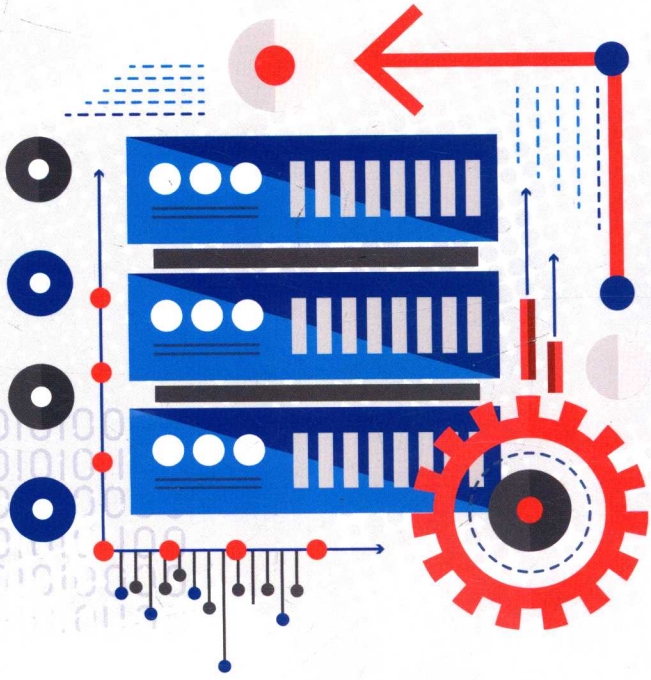
Broadview®
www.broadview.com.cn



OpenTSDB
技术内幕

百里燊 / 编著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

OpenTSDB 技术内幕

百里燊　编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

OpenTSDB 是一个分布式、可伸缩的时间序列数据库，其底层存储以 HBase 为主（这也是笔者使用的存储），当前版本也支持 Cassandra 等存储。正因为其底层存储依赖于 HBase，其写入性能和可扩展性都得到了保证。OpenTSDB 支持多 tag 维度查询，支持毫秒级的时序数据。

本书主要以 OpenTSDB 的最新版本（2.3.1 版本）为基础进行介绍。第 1 章从 OpenTSDB 的入门开始，介绍市面上多种时序数据库和云端时序数据库，OpenTSDB 的基础概念、源码环境搭建及 Grafana 的基本使用等。第 2 章主要介绍 OpenTSDB 的网络层，涉及 Java NIO 基础、Netty 基本使用，分析了 OpenTSDB 网络层的架构和实现。第 3 章介绍 OpenTSDB 中 UniqueId 组件的原理，主要讲解如何实现 UID 与字符串之间的映射。第 4 章介绍 OpenTSDB 如何实现时序数据的存储及相关优化。第 5 章介绍 OpenTSDB 如何实现时序数据的查询，其中分析了 OpenTSDB 查询中每个步骤的实现。第 6 章和第 7 章主要介绍 OpenTSDB 中的元数据及 Tree 结构的实现和功能。第 8 章主要分析 OpenTSDB 中的插件及工具类实现原理。

图书在版编目（CIP）数据

OpenTSDB 技术内幕 / 百里燊编著. —北京：电子工业出版社，2019.3
ISBN 978-7-121-36023-7

Ⅰ. ①O… Ⅱ. ①百… Ⅲ. ①云计算 Ⅳ. ①TP393.027

中国版本图书馆 CIP 数据核字（2019）第 022958 号

责任编辑：陈晓猛
印　　刷：三河市鑫金马印装有限公司
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　　印张：22.5　　字数：472.2 千字
版　　次：2019 年 3 月第 1 版
印　　次：2019 年 3 月第 1 次印刷
定　　价：79.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

前　言

OpenTSDB 是一个分布式、可伸缩的时间序列数据库，其底层存储以 HBase 为主（这也是笔者使用的存储），当前版本也支持 Cassandra 等存储。正因为其底层存储依赖于 HBase，其写入性能和可扩展性都得到了保证。OpenTSDB 支持多 tag 维度查询，支持毫秒级的时序数据。OpenTSDB 主要实现了时序数据的存储和查询功能，其自带的前端界面比较简单，笔者推荐使用强大的前端展示工具 Grafana。另外，OpenTSDB 也提供了丰富的插件接口，可以帮助开发人员对其进行扩展，在本书中也会进行详细介绍。

如何阅读本书

由于篇幅限制，本书并没有详细介绍 Java 语言的基础知识，但为便于读者理解 OpenTSDB 的设计思想和实现细节，笔者希望读者对 Java 语言的基本语法有一定的了解。

本书共 8 章，主要从源码角度深入剖析 OpenTSDB 的原理和实现。各章之间的内容相对独立，对 OpenTSDB 有一定了解的读者可以有目标地选择合适的章节开始阅读，当然也可以从第 1 章开始向后逐章阅读。本书主要以 OpenTSDB 的最新版本（2.3.1 版本）为基础进行介绍。

第 1 章介绍时序数据库的基本特征，并列举了比较热门的开源时序数据库产品及一些云厂商的时序数据库产品。接下来介绍了 OpenTSDB 的基础知识，以及 OpenTSDB 中最常用的 API，其中重点分析了 put 和 query 这两个核心接口。最后分析了 OpenTSDB 源码中提供的 AddDataExample 和 QueryExample 两个示例。

第 2 章深入分析 OpenTSDB 的网络层实现，其中介绍了 Netty 3 的基础知识，以及 OpenTSDB 网络层如何使用 Netty。另外，本章介绍了 OpenTSDB 网络层中所有的 HttpRpc 实现，重点介绍了 PutDataPointRpc 和 QueryRpc 两个 HttpRpc 实现。

第 3 章简略说明了 OpenTSDB 使用 HBase 存储时序数据的大体设计，尤其介绍了 RowKey 的设计中 UID 的原理和作用。本章具体分析了 HBase 中 tsdb-uid 表的设计，以及 UniqueId 组件管理 UID 的功能。

第 4 章主要介绍了 OpenTSDB 存储时序数据的相关组件及其具体实现。首先分析了 OpenTSDB 中存储时序数据的 TSDB 表的设计，其中涉及 RowKey 的设计、列名的格式及不同格式的列名对应的数据类型。之后又简单介绍了 OpenTSDB 中的压缩优化、追加模式及 Annotation 存储相关的内容。接下来，深入分析了 TSDB 这一核心类的关键字段、初始化过程，以及写入时序数据的具体实现。最后深入分析了 OpenTSDB 中压缩优化方面的具体实现，其中涉及 Compaction 和 CompactionQueue 两个组件的具体实现。

第 5 章主要介绍了 OpenTSDB 查询时序数据的相关组件。首先，介绍了 OpenTSDB 查询时涉及的一些基本接口类和实现类。然后，深入分析了 OpenTSDB 在查询过程中对时序数据的抽象，其中涉及 RowSeq、Span 及 SpanGroup 等组件。接下来，继续分析了 OpenTSDB 在查询时序数据的过程中涉及的其他组件。最后，分析了 TSQuery、TSSubQuery 等核心查询组件的具体实现。

第 6 章主要介绍了 OpenTSDB 中元数据的相关内容。首先，介绍了存储 TSMeta 元数据的 tsdb-meta 表的 RowKey 设计及整张 tsdb-meta 表的结构。然后，分析了 TSMeta 类的核心字段、增删改查 TSMeta 元数据的具体实现。

第 7 章主要介绍了 OpenTSDB 中 Tree（树形结构）相关的实现。首先，简单介绍了 Tree 中关键组成部分的概念及 tsdb-tree 表的结构。然后，深入剖析了 OpenTSDB 树形结构中核心组件的实现。最后，深入分析了构建一个完整 Tree 的过程。

第 8 章主要介绍了 OpenTSDB 提供的插件体系和常用工具类的实现原理。首先，介绍了 OpenTSDB 插件的公共配置及一些共性特征。然后，针对 OpenTSDB 常用的插件接口进行了介绍。接着，分析了 OpenTSDB 加载插件的大致流程。最后，详细分析了 OpenTSDB 中常用的三个工具类的实现，分别是 TextImporter、DumpSeries 及 Fsck。此外，还简单介绍了其他几个工具类的功能。

如果读者在阅读本书的过程中，发现任何不妥之处，请将您宝贵的意见和建议发送到邮箱 shen_baili @163.com，也欢迎读者朋友通过此邮箱与笔者进行交流。

致谢

感谢电子工业出版社博文视点的陈晓猛老师，是您的辛勤工作让本书的出版成为可能。同时，还要感谢许多我不知道名字的幕后工作人员为本书付出的努力！

感谢三十在技术上提供的帮助。

感谢小鱼同学，是你让我看到了星辰大海。

感谢我的母亲，谢谢您的付出和牺牲！

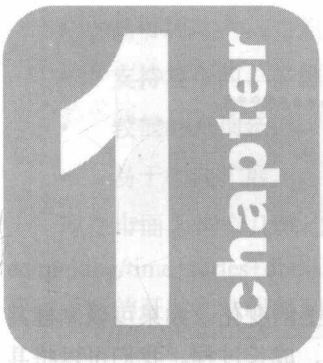
读者服务

轻松注册成为博文视点社区用户（www.broadview.com.cn），扫码直达本书页面。

- **提交勘误**：您对书中内容的修改意见可在提交勘误处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **交流互动**：在页面下方读者评论处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：http://www.broadview.com.cn/36023

目　录



第 1 章 快速入门

物联网领域的发展如火如荼，互联网企业甚至一些传统企业也在争相布局物联网。在很多物联网系统中，需要对联网的智能设备进行监控，并对监控采样得到的数据进行持久化存储，而其首选就是本章要介绍的时序数据库。

即使读者在生产实践中没有接触过时序数据库，相信对时序数据库的一些新闻也一定不陌生。例如，早在 2016 年，百度云在其物联网平台上发布了国内首个多租户的分布式时序数据库产品 TSDB，阿里云于 2017 年的 2017 云栖大会·上海峰会上发布了面向物联网场景的高性能时间序列数据库 HiTSDB 等，时序数据库作为物联网中的基础设施之一，得到了各个互联网巨头企业的重视，其热门程度可见一斑。

1.1 时序数据简介

首先来看一个简单的例子，假设我们现在关心某个 Java 程序的堆内存的使用情况，可以通过 JConsole、JMX 等多种手段获取其堆内存的使用情况，但这只是获取某个时刻的瞬时值。如果发现其堆内存使用量比较低，则可能是因为在上一时刻刚刚进行了 Full GC，如果发现其堆内存使用量比较高，也可能在下一时刻立即触发 Full GC，所以该瞬时值不能反映出任何问题。

相信读者已经想到，我们可以在一段时间内的每个时刻都记录一个瞬时值，例如一分钟记录一个值，然后将这些瞬时值按照时间顺序排列起来，就能发现该程序堆内存使用量的变化规律，从而发现一些问题。如图 1-1 所示，其中展示了该示例对应的时序数据。其实，该示例中提到的“按照时间顺序排列起来的瞬时值”就是一条时序数据，这里为“时序数据”下个简单的定义：“时序数据”（即“时间序列数据”）是同一指标按照时间顺序记录的一组数据，示例中

的“指标”（metric）就是堆内存大小。

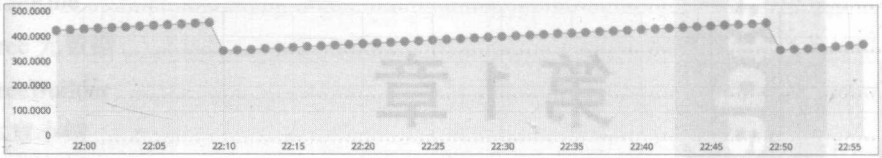


图 1-1

从更加宏观的角度看，时序数据可以描述一个物体在时间维度上的变化，如果可以掌握其关键指标的时序数据，并加以分析，就可以掌握该物体的变化规律、成长过程。我们可以将其具体化到生活中的一些细节上，例如股票中的日线图、周线图、月线图，它们表示的就是股价随时间发生的变化，很多操盘手通过分析这些时序数据进行交易。

随着大数据时代的到来，时序数据量也发生了爆发式的增长，使用传统的关系型数据库，例如 MySQL、Oracle 等，已经很难满足时序数据在存储、分析、展示等方面的需求，为了解决这些问题，市面上出现了很多时序数据库产品，其中有完全开源的产品，也有闭源的商业付费产品，本书的主角——OpenTSDB 就是一款完全开源的时序数据库产品。

通过前面对时序数据的简单描述，相信读者会发现时序数据的一些特点，这也是传统关系型数据库不好解决，而时序数据库需要解决的几个关键点：

（1）时序数据的写入比较稳定。普通应用的数据量一般与请求的 QPS 成正比，但对于时序数据来说，QPS 是稳定的，即每个固定的时间间隔都会收到相应的时序数据。

（2）只写入较近时间的数据。时序数据是随着时间推移而不断产生的，所以时序数据数据库收到的写入请求一般都是近期的数据，即使有少许延迟，也不会很大。

（3）没有更新操作。一般情况下，当一个指标在某个时刻的指标产生后，对其进行更新是没有意义的。

（4）按照时间范围进行查询，且近期数据被查询的概率更高。

（5）多维度的分析查询。在前文的示例中，只涉及一个 JVM 实例，但在实际生产中，每个应用都会涉及多个 JVM 实例，企业级的应用可能会涉及成千上万的 JVM 实例，此时需要从多个维度（例如，不同的机房、不同的功能、不同的业务线）去分析 JVM 之间的相互影响。

1.2 时序数据库

结合前文介绍的时序数据的特点，可以得出几条对时序数据库的基本要求：

- 支持高并发、高吞吐的写入。
- 支撑海量数据的存储。

- 高可用。
- 支持复杂的、多维度的查询。
- 较低的查询延迟。
- 易于横向扩展。

现在市面上也有几款比较成熟的时序数据库产品，如图 1-2 所示（https://db-engines.com/en/ranking/time+series+dbms）。这些产品都是根据不同的应用场景，关注了上述一个或几个点，经过不断的开发和迭代得到的。

26 systems in ranking, September 2018

Rank			DBMS	Database Model	Score		
Sep 2018	Aug 2018	Sep 2017			Sep 2018	Aug 2018	Sep 2017
1.	1.	1.	InfluxDB	Time Series DBMS	11.79	+0.23	+3.31
2.	2.	↑5.	Kdb+	Multi-model	3.87	+0.36	+2.10
3.	3.	3.	Graphite	Time Series DBMS	2.70	+0.10	+0.14
4.	4.	↓2.	RRDtool	Time Series DBMS	2.55	+0.08	-0.51
5.	↑6.	↓4.	OpenTSDB	Time Series DBMS	1.79	+0.38	-0.10
6.	↓5.	↑7.	Prometheus	Time Series DBMS	1.59	+0.07	+0.92
7.	7.	↓6.	Druid	Time Series DBMS	1.20	+0.02	+0.22
8.	8.	8.	KairosDB	Time Series DBMS	0.53	+0.04	+0.03

图 1-2

InfluxDB 是由 Golang 语言编写而成的，也是 Golang 社区中比较著名的一个产品，在很多 Go 语言的讲座和文章中，都会将 InfluxDB 作为示例产品进行简单介绍。在时序数据库范畴里，其知名度也非常高。InfluxDB 提供了无结构化（schemaless）的存储、高效的压缩存储算法、方便的查询语言、实时的数据采样等功能。另外，可以利用 InfluxDB 搭建可扩展的集群。InfluxDB 中还内置了用户管理和角色管理的功能，这在很多 TSDB 产品中都是不具备的，需要开发人员进行扩展支持。

如果读者准备试用一下 InfluxDB，则希望读者参考其官方文档，因为 InfluxDB 不同版本之间的差异较大，对于最新版本来说，网络上很多资料没有参考价值。

KDB+是一个商业产品，并没有开源，不过官方提供了 32 位和 64 位两个版本的试用产品，这两个试用产品也有颇多限制。例如，64 位的版本需要网络在线才能使用。KDB+是一个列式时序列数据库，其自定义了一种叫作“q”的查询语言，该查询语言非常简短、灵活。KDB+速度也比较快，可以轻松支持 TB 级别的数据量。

Graphite 创立于 2006 年，算是比较老牌的时序数据库产品了。Graphite 可以部署成分布式模式，方便横向扩展。Graphite 主要完成了时序数据存储和查询的功能，虽然没有提供数据采集功能，但支持很多第三方插件。经过多年的积累和发展，Graphite 已经可以提供丰富的函数支持，这也是其受到广大用户青睐的原因之一。

RRDTool 的全称是 Round-Robin Database Tool，从名称也能看出来，其采用固定大小的空间来存储时序数据，其中设定了一个指针，随数据的读写移动。相较于其他时序数据库产品，RRDTool 不仅实现了数据的存储，还提供了丰富的工具来绘制图表，丰富的画图功能使其从其他时序数据库产品中脱颖而出。

OpenTSDB 是一个分布式、可伸缩的时间序列数据库，其底层存储以 HBase 为主（这也是笔者使用的存储方式），当前版本也支持 Cassandra 等存储。正因为其底层存储依赖于 HBase，其写入性能、可扩展性都得到了保证。OpenTSDB 支持多 tag 维度查询，支持毫秒级的时序数据。OpenTSDB 主要实现了时序数据的存储和查询，其自带的前端界面比较简单，后面笔者会推荐一个比较强大的前端工具。另外，OpenTSDB 也提供了丰富的插件接口，可以帮助开发人员对其进行扩展。

Prometheus 由 SoundCloud 平台于 2012 年开发，其使用的主要语言是 Golang。Prometheus 是一个开源的监控系统，也是一个高性能的时序列数据库。Prometheus 采用了与 OpenTSDB 中 tag 类似的维度机制，如果读者了解 OpenTSDB，那么学习 Prometheus 也会比较简单。

HiTSDB 是阿里云开发的一套时序数据库系统，并没有相应的开源版本，其官方文档自称具有如下特点。

- 高并发写入：千万级数据秒级写入。
- 高效读取：百万数据点秒级读取。
- 低成本存储：高压缩算法优化，每个数据点平均占 2 个字节。
- 数据计算分析、数据可视化。

有消息称，HiTSDB 已经在阿里内部孵化多年，在阿里集团内部已经支持了 20 多个核心业务场景，例如阿里智慧园区的 Iot 建设。

CTSDB 是腾讯云的时序数据库产品，主打高效、安全、易用的特点。根据其官方文档，CTSDB 使用批量接口写入数据，降低网络开销。CTSDB 的写入策略是，先将时序数据写入内存，然后周期性“dump”成不可变文件，同时生成倒排索引，加速各个维度的查询，号称千万数据秒级可查。CTSDB 同样提供了历史数据聚合、数据过期清理等功能来降低存储成本。CTSDB 还提供了丰富的 RESTful API 接口，同时兼容 Elastic Search 的访问协议。横向扩展、数据自动均衡也是 CTSDB 的特性之一。

百度“天工”时序数据库是百度云提供的时序数据库，其官方文档号称千万数据点秒级写入，亿级数据点秒级查询。同时提供了数据过期自动删除、聚合等功能，支持 SQL 语句、支持与 Hadoop/Spark 大数据平台对接，还提供了丰富的 RESTful API。该产品的另外一个亮点就是三副本、分布式部署，保证了数据的可靠性。

至此，比较常见的时序数据库产品就介绍完了，读者可以根据自己的使用场景和各个时序

数据库产品的特性，选择一款产品进行深入的了解。当然，笔者还是强烈推介 OpenTSDB 的，尤其是了解 Java 语言的读者，经过本书后续的分析，相信读者能够完全了解 OpenTSDB 的实现原理。

1.3 快速入门

介绍完时下比较成熟的时序数据库产品之后，本节将带领读者快速了解 OpenTSDB。首先介绍 OpenTSDB 涉及的基础知识，然后搭建 OpenTSDB 的源码环境，接着简单介绍 OpenTSDB 中最常用的 API，最后介绍 OpenTSDB 源码中提供的 AddDataExample 和 QueryExample 两个示例，让读者初步了解 OpenTSDB 读写的大致过程，为后面深入分析 OpenTSDB 的实现打下基础。

1.3.1 基础知识

正如前文介绍的那样，时序数据库的主要功能就是管理时序数据，而一条时序数据则是由多个“点”（DataPoint）构成的。在 OpenTSDB 中，与时序数据息息相关的四个概念如下。

- **metric**：时序数据的指标名称，比如前文示例中提到的“堆内存的大小”。在 OpenTSDB 中一般不会用中文作为指标名称，而是使用一个更加简短的、类似于变量的名称，例如 JVM_Heap_Memory_Usage_MB。
- **timestamp**：表示一条时序数据中点对应的具体时间，可以是秒级或毫秒级的 UNIX 时间戳。
- **tags**：一个或多个标签（tag）组合，主要用于描述 metric 的不同维度。一个 tag 由 tagk 和 tagv 组成，tagk 指定了某个维度，tagv 则是该维度下的某个值。
- **value**：表示一条时序数据中某个 timestamp 对应的那个点的值。

除了 OpenTSDB，还有很多其他的时序数据库也有类似的概念，所以学习 OpenTSDB 之后就可以更快地上手其他时序数据库。

从图 1-3 中可以更加直观地看到 metric、timestamp、tags、value 与时序数据之间的关系，这里依然沿用前文对 JVM 堆内存的示例。

在图 1-3 中只展示了一个 tag（tagk=host，tagv=127.0.0.1），随着业务的不断发展，可能会使用多台服务器，每台服务器上部署多个 JVM 实例。要记录这些 JVM 的堆内存使用量，就会产生多条时序数据，它们有相同的 metric（即 JVM_Heap_Memory_Usage_MB），但是 host 的 tagv 值会因为所在机器的不同而有所不同。另外，还需要一个额外的 tag 来区分同一台机器中的不同 JVM 实例（这里使用 instanceId），这样就能获取多条时序数据。如图 1-4 所示，metric 都是 JVM_Heap_Memory_Usage_MB，但两条时序数据的维度不同，其中 host 这个 tagk 表示服务端

的维度，instanceId 这个 tagk 表示 JVM 实例的维度。

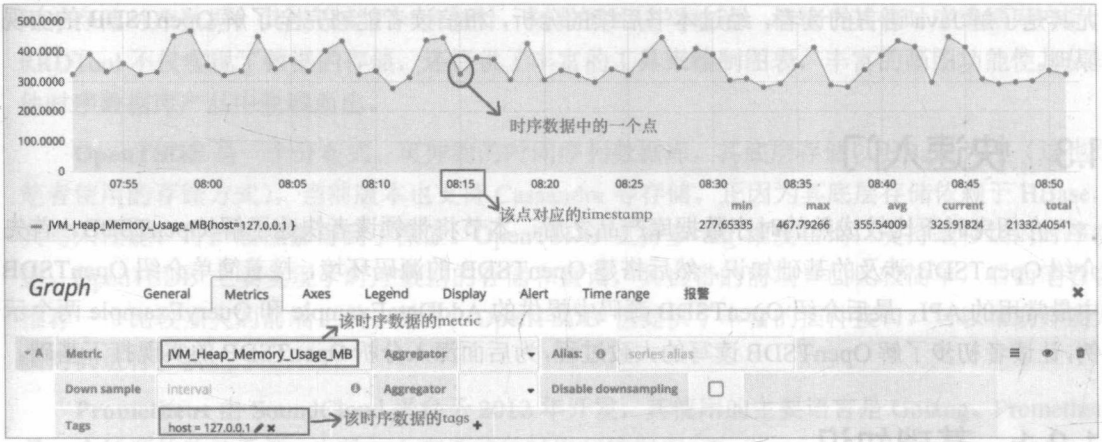


图 1-3

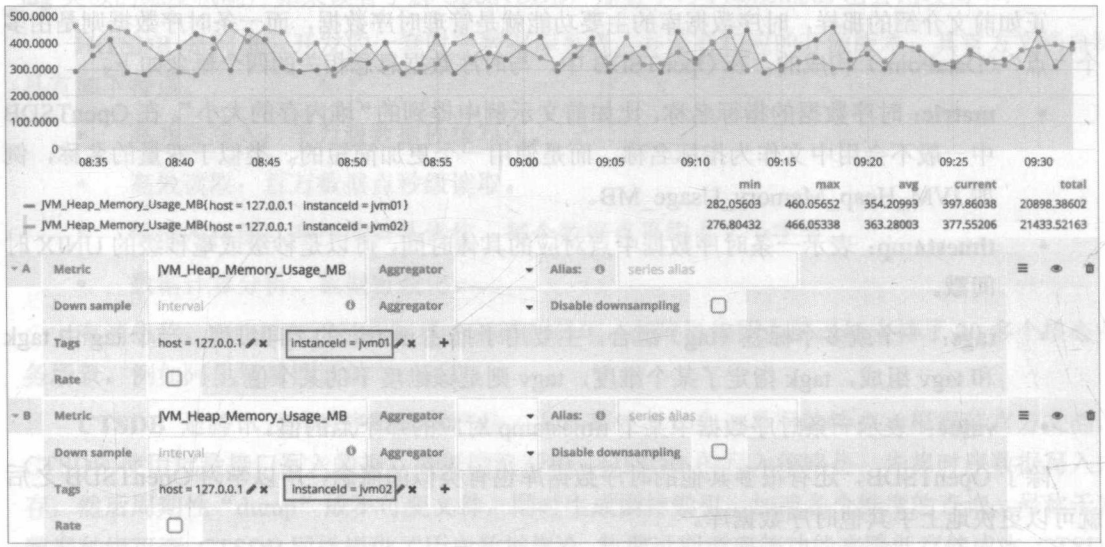


图 1-4

使用 tag 的方式来标识不同维度还有另一个好处，那就是方便聚合。在 OpenTSDB 中，如果要查询 host=127.0.0.1 这台机器上全部 JVM 实例的堆内存聚合值，那么只需要给出{metric=JVM_Heap_Memory_Usage_MB，host=127.0.0.1 }这些信息及具体的聚合方式（例如 SUM 聚合方式）即可。

除了聚合，OpenTSDB 还提供了 Downsampling 功能，在后面介绍其具体实现时再进行详细讲解。

1.3.2 HBase 简介

OpenTSDB 2.3 版本可以支持多种底层存储，例如 HBase、Cassandra 等，其中 HBase 是 OpenTSDB 默认支持的后端存储，本书也将以 HBase 作为 OpenTSDB 的底层存储来进行介绍。

HBase 是一款分布式列存储系统，其底层依赖 HDFS 分布式文件系统。HBase 是 Apache Hadoop 生态系统中的重要一员，其架构是参考 Google BigTable 模型开发的，本质上是一个典型的 KV 存储，适用于海量结构化数据的存储。HBase 相较于传统的关系型数据库有如下优点：

- HBase 是集群部署的，横向扩展方便。
- HBase 的容错性较高，这也是得益于其集群部署的特点，并且相同的数据会复制多份，存放到不同的节点上。
- 相同硬件条件下，HBase 支持的数据量级远超传统关系型数据库。
- HBase 的吞吐量较高，尤其是写入能力，远超传统关系型数据库。

当然，HBase 也有不适用的场景，例如：

- 需要全面事务支持的场景。传统数据库支持多行、多表的事务，而 HBase 只支持单行的事务。
- 传统关系型数据支持 SQL 语句的查询方式，非常灵活，而 HBase 只能通过 RowKey 进行查询或扫描。

具体在哪种场景下应该选用哪种存储，读者可以根据存储的具体特性是否能满足业务的具体要求来决定。

从逻辑上看，HBase 将数据按照表、行和列的形式进行存储，如表 1-1 所示，HBase 表中存储的数据可以非常稀疏。HBase 表中可以有多个列族（Column Family），列族需要在建表时明确指定，且后续不能自动增加。一个列族下面可以有多个列（Column），列的个数不需要在建表时指定，可以随时添加。另外，HBase 表中的数据是按照 RowKey 进行排列的。HBase 表中行和列的交叉点称为 Cell，HBase 会记录每个 Cell 的版本号（Version Number），默认值是 UNIX 时间戳，可以由用户自定义。

表 1-1

RowKey	Family1		Family2			Family3
	col1	col2	col1	col2	col3	col1
rowkey1	value1				value4	
rowkey2		value5		value2		value6
rowkey3			value7			value3