爽！可以自己亲手搭建并有效管理大数据平台了！

大数据

从基础理论到最佳实践

主　编　祁　伟
副主编　刘　冰　常志军　赵廷涛　高俊秀

清華大學出版社

大数据

从基础理论到最佳实践

主　编　祁　伟
副主编　刘　冰　常志军　赵廷涛　高俊秀

清華大學出版社
北京

内 容 简 介

本书侧重于大数据的实践性技术，系统地介绍了主流大数据平台及工具的安装部署、管理维护和应用开发。平台和工具的选择均为当前业界主流的开源产品，因此，对于读者来说，有很强的可操作性。

本书涉及的开源技术包括：HDFS、MapReduce、YARN、Zookeeper、HBase、Hive、Sqoop、Storm、Kafka、Flume 等。除介绍一般性的背景知识、安装部署、管理维护和应用开发技术外，还特别注重案例实践，重要的技术点以实际工作场景或案例为依托，使读者能快速入门，参考案例动手实践，通过具体深入的实践，体会大数据的技术本质特征，领略大数据技术带来的创新理念，更好地理解和把握信息技术的发展趋势。

本书主要内容包括以下几大部分。

大数据存储篇：以 HDFS 为基础，介绍分布式文件系统的原理、安装、fs 命令的使用、编程，介绍如何用 HDFS 实现，并通过 HTTP 调用。

大数据计算篇：以 MapReduce、YARN 为基础，介绍分布式计算的原理、部署，以及编程案例。

非关系型数据库篇：以 HBase 为基础，重点介绍非关系型数据库的优势、原理、部署，以及命令行使用，编程案例，与 Sqoop 配合使用等。

大数据仓库篇：以 Hive、数据仓库等为基础，重点介绍数据的抽取、原理、部署、分析与编程。

大数据实时计算篇：以 Storm、Kafka 为基础，介绍实时计算的架构、组成、使用与开发。

本书非常适合从事大数据技术开发与使用的初学者，以及从事大数据技术研发的企事业单位工程师学习和参考，也适合高校计算机相关专业的专科生、本科生和研究生学习使用。

图书在版编目(CIP)数据

大数据：从基础理论到最佳实践/祁伟主编. —北京：清华大学出版社，2017
ISBN 978-7-302-45743-5

Ⅰ. ①大…　Ⅱ. ①祁…　Ⅲ. ①数据处理　Ⅳ. ①TP274

中国版本图书馆 CIP 数据核字(2016)第 290300 号

责任编辑：杨作梅　宋延清
装帧设计：杨玉兰
责任校对：张　瑜
责任印制：沈　露
出版发行：清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 装 者：北京嘉实印刷有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张**：21.5　　**字　数**：499 千字
版　　次：2017 年 1 月第 1 版　　**印　次**：2017 年 1 月第 1 次印刷
印　　数：1～3000
定　　价：59.80 元

产品编号：069305-01

前　言

技术革命的浪潮推动着人类文明的发展。

第一次浪潮造就了农业革命，它在数千年前出现并持续了数千年；第二次浪潮造就了工业革命，它在数百年前出现并持续了数百年；我们今天正在经历着信息技术第三次浪潮，发端于数十年前，目前也只是处在初级阶段。

农业技术革命释放了“物之力”；工业技术革命释放了“能之力”，而今天的信息技术革命释放的是“智之力”。

距今 400 年前，培根在《伟大的复兴》中预言：知识就是力量。今天，人类终于迎来“知识经济时代”，它是人类社会经济增长方式与经济发展的全新模式。

人类认识物质世界、人类社会和精神世界的最高境界是智慧，而要达智慧的境界，必然要跨越数据、信息、知识三个层级。

数据作为基础，是信息之母、知识之初、智慧之源。正是今天的大数据技术，引燃了人们实现智慧城市、智慧医疗、智慧教育等有关人工智慧的激情。人们真切地认识到，对于人工智能，只要让数据发生质变，即使是简单的数据，也比复杂的算法更有效。

今天，移动互联网的发展，使我们在获取数据上有了质的飞跃，人类的各种社会活动都与互联网这个虚拟世界相联系，使全样本、全过程地有效测量和记录成为可能，构建了生成大数据生态的土壤，同时，人们还在期待和憧憬物联网带来更大的冲击。

另一方面，云计算发展到今天，不论从技术到产业都开始进入成熟期，这也是大数据发展的基石和推进器。

在今天这个时代中，运用大数据洞见事物蕴藏的“智慧”成为人们的渴望。大数据更新了人们对数据的认识。在技术层面，小数据时代的很多数据处理方法和工具已不再有效，需要一系列新的方法和工具。所幸，有大量平民化的开源软件可用，它们不需要特殊的硬件系统，也更适用于云计算环境。

本书正是一本介绍主流的大数据开源软件平台和工具的技术专著，侧重于大数据的实践性技术，帮助读者快速入门，通过具体深入的实践，体会大数据的技术本质特征，领略大数据技术带来的创新理念，更好地理解和把握信息技术的发展趋势。

本书定位

(1) 信息发展已步入大数据时代，当前对于大数据还缺乏面向公众的技术实践手册。

(2) 本书的创作团队有丰富的大数据规划、开发、运营等经验，多位作者成功地架构了教育部、科技部、互联网等大数据架构与分析项目。

(3) 本书的参与者均是部委信息一线工程师、著名外企架构师、国内企业资深高级工程师，所做的理论分析易于学习，实践具有可操作性。

(4) 本书重点介绍大数据的基础理论、关键技术，以及编程实践。利用本书，就可以完全搭建并能有效地管理好大数据平台。

本书特色

(1) 理念先进：均是国内外最新的大数据理念；方便读者全面了解国内外大数据研究与发展的情况。

(2) 技术领先：参与者均是国内 IT 人士；采用的平台均是业界主流开源平台，涉及大数据常用的 HDFS、MapReduce、YARN、Zookeeper、HBase、Hive、Sqoop、Storm、Kafka 等技术的介绍与编程使用。

(3) 案例丰富：提供翔实的实例与解决方法，供项目中参考。

(4) 资源齐备：本书涉及的配套下载资源可以从清华大学出版社的网站中下载。

全书关键字

大数据、分布式计算、数据仓库、数据分析、HDFS、MapReduce、YARN、Zookeeper、HBase、Hive、Sqoop、Storm、Kafka。

由于编者的水平有限，书中难免有疏漏和错误，希望业内专家和广大读者指正。

编　者

目　录

大数据存储篇

大数据计算篇

非关系型数据库篇

大数据仓库篇

大数据实时计算篇

大数据存储篇

第1章

概述

学习目标：

本章通过对大数据技术的简要概述，使读者了解大数据的基本概念，以及在分布式环境下，大数据存储和处理的基本原则，掌握大数据主流技术分布，了解大数据可能的职业发展方向，为后续的学习提供指引。

1.1 什么是大数据

有观点认为，人类过去经历了三次工业化技术革命，从蒸汽机时代，到电力时代，再到早期的计算机时代，每一次革命都释放了巨大的生产力，开创了工业的转型和经济的增长时期。人们都说，现在人类正在经历第四次技术革命，数据就是新的源动力。

的确，我们已经看到了海量数据的爆炸式增长景观，特别是来自云端的数据。云端提供了前所未有的计算能力和数据存储能力。这表明，我们已身处“大数据”时代。

但是，关于大数据的确切定义，目前尚未获得统一、公认的说法。

IBM 用 3V(Volume、Variety、Velocity)来描述大数据所拥有的特点。

大容量(Volume)，是指数据体量巨大。

多形式(Variety)，是从数据的类型角度来看的，数据的存在形式从过去的以结构化数据为主转换为形式多种多样，既包含传统的结构化数据，也包含可便于搜索的半结构化数据，如文本数据，还包含更多的非结构化数据，如图片、音频和视频数据。

高速率(Velocity)则是从数据产生效率的实时性角度来衡量的，数据以非常高的速率产生，比如大量传感器生成的实时数据。

之后，IBM 又在 3V 的基础上，增加了 Value 这个维度，即价值密度低的数据称为大数据，意指大数据伴随着从低价值的原始数据中进行深度挖掘和计算，从海量且形式各异的数据源中抽取出富含价值的信息。

由此可以看出，从具备 4V 特性的大量数据中挖掘高价值知识，是各界对于大数据的一个共识。

由于数据量的爆炸式增长，传统的数据管理模式及工具已不能高效地存储和处理如此规模的数据。新时代呼唤新思维、新技术。从维克多·迈尔·舍恩伯格所著的《大数据时代》中，可以看到大数据时代的思维变革。

(1) 不是随机样本，而是全体数据。

统计学家们证明：采样分析的精确性随着采样随机性的增加而大幅提高，但与样本数量的增加关系不大。随机采样取得了巨大的成功，成为现代社会、现代测量领域的主心骨。但这只是一条捷径，是在不可收集和分析全部数据的情况下的选择，它本身存在许多固有的缺陷。大数据是指不用随机分析法这样的捷径，而采用所有数据的方法。

(2) 不是精确性，而是混杂性。

数据多比少好，更多数据比算法系统更智能还要重要。社会从“大数据”中所能得到的益处，并非来自运行更快的芯片或更好的算法，而是来自更多的数据。大数据的简单算法比小数据的复杂算法更有效。大数据不仅让我们不再期待精确性，也让我们无法实现精确性。那些精确的系统试图让我们接受一个贫乏而规整的惨象——假装世间万物都是整齐地排列的。而事实上，现实是纷繁复杂的，天地间存在的事物也远远多于系统所设想的。要想获得大规模数据带来的好处，混乱应该是一种标准途径，而不应该是竭力避免的。

(3) 不是因果关系，而是相关关系。

在大数据时代，我们不必非得知道现象背后的原因，而是要让数据自己“发声”。通过给我们找到一个现象的良好关联物，相关关系可以帮助我们捕捉现在和预测未来。

在小数据世界中，相关关系也是有用的，但在大数据的背景下，相关关系大放异彩。通过应用相关关系，我们可以比以前更容易、更快捷、更清楚地分析事物。

大数据的相关关系分析法更准确、更快，而且不易受偏见的影响。建立在相关关系分析法基础上的预测是大数据的核心。

1.2　大数据的技术转型

直至今天，我们无论是使用 ATM 机取款、预订航班机票，还是查询交通违章信息，都离不开关系数据库，都依托于背后的关系数据库的数据事务处理。在事务处理上，关系模型无处不在。关系数据库模型成功的关键之处，在于该模型的标准化。每个数据单元在一个表中只会出现一次，这不但减少了冗余的存储成本，而且还使得数据修改只发生在一个位置，数据的一致性得以保持。表中的某一列可被指定为主键，主键是一个保证无二义性地检索单条记录的属性值，还可以在查询语法中使用主键来连接这些关系。关系数据库还创造出了结构化查询语言(Structured Query Language，SQL)来表达关系查询。关系型数据库为存储和查询数据提供了非常灵活的模型。

关系数据库模型的另一个重要概念是事务。事务管理器(或者一个事务处理服务)在一个工作单元中维护数据的完整性。一组操作被当作一个工作单元(unit)，在一个工作单元中，操作的所有部分一起成功，或失败并恢复。

凡符合关系模型的数据库，在事务处理上需要满足被称为 ACID 的特性。

- 原子性(Atomicity)：一个事务要被完全地、无二义性地做完或撤销。在任何操作出现一个错误的情况下，构成事务的所有操作的效果必须被撤销，数据应被回滚到事务开始前的状态。
- 一致性(Consistency)：一个事务应该保护所有定义在数据上的不变的属性(例如完整性约束)。在完成了一个成功的事务时，数据应处于一致的状态。一个一致的事务将保护定义在数据上的所有完整性约束。
- 隔离性(Isolation)：在同一个环境中，可能有多个事务并发执行，而每个事务都应表现为独立执行。串行地执行一系列事务的效果应该等同于并发地执行它们。在一个事务执行的过程中，数据的中间状态不应该被暴露给所有的其他事务；两个并发的事务应该不能操作同一项数据。
- 持久性(Durability)：一个被完成的事务的效果应该是持久的。只要事务成功结束，它对数据库所做的更新就必须永久保存下来。即使发生系统崩溃，重新启动数据库系统后，数据库还能恢复到事务成功结束时的状态。

关系数据库有两个强制性要求：对数据要预先理解，在进行插入操作之前，必须先定义好模式和关系；在写入操作发生后，保持数据的一致性。

可扩展性是这种计算模式的一大缺陷。当数据容量更大、并发处理性能需求更高时，唯有提高服务器性能指标和可靠性，这是典型的向上扩展模式(Scale Up)。即使可采用并行数据库集群，最多也只能管理有限数量的服务器，而且这种并行数据库也同样要求高配置的服务器才可以运转，其成本之高可以想象。

随着信息技术的进步，相比较而言，软件的重要性将下降，数据的重要性将上升。人类处理、传输和保存数据的能力不断增强。

20 多年来，CPU 的性能提高了 3500 倍；内存和磁盘的价格下降到了原来的 450 万分之一和 360 万分之一；主干网络带宽每 6 个月增加 1 倍，而每比特费用将趋于零。

另一方面，人类生产数据的能力也在增强，数据正在发生井喷式的增长。这与互联网、移动互联网、社交网络以及未来物联网大潮的高速发展直接相关。特别是智能手机等手持设备和社交网络的广泛使用，使得越来越多的人能够将可支配时间投入到各种应用中，而未来物联网的发展，任意物品和设施都有可能 24 小时不间断地产生状态数据。

让我们看一看当今互联网应用场景：Facebook 管理了超过 400 亿张图片，所需存储空间超过 100PB，每天发布的新消息超过 60 亿条，所需的存储空间超过 10TB；Twitter 一天产生 1.9 亿条微博；搜索引擎一天产生的日志高达 35TB，Google 一天处理的数据量超过 25PB；YouTube 一天上传的视频总时长为 5 万小时……。

数据量的衡量单位，从小到大依次为 KB、MB、GB、TB、PB、EB 和 ZB，相互之间的转换公式为 1024KB=1MB；1024MB=1GB；1024GB=1TB；1024TB=1PB；1024PB=1EB；1024EB=1ZB。我们曾经经历过 MB 存储时代、GB 存储时代，随着 IT 技术的快速发展，我们迈入了 TB 时代，而现在正向 PB、EB 时代迁移。

尽管随着时间的推移，商用计算机硬件变得越来越便宜，但是，从历史和经济的角度来看，持续不断地升级到更高配置的服务器硬件是不可行的。花费数倍的价钱升级一个大型机器，可能无法提供同样倍数的性能。相比之下，性能一般的小型服务器仍然很便宜。一般情况下，从经济的角度来看，水平扩展更有意义。换句话说，应该简单地为系统增加更多便宜的机器，而不是试图将一个关系型数据库放到一台昂贵的大型服务器上。

以关系数据库技术，不可能支撑今天大数据的应用场景。

对于很多应用场景，尤其是互联网相关应用来说，并不像银行业务等对数据的一致性有很高的要求，而更看重数据的高可用性以及架构的可扩展性等技术因素。因此，NoSQL 数据库应运而生，作为适应不同应用场景要求的新型数据存储与处理架构，它与传统数据库有很强的互补作用，而且应用场景更加广泛。例如，Yahoo 公司通过部署包含 4000 台普通服务器的 Hadoop 集群，可以存储和处理高达 4PB 的数据，整个分布式架构具有非常强的可扩展性。NoSQL 数据库的广泛使用，代表了一种技术范型的转换。

1.3 数 据 分 片

在大数据环境下，数据量已经由 GB 级别跨越到 PB 级别，依靠单台计算机已经无法存储与处理如此规模的数据，唯一的出路，是采用大规模集群来对这些数据进行存储和处理，所以，系统的可扩展性成为衡量系统优劣的关键因素。

传统关系数据库系统为了支持更多的数据，采用纵向扩展(Scale Up)的方式，即不增加机器数量，而是通过改善单机硬件资源配置，来解决问题。如今这种方式已经行不通了。

目前主流的大数据存储与计算系统通常采用横向扩展(Scale Out)的方式支持系统可扩展性，即通过增加机器数目来获得水平扩展能力。与此对应，对于待存储处理的海量数据，需要通过数据分片(Shard/Partition)来对数据进行切分并分配到各个机器中去，通过数据分片

实现系统的水平扩展。

目前，大规模存储与计算系统都是采用普通商用服务器来作为硬件资源池的，系统故障被认为是常态。因此，与数据分片密切相关的是数据复制，通过数据复制来保证数据的高可用性。数据复制是将同一份数据复制存储在多台计算机中，以保证数据在故障常发环境下仍然可用。从数据复制还可以获得另一个好处，即可以增加读操作的效率，客户端可以从多个备份数据中选择物理距离较近的进行读取，既增加了读操作的并发性，又可以提高单次的读取效率。

数据复制带来的难题是如何保证数据的一致性。由于每份数据存在多个副本，在并发地对数据进行更新时，如何保证数据的一致性就成为关键问题。

可以将数据分片的通用模型看作是一个二级映射关系。第一级映射是key-partition映射，即把数据记录映射到数据分片空间，通常，一个数据分片包含多条记录数据；第二级映射是 partition-machine 映射，把数据分片映射到物理机器中，即一台物理机器通常可以容纳多个数据分片。

在做数据分片时，根据 key-partition 关系，将数据水平切割成众多数据分片，然后再按照 partition-machine 映射关系，将数据分片存储到对应的物理机器上。而在做数据访问时，比如要查找某条记录的值 Get(key)，首先根据 key-partition 映射找到对应的数据分片，然后再查找 partition-machine 关系表，就可以找到哪台物理机器存储该条数据，之后，即可从相应的物理机器读取 key 对应的 value 内容了。

数据分片有两种常用策略：哈希分片与范围分片。对于哈希分片来说，因为其主要通过哈希函数来建立 key-partition 映射关系，因此，哈希分片只支持“单点查询”(Point Query)，即根据某个记录的主键(key)获得记录内容，而无法支持“范围查询”(Range Query)，即指定记录的主键范围，一次读取多条满足条件的记录。采取哈希分片的实际系统众多，大多数 KV(key-value，键值)存储系统都支持这种方式。

与此相对应地，范围分片的系统则既可以支持单点查询，也可以支持范围查询。范围分片首先对所有记录的主键进行排序，然后在排好序的主键空间里将记录划分成数据分片，每个数据分片存储有序的主键空间片段内的所有记录。

1.4　数据一致性

在大数据系统中，为了获得系统可用性，需要为同一数据分片存储多份副本，业界的常规做法是一个数据分片同时保存三个副本。将数据复制成多份除了能增加存储系统的可用性，同时还能增加读操作的并发性，但引发了数据一致性问题，即同一数据分片存在多个副本。在并发的写请求下，如何保持数据一致性尤为重要，即在存储系统外部的使用者看来，即使存在多个副本数据，它与单份数据也应该是一样的。

CAP、BASE、ACID 等基本原则是分布式环境下数据一致性方案设计重要的指导原则。

1.4.1　CAP 原则

CAP 是对 Consistency/Availability/Partition Tolerance 的一种简称，CAP 原则对分布式系

统中的三个特性进行了如下归纳。

- 一致性(Consistency)：在分布式系统中的所有数据备份，在同一时刻是否具有同样的值(等同于所有节点访问同一份最新的数据副本)。
- 可用性(Availability)：在集群中，一部分节点出现故障后，集群整体是否还能响应客户端的读写请求(对数据更新具备高可用性)。
- 分区容错性(Partition Tolerance)：从实际效果而言，分区相当于对通信的时限要求。系统如果不能在时限内达成数据一致性，就意味着发生了分区的情况，必须就当前操作在 C 和 A 之间做出选择。

CAP 原则是指对于一个大规模分布式数据系统来说，CAP 三个特性不可兼得，同一个系统至多只能实现其中的两个，而必须放宽第三个要素来保证其他两个要素被满足。即要么 AP，要么 CP，抑或 AC，但是不存在 CAP，如图 1-1 所示。

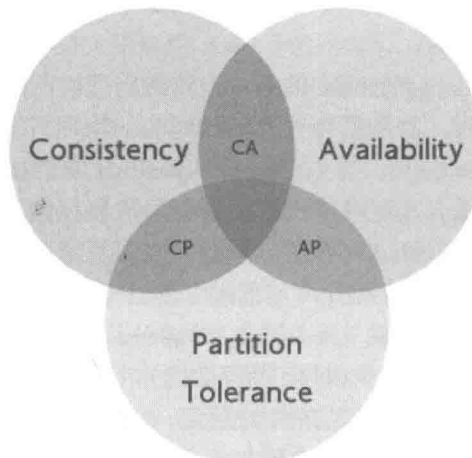

图 1-1　CAP 原则示意

我们可以认为，在网络环境下，运行环境出现网络分区是不可避免的，所以系统必须具备分区容忍性特性，于是，一般在此种场景下，设计大规模分布式系统时，架构师往往在 AP 和 CP 中进行权衡和选择，有所强调、有所放弃。

在一个分布式系统中，如果数据无副本，那么系统首先就满足强一致性条件，单一副本数据，不可能发生数据不一致的情况。此时，系统具备 C，如果我们容忍分区容错性 P，意味着系统可能发生网络分区状况，如有宕机现象出现时，就必然导致某些数据不可访问，此时，可用性 A 是不能被满足的，即在此情形下获得了 CP 特性，但 CAP 不可同时满足。

如果系统中数据有副本，假设一份数据有分别存储在不同机器上的两个副本，最初数据是保持一致的，某一时刻，机器 1 上对这个数据进行了更新操作，这个更新操作随后会同步到机器 2 上，使两个副本保持一致性。但网络分区是不可忽视的，可以设想，在数据复制同步未完成的情况下，发生了网络分区，导致两台机器无法通信，这时，我们就不得不在 C 或 A 之间做权衡和选择。如果希望系统可用性优先(选择 A)，那么对于读取机器 2 上的数据查询请求返回的并非是最新的数据，此种选择就放弃一致性(C)，产生数据不一致情况。如果选择一致性优先(选择 C)，那么，在两台机器恢复通信并将数据同步到一致状态

前，对于机器 2 就要拒绝对数据的读请求，此时，可用性无法保证(放弃 A)。所以不论选择哪一个，必然以牺牲另外一个因素作为代价，也就是说，要么 AP，要么 CP，但不会有 CAP。

对于分布式系统来说，分区容错性是天然具备的，所以在设计具体分布式架构技术方案时，只能对一致性和可用性两个特性做出取舍，要么选择强一致性，减弱服务可用性，要么选择高可用性而容忍弱一致性。

我们可以回顾一下传统关系数据库的 ACID 特性，在 CAP 三要素中，传统关系数据库选择 CA 两个特性，即强一致性、高可用性，与分区容错性这个天然要素不可调和，这是造成其在分布式环境下可扩展性差的根本原因。而 NoSQL 系统则更关注 AP 因素，即高可用性和分区容错性，也意味着兼顾高可用性和高可扩展性，而牺牲强一致性。对于绝大多数互联网应用来说，高可用性直接涉及用户体验，而对数据一致性要求并不高，NoSQL 系统这类以弱一致性作为代价的系统，正是适应了这一现实需求。

网络分区(P)虽然是个天然属性，但在现代的实际系统中，依然是个小概率事件。以此为出发点，我们并不应该为了容忍这种小概率事件而在设计之初就选择放弃 A 或者放弃 C，在大多数没有出现网络分区的状况下，还是可以尽可能兼顾 AC 两者，即有条件地兼顾 CAP 三要素。

另一个需要说明的是，即使必须在 AC 之间做出取舍，也不应该是粗粒度地在整个系统级别进行取舍，即整个系统要么取 A 舍 C，要么取 C 舍 A，而是应该考虑系统中存在不同的子系统，甚至应该在不同的系统运行时或者在不同的数据间进行灵活的差异化的细粒度取舍，即可能对不同子系统采取不同的取舍策略，在不同的系统运行时，对不同数据采取不同的取舍策略。因此，CAP 三者并非是绝对的两要素有或没有，可以看作是在一定程度上的有或没有，考虑的是在多大程度上进行取舍。

由此可以看出，CAP 的修改策略是可取的。在绝大多数系统未产生网络分区的情形下，应该尽可能保证 AC 两者兼得，也即大多数情况下，考虑 CAP 三者兼得，当发生网络分区时，系统应该能够识别这种状况并对其进行正确处理，在网络分区场景下进入明确的分区模式，此时，可能会限制某些系统操作，最后，在网络分区解决后，能够进行善后处理，即恢复数据的一致性，或者弥补分区模式中产生的错误。

1.4.2　CAP 与 ACID

谈到 CAP 与 ACID 之间存在的关系，首先因为 CAP 和 ACID 两者都包含了 A 和 C，但是其具体含义有所不同；其次，如果 CAP 中选择 A 的话，在一定程度上，会影响 ACID 中的部分要求。

CAP 与 ACID 两者中尽管都包含一致性，但是，两者的含义不同，ACID 中的 C 指的是对操作的一致性约束，而 CAP 中的 C 指的是数据的强一致性(多副本对外表现类似于单副本)，所以，可以将 CAP 中的 C 看作一致性约束的一种，即 CAP 中的 C 是 ACID 中的 C 所涵盖语义的子集。在出现网络分区的情形下，ACID 中的 C 所要求的一致性约束是无法保证的，所以，在网络分区解决后，需要通过一定手段来恢复 ACID 中要求的一致性。

当出现网络分区时，ACID 中的事务独立只能在多个分区中的某个分区执行，因为事务的序列化要求通信，而当网络分区时，明显无法做到这点，所以只能在某个分区执行。如果多个分区都可以各自进行 ACID 中的数据持久化(D)操作，当网络分区解决后，如果每个