第 1 章
时序数据库 InfluxDB 简介

作为目前主流的时序数据库之一，InfluxDB 已经拥有了大量的拥护者，作为开发人员，必须了解以 InfluxDB 为首的时序数据库是怎样对数据进行处理的。

本章将从数据简史出发，首先介绍数据以及时序数据的概念。再过渡到时序数据库的概念及其发展，进而介绍本书的重点——InfluxDB 时序数据库。在 InfluxDB 的讲解中，先带领读者了解 InfluxDB 的基本概念及其发展，再介绍其独特的设计思想和核心特性，最后是 InfluxDB 在实际场景中的使用。通过本章的学习，将掌握以下的知识：

- 了解什么是时序数据。
- 时序数据库 InfluxDB 的基本概念、设计理念及其主要特性。
- 时序数据库 InfluxDB 在各个场景的应用。

1.1 数据简史

数据（data）是事实或观察的结果，是对客观事物的逻辑归纳，是用于表示客观事物的未经加工的原始素材。数据可以是连续的值，例如音频、图像，称为模拟数据；也可以是离散的，如符号、文字，称为数字数据。显然，在数据库领域，连续型数据和离散型数据，都需要研究。

这些年，无论是学术界还是大众媒体，都在谈论大数据，各企业也都意识到数据的重要性，但是却很少有技术能从纷繁的数据“金矿”中，挖掘出真正对生活、生产有帮助的数据。

全球数据量正在爆炸性地增长，据国际数据公司统计，到 2025 年，全球数据量将达到 175ZB。ZB 是什么概念呢？ ZB 是泽字节，是 2^{70} 字节，大家熟悉的 TB 是 2^{40} 字节，IZB=2^{30}TB，以 1TB 的硬盘计算，存储 175ZB 数据需要购买 175×2^{30} 个硬盘。2012—2025 年全球数据规模增长趋势如图 1-1 所示。

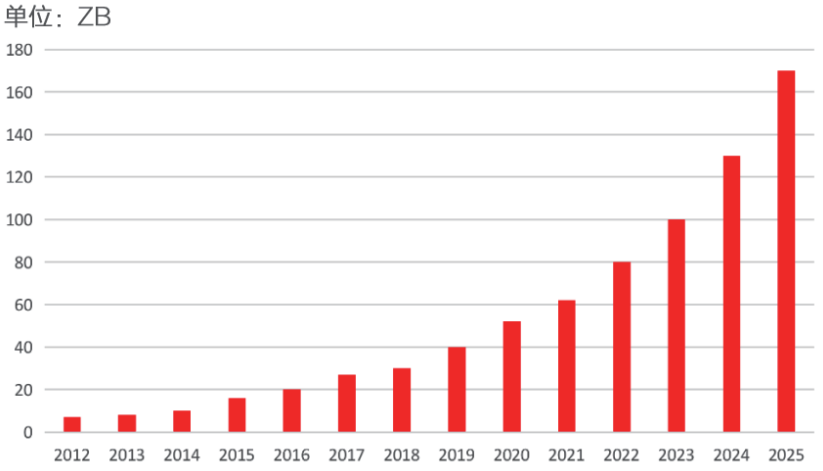

图 1-1 数据规模增长趋势

诙谐点说，我们不知道未来的数据会有多值钱，但是假设 2025 年 1 个 1TB 的硬盘为 200 元，那么单是卖硬盘，就可以营收 175×2^{30}×200 元，约为 35 万亿元，这是一个天文数字。

这些数据其中一部分是时序数据，关于时序数据的概念，后面会详细介绍。随着物联网技术的发展，预计到 2025 年，全球将会有 1500 亿台设备联网，这些设备中，大多数会产生以时间为中心的时序数据。例如，很快就会普及的自动驾驶汽车，在行驶过程中，全车的传感器会产生大量的基于时间的测点数据，这些数据会被计算，从而影响自动驾驶的策略。这些数据中一部分数据处理后会被抛弃，另一部分会被存储下来，进行深度分析。

2017 年，时序数据占全部数据的 15%，到 2025 年将接近 30%，达到 52.5ZB。无法想象这些数据有多庞大，但是能预测需要很强大的时序数据存储系统，才能将这些数据有序地存储下来，并进行处理分析。过去的 20 年，可能每一位 IT 从业者，都需要了解并使用类似 MySQL、Oracle、SQL Server 的关系型数据库，但是未来 20 年，每一位 IT 从业者，都有必要学习时序数据库的知识。

1.1.1　什么是时序数据

不知道大家是否思考过这样一个问题：在日常生活生产过程中，需要对某一些数据按照时间的变化进行统计，例如，在气象监测领域，数据时刻在变换，工作人员要将这些数据的变化记录下来用作监控分析，那么该如何表示这些数据呢？其实为每一条数据记录加上时间戳即可以表示该条数据产生的时间。

简单来说，就是这类数据描述了被测量的主体在时间范围内每个时间节点上的测量值，用于描述物体在历史时间维度上的状态变化信息。而对于时序数据的分析，就是尝试把控其变化规律的过程，按照以上过程的定义，可以得知，时序数据有以下特点：

- 是按照时间维度索引的数据。
- 是一段时间内的测量值，测量值可以为多种类型。
- 用来监控状态变化信息。
- 数据量大，数据量随时间增加。

了解时序数据之后，想必各位读者对时序数据库也有了自己的理解。顾名思义，时序数据库的作用就是为了处理时序数据。其发展至今，经过市场的不断淘汰更新，一批稳定的时序数据库占据了市场主流，下面带大家了解一下时序数据库的发展历史。

1.1.2　时序数据库的历史

笼统地说，时序数据库又叫实时数据库，但是实时数据库包含时序数据库，由于实时数据库和时序数据库的核心共性都是研究时间数据的关系，所以本书，对实时数据库和时序数据库没有进行区分。

实时数据库概念诞生于 20 世纪 70 年代，80 年代进入理论研究，90 年代诞生商用产品。相比于国外，国内 90 年代才开始进行理论研究，中国无论是从理论还是实践上都起步较晚。世界时序数据库发展的历史如图 1-2 所示。

国内对时序数据库的研究虽然起步较晚，但是近几年发展却非常迅猛，以 IoTDB 为代表的时序数据库，已经在某些底层内核、性能、功能上超过了国外先进的时序数据库，拥有自主、可控、安全的国产时序数据库已经成为现实。

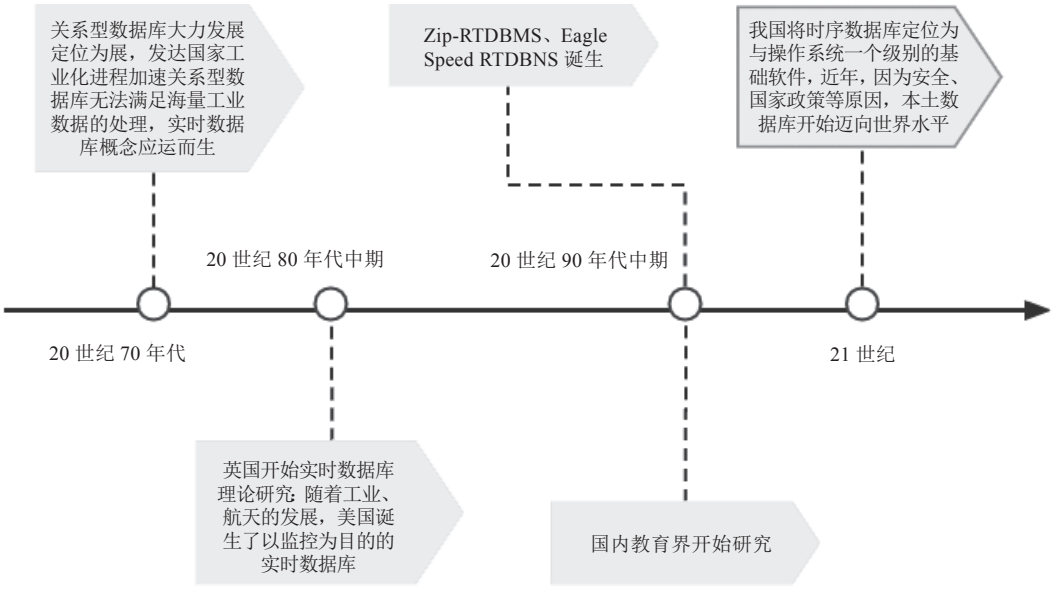

图 1-2　世界时序数据库发展历史

1.2　InfluxDB 简介

时序数据库发展到目前，市场上已经诞生了很多稳定高效的产品，而 InfluxDB 就是其代表之一。

InfluxDB 作为目前流行的时序数据库，拥有众多的竞争对手。本书将重点介绍 InfluxDB，只要大家学会了 InfluxDB，其他同类的时序数据库也就学会了，因为目前大多数时序数据库的概念都是相通的。

1.2.1　什么是 InfluxDB

InfluxDB 是一个由 InfluxData 公司于 2013 年开发的开源分布式时间序列数据库，其设计意图就是为了能够存储带有大量时间戳的数据，例如物联网设备、自动驾驶汽车产生的数据等，InfluxDB 致力于对这些数据进行海量的写入以及高负载查询，由于是由 Go 语言开发，无须外部依赖，安装配置十分便捷，被广泛用于存储系统的监控数据等领域，成为目前主流的时序数据库之一。

InfluxData 公司成立于 2012 年 1 月，其创始人 Paul Dix 和 Todd Person 在 2013 年开始着手 InfluxDB 的开发，之后获得了多轮投资。发展至今，历经了多次迭代，目前在 InfluxDB 版本中，部署方式分为单机版和集群版，单机版走开源路线。在 2016 年 3 月，InfluxData 公司宣布他们会将用于支撑 InfluxDB 集群水平扩展的组件作为闭源产品单独

销售，从而为 InfluxDB 的持续开发建立一个稳定的收入来源。

InfluxDB 单机版是免费使用的，关于 InfluxDB 集群版的售价可在其官网上查阅，价格是随着服务器节点和每个节点的核心数变化的，见表 1-1。

表 1-1　InfluxDB 售价

数据节点数	每个数据节点的核心数	年度订阅价格	每秒写入的数据点数
2	2	9500 美元	75000
2	4	17000 美元	150000
超过两个	超过四个	联系 InfluxDB	>150000

1.2.2　InfluxDB 的历史

随着工业的快速发展，传统的关系型数据库面对快速增长的时序数据开始显得逐渐吃力，由于采用 BTree 的随机读写的模式，在寻道上会消耗很多时间，在目前物联网数据的大量写入要求下，传统关系型数据库的效率太过低下，于是时序数据库应运而生。

从图 1-3 中可以看出，时序数据库最早可追溯到 20 世纪 90 年代，为满足监控领域时序数据的存储要求，以 RRDTool 和 KDB+ 为代表，出现了第一批时序数据库，它们将一个固定大小的数据库内嵌在监控系统中，来达到随时间变化快速存储数据的能力，但其缺点是读取能力依旧较弱，并且处理的数据比较单一。

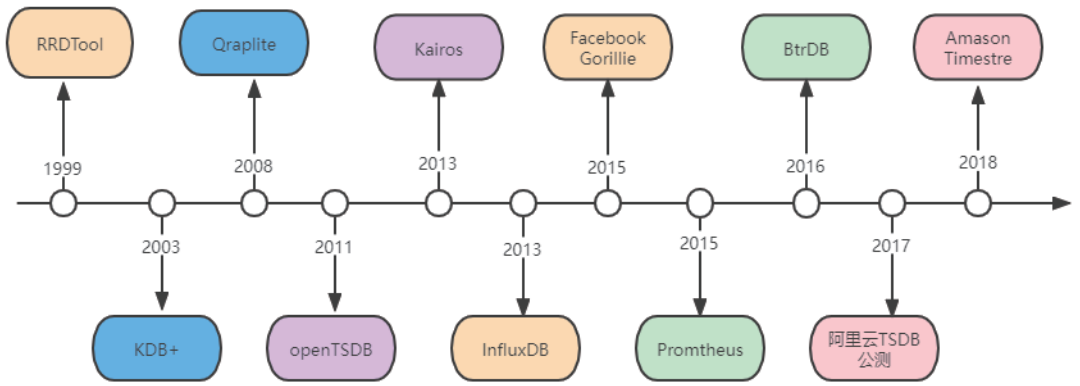

图 1-3　时序数据库发展史

后来随着大数据发展，不只是监控领域，其他领域也出现了需要处理大量时序数据的需求。从 2011 年年始，陆续出现了以 openTSDB、Kairos 为代表的基于分布式存储的数据库，其对时间进行了针对性优化。例如，openTSDB 的底层依赖 HBase 集群存储，根据时序的特征对数据进行压缩，节省了存储空间。相较于之前的时序数据库，在存储和读写性能方面上有了显著提升，但其依赖 Hadoop 和 HBase 环境，使得部署及维护成本极高。

由于 openTSDB 本身的不足再加上部署维护不便，促成了低成本时序数据库的诞生。在这场混战中，InfluxDB 得益于其高效的数据读取存储的能力和算法，慢慢地占据了主流市场。

Errplane 公司在 2013 年下半年开始以开源项目的形式开始了 InfluxDB 的研发。其目的是提供一个高性能的监控以及告警的解决方案。2014 年 11 月，Errplane 公司获得了梅菲尔德风险投资公司与 Trinity Ventures 领投的 A 轮投资，金额高达 810 万美元。在 2015 年，Errplane 正式更名为 InfluxData，在 2016 年 9 月获得了金额高达 1600 万美元的 B 轮投资，又于 2018 年 2 月获得 3500 万美元的 C 轮投资。

1.2.3 InfluxDB 现状及未来

2013 年 9 月，Errplane 公司正式发行了 InfluxDB 1.0 版本。2019 年 1 月 23 日，InfluxDB 推出了 InfluxDB 2.x 的 alpha 内部测试版，通过了二十几个版本的迭代，到 2020 年 1 月 8 日，InfluxDB 2.0 开始推出 Beta 公开测试版，一直持续到同年 10 月，最终测试版本 v2.0.0-rc.0 正式推出。后来又推出了 v2.0.1 通用版，截至 2021 年 11 月，InfluxDB 版本已经迭代到了 v2.1.1。

历经这么多版本的迭代，InfluxDB 在用户界面上不断进行完善，对旧版本的 bug 进行不断改进，并在每个版本上增加了新功能。以最新版本为例，InfluxDB 对 api 和 cli 做出了更新，同时包括 InfluxQL 的更新。在最新的 DB-ENGINES 给出的时间序列数据库的排名中，InfluxDB 高居第一位，可以预见，InfluxDB 会得到越来越广泛的使用，如图 1-4 所示。

不过值得注意的是，InfluxDB 目前正在进行重构，相信不久之后就会推出更为强劲的版本。

Rank			DBMS	Database Model	Score		
Mar 2021	Feb 2021	Mar 2020			Mar 2021	Feb 2021	Mar 2020
1.	1.	1.	InfluxDB	Time Series, Multi-model	26.87	+0.62	+4.44
2.	2.	2.	Kdb+	Time Series, Multi-model	7.76	-0.02	+2.41
3.	3.	3.	Prometheus	Time Series	5.90	+0.15	+1.75
4.	4.	4.	Graphite	Time Series	4.67	+0.06	+1.23
5.	5.	5.	RRDtool	Time Series	2.90	-0.10	+0.20
6.	6.	↑ 7.	TimescaleDB	Time Series, Multi-model	2.82	-0.04	+0.94
7.	7.	↑ 8.	Apache Druid	Multi-model	2.69	+0.03	+0.84
8.	↑ 9.	↑ 9.	Fauna	Multi-model	1.83	-0.07	+0.89
9.	↓ 8.	↓ 6.	OpenTSDB	Time Series	1.77	-0.26	-0.21
10.	10.	↑ 11.	GridDB	Time Series, Multi-model	0.97	+0.15	+0.52
11.	11.	↑ 15.	DolphinDB	Time Series	0.83	+0.02	+0.52
12.	12.	↓ 10.	KairosDB	Time Series	0.73	-0.01	+0.18

图 1-4　时序数据库排名

1.3　InfluxDB 的基本概念

InfluxDB 主要有七个概念：database、measurement、timestamp、filed、tags、point、series。下面简单介绍一下各个概念的含义，更详细的概念将在后面的章节介绍。

- database：数据库。用户、保留策略、连续查询和时间序列数据的逻辑容器。
- measurement：相当于关系型数据库中表的概念。可以理解为一条条记录都是存储于 measurement 中的，一个数据库中可以有多个 measurement，一个 measurement 中可以存储很多的数据。
- timestamp：时间戳。与 point 关联的日期和时间，每条记录都会带有一个单独的时间戳。
- filed：未加索引的字段，用来存储具体的时序数据，即随着时间变化而变化的数据。是 InfluxDB 数据结构中记录元数据和实际数据的键值对，用来保存真实数据的结构，由 key-value 键值对组成，在 InfluxDB 的数据结构中是必需的。
- tags：标签，一般用于存储标识数据来源的属性信息。是记录元数据的键值对，在 InfluxDB 的数据结构中是可选的，tags 和 filed 一样是 key-value 的结构，但会在 tags 上加索引，因此对 tags 的查询是高效的。
- point：一条记录。相当于关系数据库中的一条记录。
- series：InfluxDB 中一些数据的集合。在同一个 database 中，retention policy、measurement、tag 完全相同的数据属于一个 series，同一个 series 的数据在物理上会按照时间顺序排列存储在一起。

为了更好地理解这些概念，表 1-2 将 InfluxDB 与 MySQL 的关键概念做了对比。

表 1-2　InfluxDB 与 MySQL 概念对比

术语	InfluxDB 概念	MySQL 概念
database	数据库	数据库
measurement	InfluxDB 中的表	数据库表
timestamp	InfluxDB 时序数据库特有属性，时间戳	无
filed	未加索引的字段	列名
tags	标签	加索引的字段
point	一条记录	一行数据

1.4　InfluxDB 的设计理念

InfluxDB 是一个时序数据库，为了最大程度发挥时序数据库高写入高查询的功能，不得不对其设计优化做出一些权衡，主要是牺牲部分功能提升查询和写入的性能，其主要设计理念如下：

- 如果在同一时间点有相同的数据被写入，就会被认为是重复写入，这样设计的目的是解决数据重复的冲突，但在极少数情况下，可能会发生数据丢失的情况。
- 为了提高查询和写入的性能，InfluxDB 极少出现删除数据的情况，即使要删除数据，也基本是清理过期数据，由于这一缺陷，其删除功能受到了很大限制。
- InfluxDB 极少更新已有的数据，且不会出现有争议的更新，时间序列中的数据总是新数据，这样做的目的是限制对更新的访问以提高查询和写入的功能，所以其更新功能是受到限制的。
- 绝大多数写入是针对时间戳最近的数据，并且数据按时间升序添加。这样使得添加数据的性能明显更高，但如果使用随机时间或者不按升序时间写入，写入性能就会降低。
- InfluxDB 处理的数据规模会非常大，它的出现就是为了解决大量数据的快速写入和查询的需求。必须能够处理大量的读写操作，为了解决大量数据的快速写入和查询的需求，InfluxDB 团队为了满足需求不得不做出权衡以提高其性能。
- 能够写入和查询数据会比强一致性更重要，这样做使得在对数据库进行查询插入操作的时候，多个客户端可以在高负载的情况下完成，但如果数据库负载过重，查询返回的结果可能不包括最近的 point。

1.5 InfluxDB 的核心特性

得益于 InfluxDB 的设计理念，InfluxDB 造就了如下特殊的核心特性。

1.5.1 内置 REST 接口

InfluxDB 原生操作模式分为两种：一种是用户可以通过命令行的方式访问 Influx 服务，对其操作较为方便；另一种就是 InfluxDB 内置的 REST 接口，用户可以启动 Influx 服务，并连接到 InfluxDB 服务器进而执行操作。

InfluxDB API 的一大特性就是可编程性强，而且编程语言较为友好，开发者可以快速上手。InfluxDB 提供了一系列 HTTP API 供开发者调用，使用 HTTP 响应代码、HTTP 身份验证，并以 JSON 格式返回响应。

InfluxDB 提供的各种语言的 HTTP API 接口的封装可具体参考：https://docs.influxdata.com/influxdb/v2.0/api/。

1.5.2 数据 Tag 标记

Tag 标记是 InfluxDB 的一个独特概念，其作用一般是为了创建索引，提高查询性能，是 InfluxDB 数据结构中记录元数据的键值对，在 InfluxDB 数据结构中属于可选部分，

但它们对于存储常用查询的元数据很有用。

Tag 标记一般存放的是标识数据点来源的属性信息，以代码清单 1-1 为例，其中 host、monitor_name 就是标签键，host 对应的值是 127.0.0.1，monitor_name 对应的值是 test1。

代码清单 1-1

```
> insert test,host=127.0.0.1,monitor_name=test1, count=2,num=3
> select * from test
name: test
time                    count    host       monitor_name  num
----                    -----    ----       ------------  ---
1585897703920290000      1      127.0.0.1     test1         3
1585897983909417000      2      127.0.0.1     test1         3
1585898383503216000      2      127.0.0.1     test1         3
```

1.5.3　类 SQL 的查询语句

InfluxDB 的另一大优点是拥有类似 SQL 的查询语言——InfluxQL，这使得使用传统 SQL 开发的从业人员能够快速上手，InfluxQL 语言经过仔细设计，目前已发展得较为成熟。

下面以 SELECT 语句举例，在特定条件下从指定表名中查询相关数据。InfluxQL 语句遵循 SQL SELECT 语句的形式：

SELECT <stuff> FROM <measurement_name> WHERE <some_conditions>

上述语句的参数解释如下：

- stuff：要查询的内容。
- measurement_name：表名，表示要从哪一张表中查询信息。
- some_conditions：条件约束，指定在该条件下查询内容。

InfluxQL 还支持表达式的算术运算、正则表达式、SHOW 语句、GROUP BY 语句等，同时也支持 SQL 的大部分函数，如 COUNT()、MIN()、MAX() 等，有关 InfluxDB 函数的更多内容，读者可参考本书第 5 章。

1.5.4　高性能

时序数据库的一个重要特性就是高性能，这是支撑其处理大量数据的重要因素，决定了数据库在高写入高存储时的表现。InfluxDB 在这方面表现得很优秀，在同类别的时序数据库中，InfluxDB 的查询性能测试都领先于其他数据库数倍；在内置函数测试方面，其性能也是其他数据库的数倍。

要做到如此高性能，首先需要解决高写入吞吐量带来的问题，InfluxDB 一开始选择了 LevelDB 作为存储引擎。然而，面对越来越多的时间序列数据的需求，InfluxDB 遇

到了一些无法克服的挑战。诸如 LevelDB 不支持热备份，对数据库进行安全备份需要关闭后才能复制；删除过期数据代价过高，等等。

后来 InfluxDB 采用自研的 TSM(Time-Structured Merge Tree) 作为存储引擎，其设计的核心思想就是牺牲部分功能来达到极致的性能。得益于自研的 TSM 存储引擎，InfluxDB 有着极强的写入能力，它允许高吞吐量，压缩和实时查询同一数据。

1.5.5 开源

数据库产品属于基础软件，与操作系统的发展有同样的路径，属于头部垄断，就像奥运会的获奖模式，只有金银铜牌才会被客户认同。

实力弱小、市场份额占有率有限的企业，以开源切入是一种主流的商业模式，在时序数据库排行前十名中只有 Kdb+ 和 Fauna 闭源。以非关系型数据库 MongoDB 为例，它于 2007 年成立，以开源、深耕社区、产品易用性极致体验为切入点，2012 年左右，就成为非关系型数据库的标杆产品。这些年的开源使其积累了大量的用户，于 2013 年推出自己的商业化版本，用更好的工具、服务进行商业变现，例如推出数据托管服务、付费技术支持。以开源为切入点的关键在于，客户相信原厂是最了解开源数据库的服务企业，虽然其他服务器也可以为开源软件服务，但是比起开源软件的维护者来说，还是落了下风。

InfluxDB 开源也是其受众广泛的原因之一，截至目前，InfluxDB 在 GitHub 上已经有了 2 万多的 star，足以见得其拥有良好的生态。不过目前只有单机版的 InfluxDB 开源，InfluxDB 开发商为了维持后续稳定的开发费用，对集群版做了闭源，实行收费使用，走商业路线。

1.6 时序数据库的应用场景

时序数据库经常应用于工业环境监控、物联网 IoT 设备采集存储、互联网业务、性能监控服务、自动驾驶等基于时间线且多源数据连续涌入数据平台的应用场景，InfluxDB 专为时序数据存储而生，尤其是在工业领域的智能制造方面应用潜力巨大。

1.6.1 在工业环境监控中的应用

在工业领域，通常需要对实时产生的大量时序数据进行采集、处理、分析。处理这些数据，最重要的一点就是要做到实时，实时写入、实时查询，InfluxDB 在这方面做得很好。

假若某公司有如下需求：由于工业要求，需要将生产过程中产生的工况数据采集起

来做分析，公司一共有 20 个厂区，每个厂区有 20000 个采集点，要求 500 毫秒为一个采集周期，通过计算，每天的数据量高达 700 多万点，一年的数据量更是达到了 26 万亿点，假如每个数据点大小为 50Byte，那么总的数据大小将达到 1PB，如果每台服务器硬盘大小为 10TB，那么总共需要 100 多台服务器。且这些数据不只要快速生成写入存储，还要支持快速查询，做可视化展示。

如此一来，传统关系型数据库便不能达到要求，而 InfluxDB 因特殊设计的存储结构，能够做到每秒钟支持上千万甚至上亿的数据写入和秒级的对上亿数据的分组聚合计算，所以在面对海量数据涌入的时候能够实时处理，并且快速查询。

1.6.2　在物联网 IoT 设备采集存储中的应用

物联网是指通过各种传感器、红外感应等装置与技术，实时采集任何需要监控、连接的物体，采集内容包括声、光、热、力、电等各种信息，并对这些信息进行集中分析处理，实现对这些事物的智能感知和分析。

例如智能家居系统中，需要对各种电器进行实时的数据采集并进行快速分析，从而达到监控报警、大数据计算、业务报表等功能。InfluxDB 等时序数据库就起到了重要作用，物联网平台和时序数据库进行数据打通，实现了物联网设备的开发管理、数据分析等一体化方案，构建智慧互联网平台。其具体架构如图 1-5 所示。

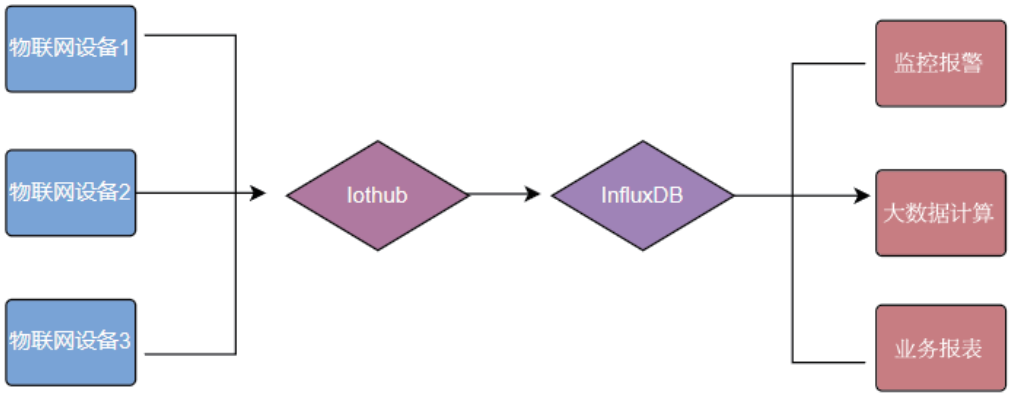

图 1-5　智慧物联架构图

物联网设备通过 IoT 套件设备连接管理，再将数据发送到 InfluxDB 数据库，然后基于 InfluxDB 对数据进行分析、监控等。

1.6.3　互联网业务性能监控服务

在互联网服务中，需要采集用户的访问延迟大小、业务服务指标监控数据以及查询返回效率等，时序数据库 InfluxDB 就可以对这些数据进行多维度分组聚合分析和监控项展示。

举一个例子，某网站需要实时统计每个网页的流畅度、清晰度、单击量、访问量等信息，于是就可以将这些监控项以某一频率写入时序数据库中，通过一定的聚合计算，可以得到某一时间段内哪一个运营商的网络更流畅；查看任意一段时间内某个站点的流畅度曲线；单击率随着流畅度的变化规律。

时序数据库对这些写入的信息进行分析计算，再通过 API 获取信息，实现高效的实时分析以及建模等需求。

1.6.4 在智能汽车中的应用

作为一个一百年来未变的产业，汽车产业正在以前所未有的速度发生变革，我们将迎来一个脱胎换骨的产业重构。

智能汽车是时代的必然产物，正在一步步地改变我们的生活。从技术层面讲，智能汽车在运行时需要监控各种状态，包括坐标、速度、方向、温度、湿度等，并且需要将数据及时收集起来做大数据分析。这样一来，每年累积的数据量是十分惊人的。如果只是存储下来不做查询也还好，但如果需要进行例如“今天下午六点在滨湖路速度超过 70% 的汽车有哪些”这种多维度分组聚合查询的操作时，时序数据库会是一个很好的选择。

1.7 小结

通过本章的学习，我们了解了什么是时序数据和时序数据库，包括时序数据库数据的写入、读取以及存储的特点。并且对 InfluxDB 有了基本的掌握，知道 InfluxDB 的基本概念，了解其独特的设计理念和核心特性。作为开发人员，在未来的工作中应该学会善用 InfluxDB，良好地运用其强大的功能来满足日益增长的需求。