用InfluxDB当下的优点：

1. 老牌时序数据库，性能或者发布版本趋于稳定，可能遇到的问题相对较少；
2. 当下公司的需求，若每5秒采集一个学生的心率数据，每天的数据量为12\*60\*24=17280，假如以小学生学制来算，则一个学生的数据总量为17280\*365\*6=37843200，对于InfluxDB来说，数据量还未达到它的亿级容量，所以性能方面暂时不存在瓶颈；
3. InfluxDB生态较好，有web端的可视化界面，方便查询或者操作数据。
   * 1. 选用InfluxDB存在的问题是：
4. InfluxDB存在使用门槛，即需要学习Flux语言以完成SQL查询，不过相对来说还是比较容易上手；
5. 它的底层存储引擎没有稳定，可能存在后续改底层架构，存在后续需要迁移数据的可能。
   * 1. 选用TDengine当下的优点：
6. 国产时序数据库，当下国内也在大力推进企业数据库国产化替代，选用该数据库可以一劳永逸；
7. TDengine的发展迅速，且在数据的插入和查询方面性能是优于InfluxDB的；
8. TDengine当下提供的功能已经满足目前实际的需要。
   * 1. 选用TDengine当下的缺点：
9. 生态相对较差，版本不是十分稳定，可能存在一些使用问题；
10. 遇到问题，寻求解决办法的途径不是很方便，需要联系公司的开发人员。
    * 1. Apache IoTDB

对于Apache IoTDB来说，如果要从TDengine和它选一个，肯定是选TDengine，不管是当前的发展趋势还是预测以后的发展趋势，TDengine都拥有更大的潜力，因为一家公司专注于做一件事情肯定有更大的优势。

* + 1. 结论

所以，综上所述，就个人观点，还是首推InfluxDB，再是TDengine。不过TDengine也是可以尝试，毕竟只有在试错中，才能成就更好的国内产品。

1. 附录

TDengine遇到的一些问题Q&A:

* FQDN域名绑定这个真的不方便，希望尽快能提供直接ip解析
* 静态数据TAG好像不是很有用处的样子，可能是我们的数据都是针对个体设备做处理
* 远程连接不稳定，默认短sql语句用udp经常会出现连接失败
* 很多时候还是需要使用索引，现在的表设计好像不支持column的索引
* SQL语法还是尽量兼容吧，文档也希望尽快完善起来
* 聚合操作希望以后能支持lambda操作，否则功能意义不大
* 版本前向兼容麻烦花点精力处理吧，强一致性有点版本没控制好的味道
* 既然已经开源了，构建一个社区比建微信群对构建生态友好度要好太多
* 我们有些按小时、按天、按月的指标计算的情况，如果当时的维表数据有误需要重算，重算的数据需要覆盖之前的数据或者把之前的数据删除掉重新写入。不过 TDengine 目前不支持数据删除，希望未来能够支持删除功能。
* 目前更新只支持更新相同时间戳数据，希望 tag 和普通字段能够联合筛选做 update；也有业务场景需要此类功能。
* 另外，目前排序还存在一定的限制，希望 tag 和普通字段能够进行自由升降序排列。