第1章 数据存储与管理

（PPT版本号：2022--V1）

CONTENTS

6.1 传统的数据存储与管理技术

6.1.1文件系统

6.1.2关系数据库

6.1.3数据仓库

6.1.4并行数据库

6.1.1文件系统

文件系统是操作系统用于明确存储设备（常见的是磁盘，也有基于NAND Flash的固态硬盘）或分区上的文件的方法和数据结构，即在存储设备上组织文件的方法。操作系统中负责管理和存储文件信息的软件机构称为文件管理系统，简称“文件系统”。

我们平时在计算机上使用的WORD文件、PPT文件、文本文件、音频文件、视频文件等，都是由操作系统中的文件系统进行统一管理的。

6.1.2关系数据库

一个关系数据库可以看成是许多关系表的集合，每个关系表可以看成一张二维表格

目前市场上常见的关系数据库产品包括Oracle、SQL Server、MySQL、DB2等

表 学生信息表

学号	姓名	性别	年龄	考试成绩
95001	张三	男	21	88
95002	李四	男	22	95
95003	王梅	女	22	73
95004	林莉	女	21	96

6.1.2关系数据库

总体而言，关系数据库具有如下特点：

（1）存储方式。关系数据库采用表格的储存方式，数据以行和列的方式进行存储，要读取和查询都十分方便。

（2）存储结构。关系数据库按照结构化的方法存储数据，每个数据表的结构都必须事先定义好（比如表的名称、字段名称、字段类型、约束等），然后再根据表的结构存入数据，这样做的好处就是，由于数据的形式和内容在存入数据之前就已经定义好了，所以，整个数据表的可靠性和稳定性都比较高，但是带来的问题就是，数据模型不够灵活，一旦存入数据后，如果需要修改数据表的结构就会十分困难。

（3）存储规范。关系数据库为了规范化数据、减少重复数据以及充分利用好存储空间，把数据按照最小关系表的形式进行存储，这样数据管理就可以变得很清晰、一目了然。当存在多个表时，表和表之间通过主外键关系发生关联，并通过连接查询获得相关结果。

（4）扩展方式。由于关系数据库将数据存储在数据表中，数据操作的瓶颈出现在多张数据表的操作中，而且数据表越多这个问题越严重。如果要缓解这个问题，只能提高处理能力，也就是选择速度更快、性能更高的计算机，这样的方法虽然具有一定的拓展空间，但是这样的拓展空间是非常有限的，也就是一般的关系型数据库只具备有限的纵向扩展能力。

6.1.2关系数据库

（5）查询方式。关系数据库采用结构化查询语言（即SQL：Structured Query Language）来对数据库进行查询。结构化查询语言是高级的非过程化编程语言，允许用户在高层数据结构上工作。它不要求用户指定对数据的存放方法，也不需要用户了解具体的数据存放方式，所以，各种具有完全不同底层结构的数据库系统，可以使用相同的结构化查询语言作为数据输入与管理的接口。结构化查询语言语句可以嵌套，这使它具有极大的灵活性和强大的功能。

（6）事务性。关系数据库可以支持事务的ACID特性（原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）、持久性（Durability））。当事务被提交给了DBMS（数据库管理系统），则DBMS需要确保该事务中的所有操作都成功完成且其结果被永久保存在数据库中，如果事务中有的操作没有成功完成，则事务中的所有操作都需要被回滚，回到事务执行前的状态，从而确保数据库状态的一致性。

（7）连接方式。不同的关系数据库产品都遵守一个统一的数据库连接接口标准，即ODBC（Open Database Connectivity）。ODBC的一个显著优点是，用它生成的程序是与具体的数据库产品无关的，这样可以为数据库用户和开发人员屏蔽不同数据库异构环境的复杂性。ODBC提供了数据库访问的统一接口，为应用程序实现与平台的无关性和可移植性提供了基础，因而获得了广泛的支持和应用。

6.1.3数据仓库

数据仓库（Data Warehouse）是一个面向主题的、集成的、相对稳定的、反映历史变化的数据集合，用于支持管理决策：

（1）面向主题。操作型数据库的数据组织面向事务处理任务，而数据仓库中的数据是按照一定的主题域进行组织。主题是指用户使用数据仓库进行决策时所关心的重点方面，一个主题通常与多个操作型信息系统相关。

（2）集成。数据仓库的数据来自于分散的操作型数据，将所需数据从原来的数据中抽取出来，进行加工与集成、统一与综合之后才能进入数据仓库。

（3）相对稳定。数据仓库是不可更新的，数据仓库主要是为决策分析提供数据，所涉及的操作主要是数据的查询。

（4）反映历史变化。在构建数据仓库时，会每隔一定的时间（比如每周、每天或每小时）从数据源抽取数据并加载到数据仓库。

6.1.3数据仓库

一个典型的数据仓库系统通常包含数据源、数据存储和管理、OLAP服务器、前端工具和应用等四个部分

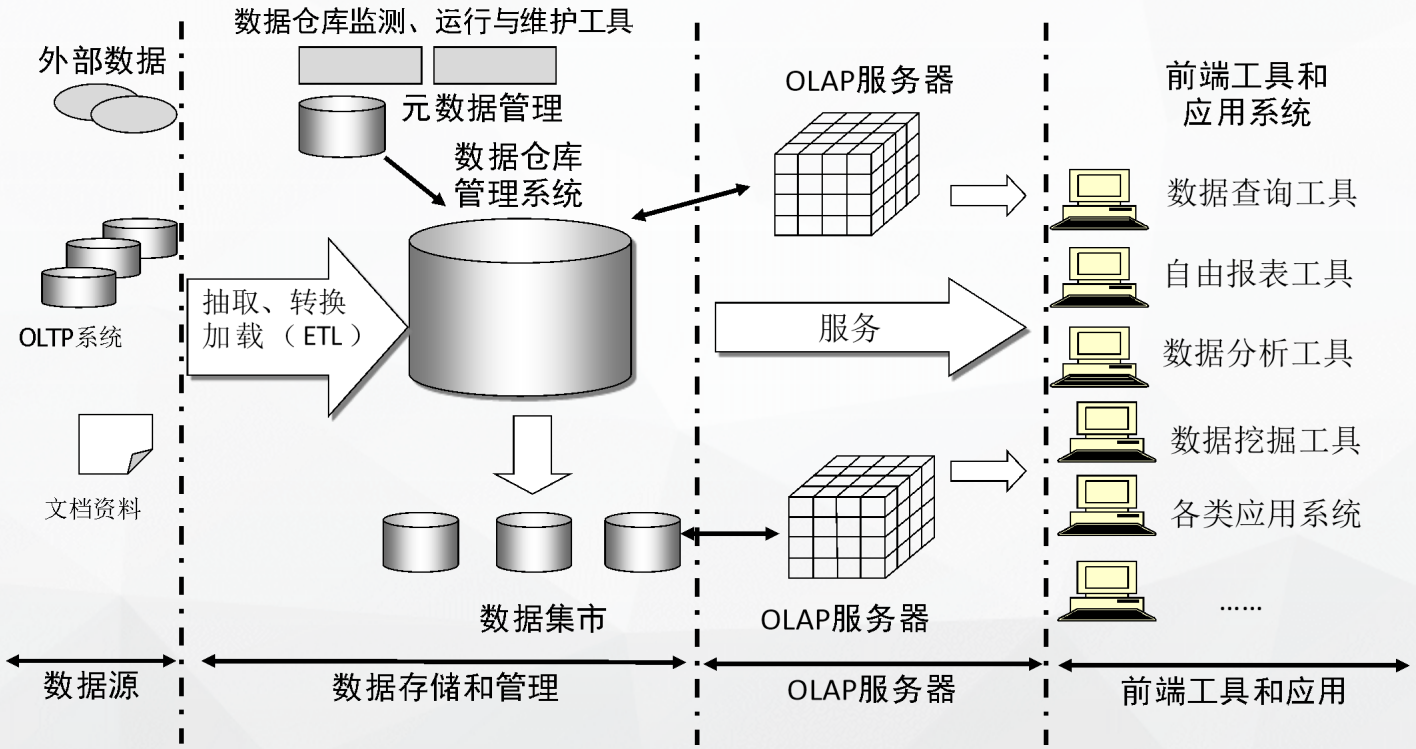


图 数据仓库体系架构

6.1.4并行数据库

并行数据库是指那些在无共享的体系结构中进行数据操作的数据库系统

这些系统大部分采用了关系数据模型并且支持SQL语句查询，但为了能够并行执行SQL的查询操作，系统中采用了两个关键技术：关系表的水平划分和SQL查询的分区执行

并行数据库系统的目标是高性能和高可用性,通过多个节点并行执行数据库任务，提高整个数据库系统的性能和可用性

6.1.4并行数据库

并行数据库系统的主要缺点就是没有较好的弹性，而这种特性对中小型企业和初创企业是有利的。人们在对并行数据库进行设计和优化的时候认为集群中节点的数量是固定的，若需要对集群进行扩展和收缩，则必须为数据转移过程制订周全的计划。这种数据转移的代价是昂贵的，并且会导致系统在某段时间内不可访问，而这种较差的灵活性直接影响到并行数据库的弹性以及现用现付商业模式的实用性。

并行数据库的另一个问题就是系统的容错性较差，过去人们认为节点故障是个特例，并不经常出现，因此系统只提供事务级别的容错功能，如果在查询过程中节点发生故障，那么整个查询都要从头开始重新执行。这种重启任务的策略使得并行数据库难以在拥有数以千个节点的集群上处理较长的查询，因为在这类集群中节点的故障经常发生。基于这种分析，并行数据库只适合于资源需求相对固定的应用程序。不管怎样，并行数据库的许多设计原则为其他海量数据系统的设计和优化提供了比较好的借鉴。

6.2

大数据时代的数据存储与管理技术

6.2.1分布式文件系统

6.2.2NewSQL和NoSQL数据库

6.2.3云数据库

6.2.1分布式文件系统

分布式文件系统（Distributed File System）是一种通过网络实现文件在多台主机上进行分布式存储的文件系统

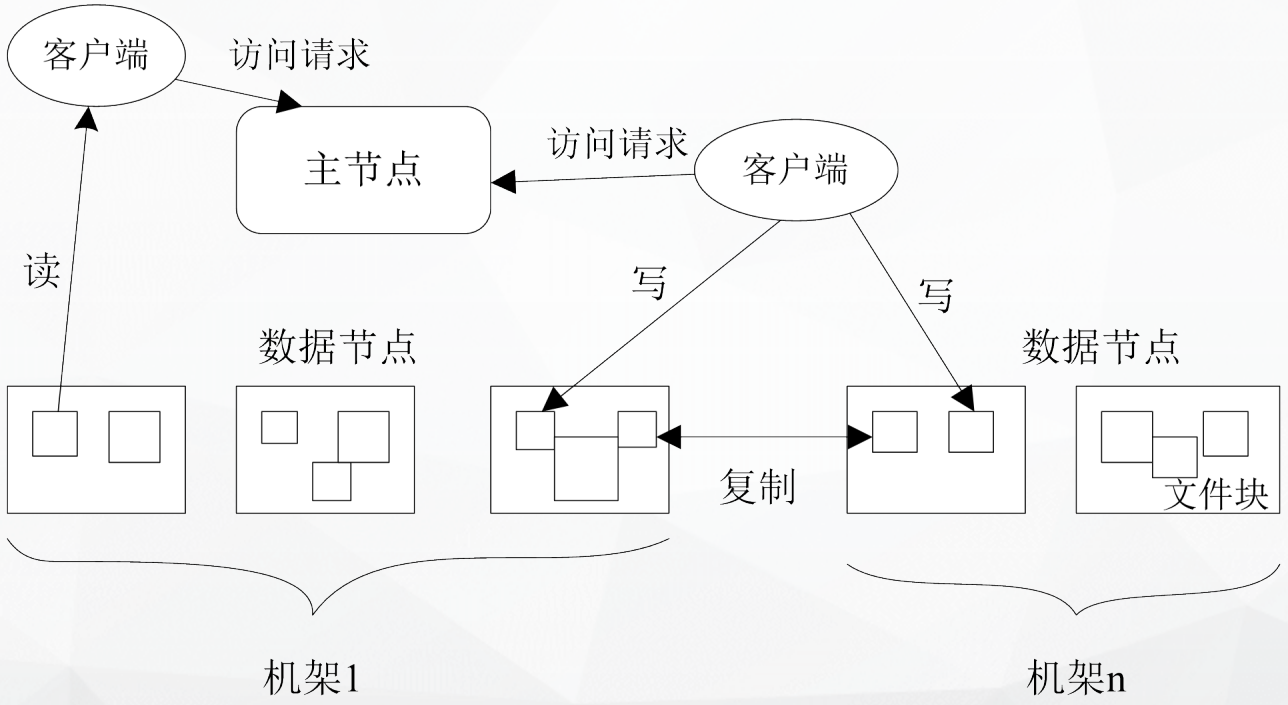


图 分布式文件系统的整体结构

6.2.2NewSQL和NoSQL数据库

1.NewSQL数据库

NewSQL是对各种新的可扩展、高性能数据库的简称，这类数据库不仅具有对海量数据的存储管理能力，还保持了传统数据库支持ACID和SQL等特性

目前具有代表性的NewSQL数据库主要包括Spanner、Clustrix、GenieDB、ScalArc、Schooner、VoltDB、RethinkDB、ScaleDB、Akiban、CodeFutures、ScaleBase、Translattice、NimbusDB、Drizzle、Tokutek、JustOne DB等

6.2.2NewSQL和NoSQL数据库

2.NoSQL数据库

NoSQL是一种不同于关系数据库的数据库管理系统设计方式，是对非关系型数据库的统称，它所采用的数据模型并非传统关系数据库的关系模型，而是类似键/值、列族、文档等非关系模型

NoSQL数据库没有固定的表结构，通常也不存在连接操作，也没有严格遵守ACID约束，因此，与关系数据库相比，NoSQL具有灵活的水平可扩展性，可以支持海量数据存储

6.2.2NewSQL和NoSQL数据库

3.大数据引发数据库架构变革

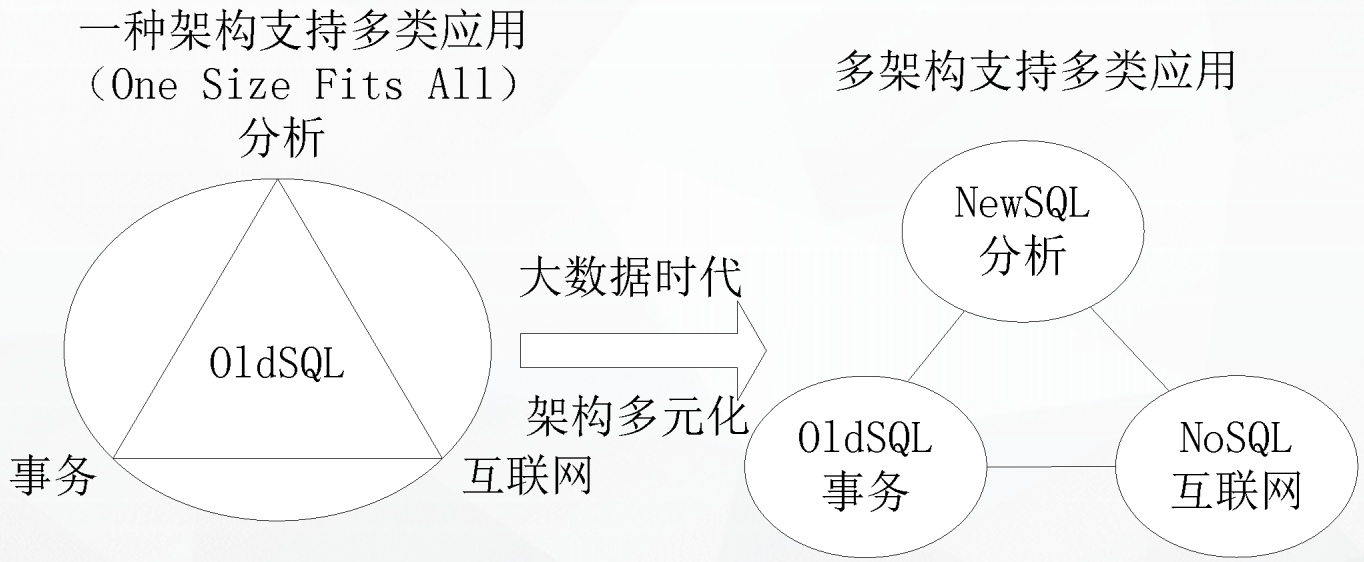


图 大数据引发数据库架构变革

6.2.3云数据库

研究机构IDC预言，大数据将按照每年60%的速度增加，其中包含结构化和非结构化数¬据。 如何方便、快捷、低成本地存储这些海量数据，是许多企业和机构面临的一个严峻挑战。云数据库就是一个非常好的解决方案，目前云服务提供商正通过云技术推出更多可在公有云中托管数据库的方法，将用户从繁琐的数据库硬件定制中解放出来，同时让用户拥有强大的数据库扩展能力，满足海量数据的存储需求。此外，云数据库还能够很好地满足企业动态变化的数据存储需求和中小企业的低成本数据存储需求。可以说，在大数据时代，云数据库将成为许多企业数据的目的地。

6.2.3云数据库

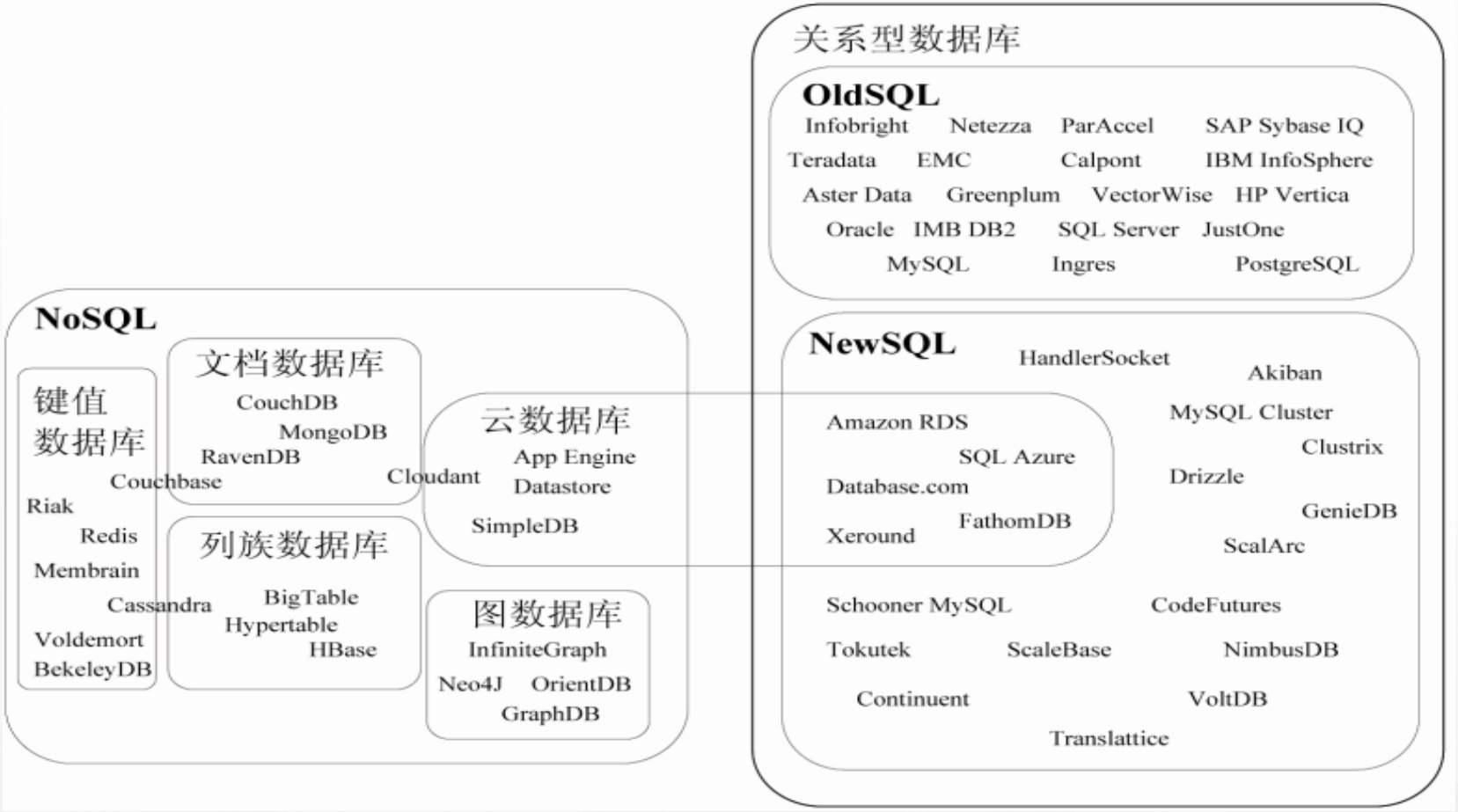


图 关系数据库、NoSQL、NewSQL和云数据库产品分类

6.3 大数据处理架构Hadoop

6.3.1 Hadoop特性

6.3.2 Hadoop生态系统

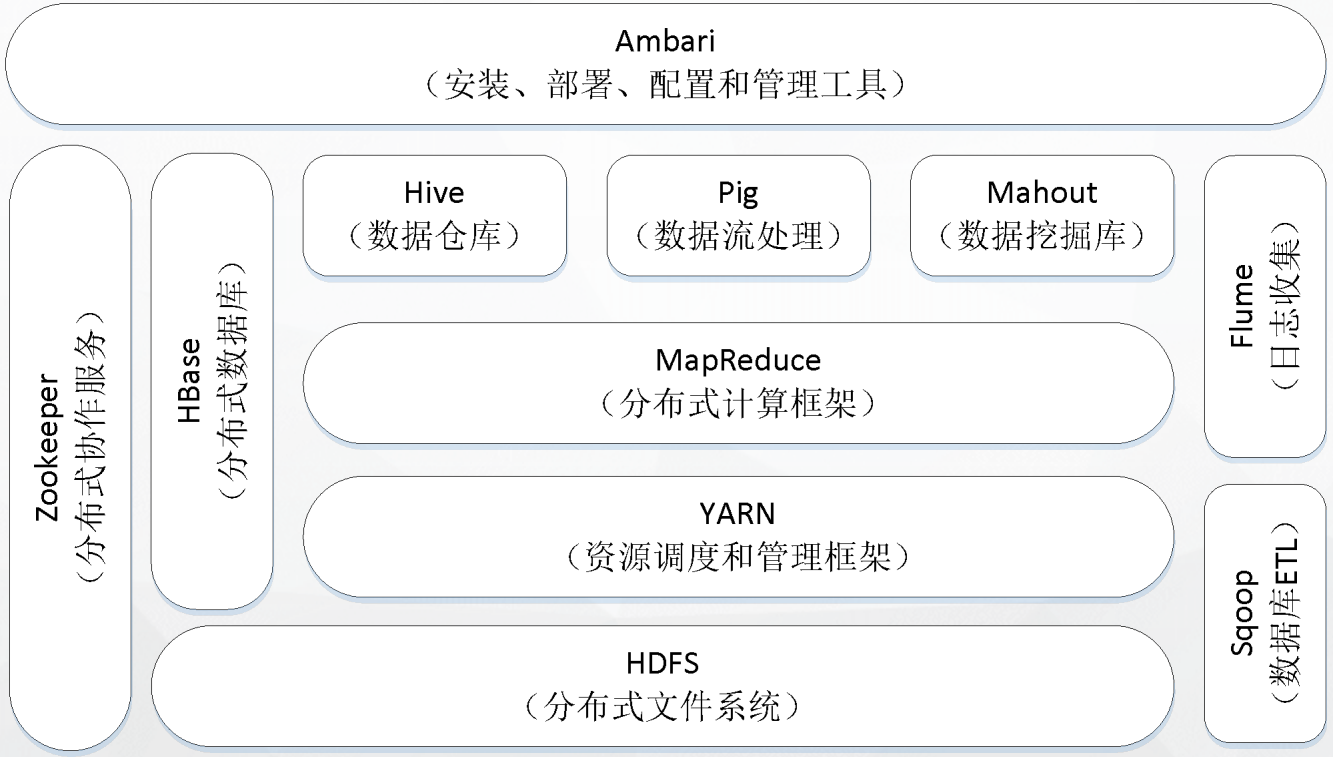
6.3.1 Hadoop特性

Hadoop是一个能够对大量数据进行分布式处理的软件框架，并且是以一种可靠、高效、可伸缩的方式进行处理的，它具有以下几个方面的特性：

高可靠性
高效性
高可扩展性
高容错性
成本低
运行在Linux平台上
支持多种编程语言

6.3.2 Hadoop生态系统

经过多年的发展，Hadoop生态系统不断完善和成熟，目前已经包含了多个子项目。除了核心的HDFS和MapReduce以外，Hadoop生态系统还包括Zookeeper、HBase、Hive、Pig、Mahout、Sqoop、Flume、Ambari等功能组件。



6.4 分布式文件系统HDFS

6.4.1 HDFS的设计目标

6.4.2 HDFS体系结构

6.4.1 HDFS的设计目标

总体而言，HDFS要实现以下目标：

- 兼容廉价的硬件设备
- 流数据读写
- 大数据集
- 简单的文件模型
- 强大的跨平台兼容性

HDFS特殊的设计，在实现上述优良特性的同时，也使得自身具有一些应用局限性，主要包括以下几个方面：

- 不适合低延迟数据访问
- 无法高效存储大量小文件
- 不支持多用户写入及任意修改文件

6.1.2关系数据库

HDFS采用了主从（Master/Slave）结构模型，一个HDFS集群包括一个名称节点（NameNode）和若干个数据节点（DataNode）（如图3-4所示）。名称节点作为中心服务器，负责管理文件系统的命名空间及客户端对文件的访问。集群中的数据节点一般是一个节点运行一个数据节点进程，负责处理文件系统客户端的读/写请求，在名称节点的统一调度下进行数据块的创建、删除和复制等操作。每个数据节点的数据实际上是保存在本地Linux文件系统中的

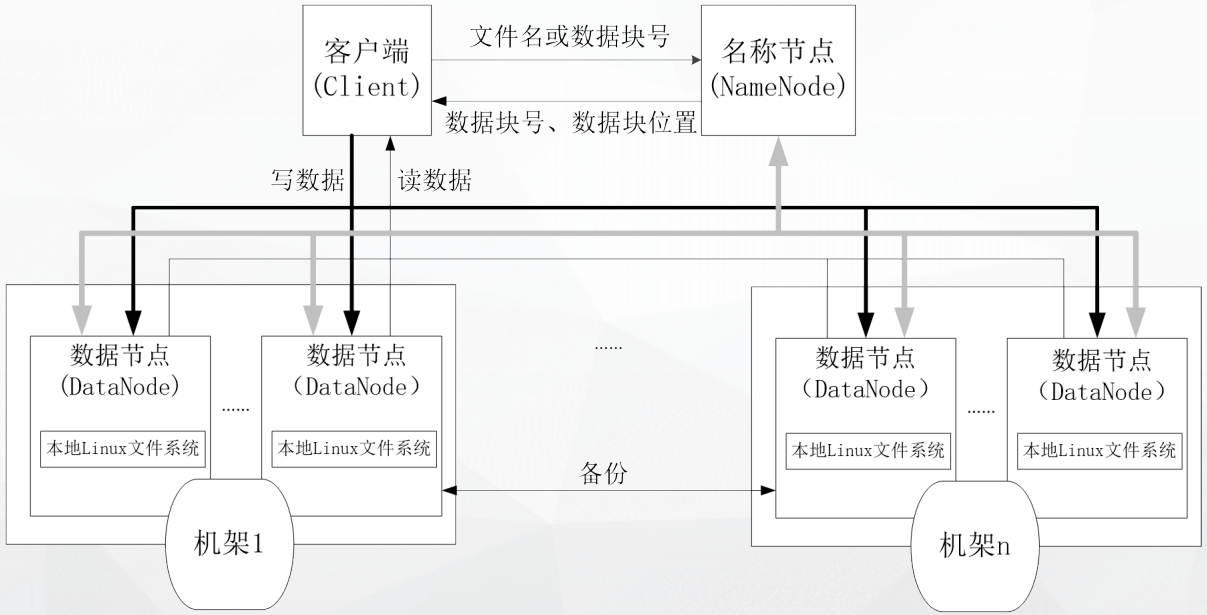


图 HDFS体系结构

6.5 分布式文件系统HDFS

6.5.1 键值数据库

6.5.2 列族数据库

6.5.3 文档数据库

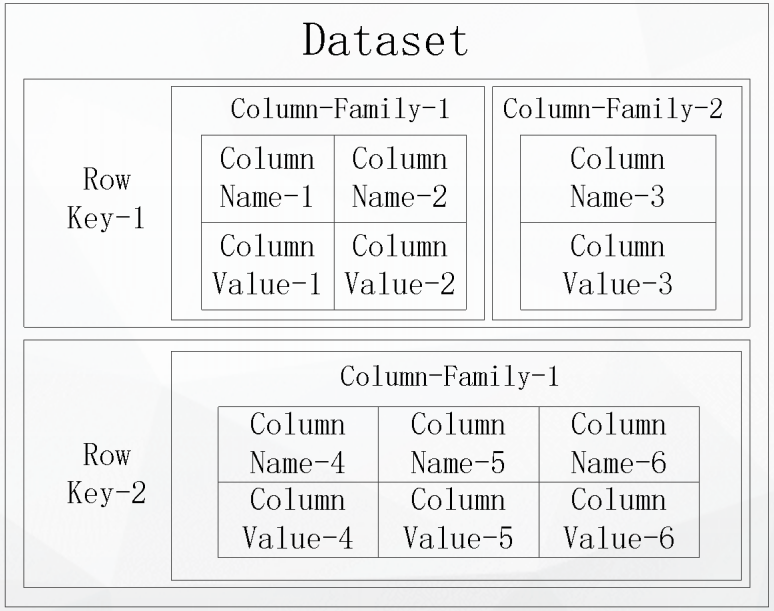
6.5.4 图数据库

6.5 NoSQL数据库

NoSQL数据库虽然数量众多，但是，归结起来，典型的NoSQL数据库通常包括键值数据库、列族数据库、文档数据库和图形数据库

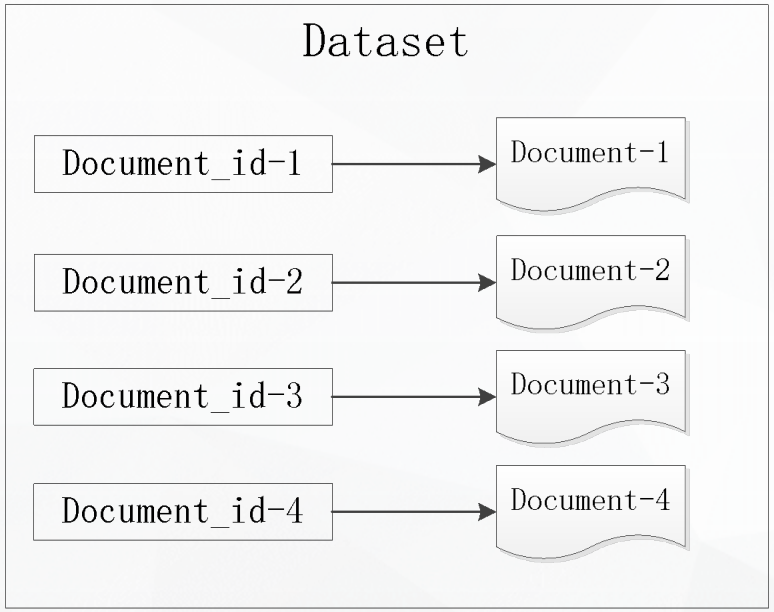
Key_1	Value_1
Key_2	Value_2
Key_3	Value_1
Key_4	Value_3
Key_5	Value_2
Key_6	Value_1
Key_7	Value_4
Key_8	Value_3

键值数据库

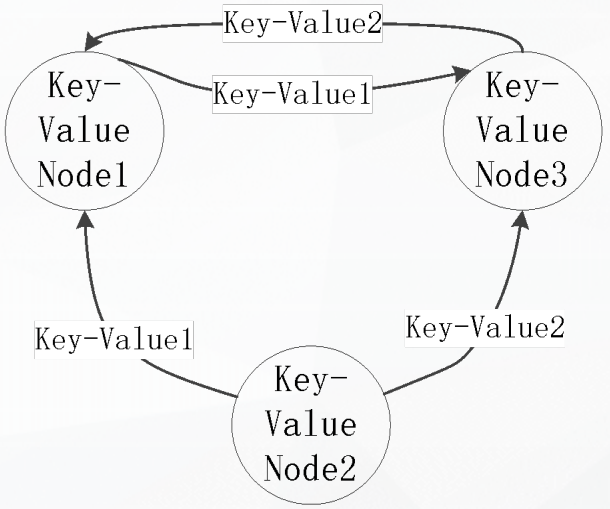


列族数据库

6.5 NoSQL数据库



文档数据库

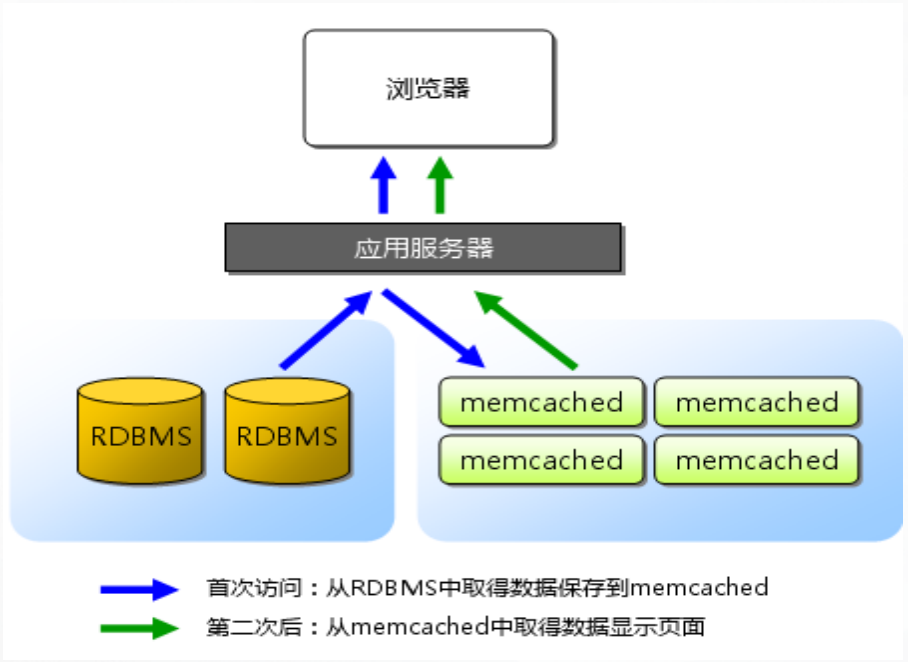


图数据库

6.5.1 键值数据库

相关产品	Redis、Riak、SimpleDB、Chordless、Scalaris、Memcached
数据模型	键/值对 键是一个字符串对象 值可以是任意类型的数据，比如整型、字符型、数组、列表、集合等
典型应用	涉及频繁读写、拥有简单数据模型的应用 内容缓存，比如会话、配置文件、参数、购物车等 存储配置和用户数据信息的移动应用
优点	扩展性好，灵活性好，大量写操作时性能高
缺点	无法存储结构化信息，条件查询效率较低
不适用情形	不是通过键而是通过值来查：键值数据库根本没有通过值查询的途径 需要存储数据之间的关系：在键值数据库中，不能通过两个或两个以上的键来关联数据 需要事务的支持：在一些键值数据库中，产生故障时，不可以回滚
使用者	百度云数据库（Redis）、GitHub（Riak）、BestBuy（Riak）、Twitter（Redis和Memcached）、StackOverFlow（Redis）、Instagram（Redis）、Youtube（Memcached）、Wikipedia（Memcached）

6.5.1 键值数据库



键值数据库成为理想的缓冲层解决方案

Redis有时候会被人们称为“强化版的Memcached”

支持持久化、数据恢复、更多数据类型

6.5.2 列族数据库

相关产品	BigTable、HBase、Cassandra、HadoopDB、GreenPlum、PNUTS
数据模型	列族
典型应用	分布式数据存储与管理 数据在地理上分布于多个数据中心的应用程序 可以容忍副本中存在短期不一致情况的应用程序 拥有动态字段的应用程序 拥有潜在大量数据的应用程序，大到几百TB的数据
优点	查找速度快，可扩展性强，容易进行分布式扩展，复杂性低
缺点	功能较少，大都不支持强事务一致性
不适用情形	需要ACID事务支持的情形，Cassandra等产品就不适用
使用者	Ebay（Cassandra）、Instagram（Cassandra）、NASA（Cassandra）、Twitter（Cassandra and HBase）、Facebook（HBase）、Yahoo!（HBase）

6.5.3 文档数据库

“文档”其实是一个数据记录，这个记录能够对包含的数据类型和内容进行“自我描述”。XML文档、HTML文档和JSON 文档就属于这一类。SequoiaDB就是使用JSON格式的文档数据库，它的存储的数据是这样的：

```
{
“ID” :1,
“NAME” : “SequoiaDB”,
“Tel” : {
        “Office” : “123123”, “Mobile” : “132132132”
      }
“Addr” : “China,GZ”
}
```

关系数据库：
必须有schema信息才能理解数据的含义
学生（学号，姓名，性别，年龄，系，年级）
（1001，张三，男，20，计算机，2002）

一个XML文档：

```
<configuration>
<property>
<name>hbase.rootdir</name>
<value>hdfs://localhost:9000/hbase</val
ue>
</property>
</configuration>
```

6.5.3 文档数据库

```
{
 “ID” :1,
 “NAME” : “SequoiaDB” ,
 “Tel” : {
        “Office” : “123123” , “Mobile” : “132132132”
      }
 “Addr” : “China,GZ”
}
```

数据是不规则的，每一条记录包含了所有的有关“SequoiaDB”的信息而没有任何外部的引用，这条记录就是“自包含”的

这使得记录很容易完全移动到其他服务器，因为这条记录的所有信息都包含在里面了，不需要考虑还有信息在别的表没有一起迁移走

同时，因为在移动过程中，只有被移动的那一条记录（文档）需要操作，而不像关系型中每个有关联的表都需要锁住来保证一致性，这样一来ACID的保证就会变得更快速，读写的速度也会有很大的提升

6.5.3 文档数据库

相关产品	MongoDB、CouchDB、Terrastore、ThruDB、RavenDB、SisoDB、RaptorDB、CloudKit、Perservere、Jackrabbit
数据模型	键/值 值（value）是版本化的文档
典型应用	存储、索引并管理面向文档的数据或者类似的半结构化数据比如，用于后台具有大量读写操作的网站、使用JSON数据结构的应用、使用嵌套结构等非规范化数据的应用程序
优点	性能好（高并发），灵活性高，复杂性低，数据结构灵活 提供嵌入式文档功能，将经常查询的数据存储在同一个文档中 既可以根据键来构建索引，也可以根据内容构建索引
缺点	缺乏统一的查询语法
不适用情形	在不同的文档上添加事务。文档数据库并不支持文档间的事务，如果对这方面有需求则不应该选用这个解决方案
使用者	百度云数据库（MongoDB）、SAP （MongoDB）、Codecademy （MongoDB）、Foursquare （MongoDB）、NBC News （RavenDB）

6.5.4 图数据库

相关产品	Neo4J、OrientDB、InfoGrid、Infinite Graph、GraphDB
数据模型	图结构
典型应用	专门用于处理具有高度相互关联关系的数据，比较适合于社交网络、模式识别、依赖分析、推荐系统以及路径寻找等问题
优点	灵活性高，支持复杂的图形算法，可用于构建复杂的关系图谱
缺点	复杂性高，只能支持一定的数据规模
使用者	Adobe（Neo4J）、Cisco（Neo4J）、T-Mobile（Neo4J）

不同类型数据库比较分析

MySQL产生年代较早，而且随着LAMP大潮得以成熟。尽管其没有什么大的改进，但是新兴的互联网使用的最多的数据库

MongoDB是个新生事物，提供更灵活的数据模型、异步提交、地理位置索引等五花十色的功能

HBase是个“仗势欺人”的大象兵。依仗着Hadoop的生态环境，可以有很好的扩展性。但是就像象兵一样，使用者需要养一头大象(Hadoop),才能驱使他

Redis是键值存储的代表，功能最简单。提供随机数据存储。就像一根棒子一样，没有多余的构造。但是也正是因此，它的伸缩性特别好。就像悟空手里的金箍棒，大可捅破天，小能成缩成针

6.6 云数据库

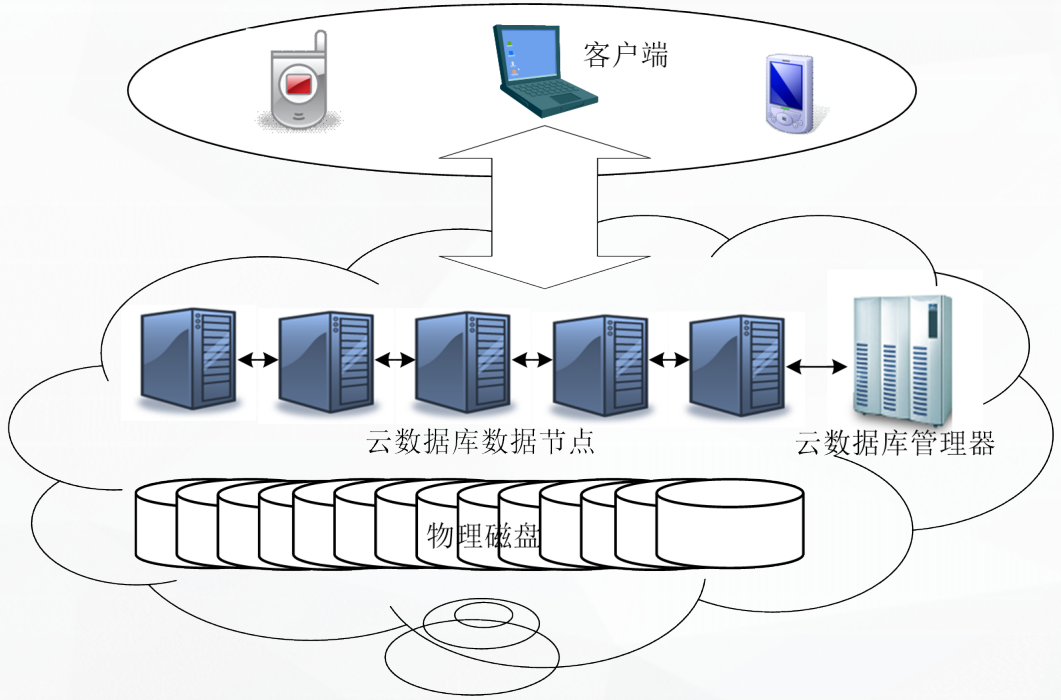
6.6.1 云数据库的概念

6.6.2 云数据库的特性

6.6.3 云数据库与其他数据库的关系

6.6.4代表性云数据库产品

6.6.1 云数据库的概念



云数据库是部署和虚拟化在云计算环境中的数据库。云数据库是在云计算的大背景下发展起来的一种新兴的共享基础架构的方法，它极大地增强了数据库的存储能力，消除了人员、硬件、软件的重复配置，让软、硬件升级变得更加容易。云数据库具有高可扩展性、高可用性、采用多租形式和支持资源有效分发等特点。

6.6.2 云数据库的特性

云数据库具有以下特性：

(1) **动态可扩展**
(2) 高可用性
(3) **较低的使用代价**
(4) **易用性**
(5) **高性能**
(6) **免维护**
(7) **安全**

表6-2 腾讯云数据库和自建数据库的比较

	自建数据库	腾讯云数据库
数据安全性	开发者自行解决，成本高昂	15种类型备份数据，保证数据安全
服务可用性		99.99%高可靠性
数据备份		0花费，系统自动多时间点数据备份
维护成本		0成本，专业团队7x24小时帮助维护
实例扩容		一键式直接扩容，安全可靠
资源利用率		按需申请，资源利用率高达99.9%
技术支持		专业团队一对一指导、QQ远程协助开发者

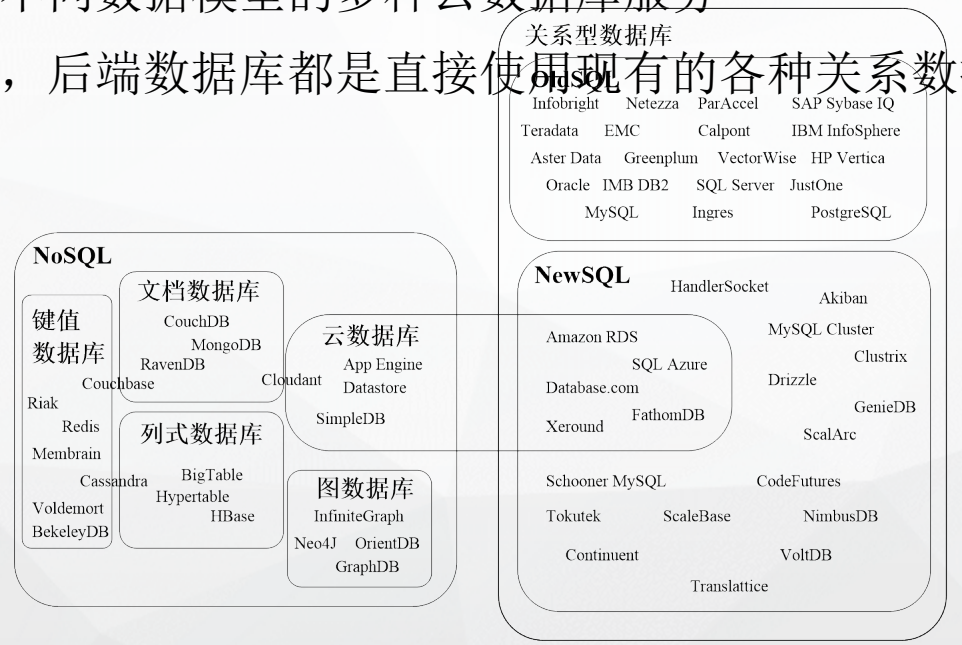
6.6.3 云数据库与其他数据库的关系

从数据模型的角度来说，云数据库并非一种全新的数据库技术，而只是以服务的方式提供数据库功能

云数据库并没有专属于自己的数据模型，云数据库所采用的数据模型可以是关系数据库所使用的关系模型（微软的SQL Azure云数据库、阿里云RDS都采用了关系模型），也可以是NoSQL数据库所使用的非关系模型（Amazon Dynamo云数据库采用的是“键/值”存储）

同一个公司也可能提供采用不同数据模型的多种云数据库服务

许多公司在开发云数据库时，后端数据库都是直接使用现有的各种关系数据库或NoSQL数据库产品



6.6.4代表性云数据库产品

表 云数据库产品

企业	产品
Amazon	Dynamo、SimpleDB、RDS
Google	Google Cloud SQL
Microsoft	Microsoft SQL Azure
Oracle	Oracle Cloud
Yahoo!	PNUTS
Vertica	Analytic Database v3.0 for the Cloud
EnerpriseDB	Postgres Plus in the Cloud
阿里	阿里云RDS
百度	百度云数据库
腾讯	腾讯云数据库

6.7 分布式数据库HBase

6.7.1 从BigTable说起

6.7.2 HBase简介

6.7.3 HBase数据模型

6.7.4 HBase系统架构

6.7.1 从BigTable说起

BigTable是一个分布式存储系统

BigTable起初用于解决典型的互联网搜索问题

建立互联网索引

1 爬虫持续不断地抓取新页面，这些页面每页一行地存储到BigTable里

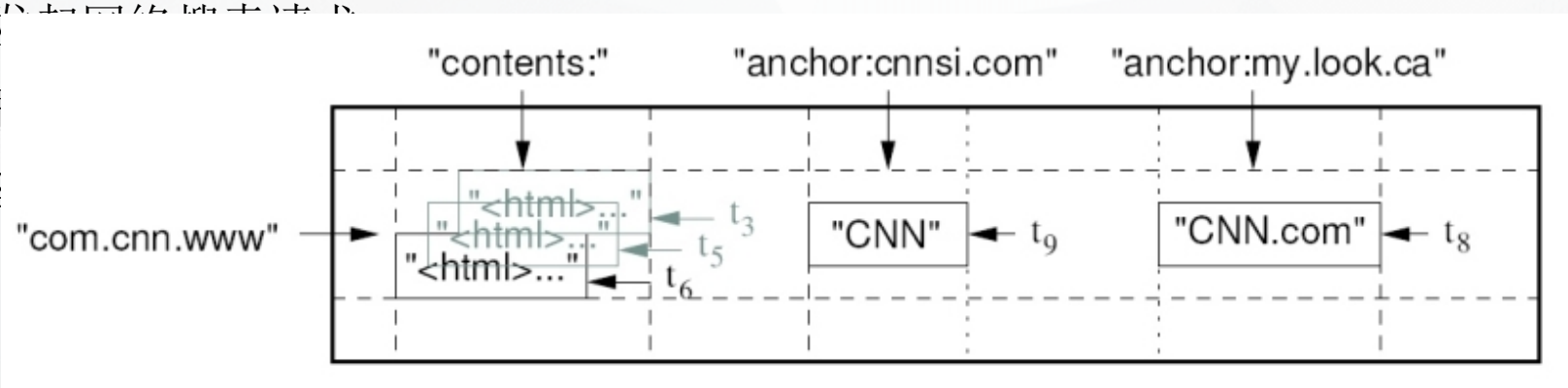
2 MapReduce计算作业运行在整张表上，生成索引，为网络搜索应用做准备

搜索互联网

3 用户

4 网络

5 搜索



网页在BigTable中的存储样例

6.7.1 从BigTable说起

BigTable是一个分布式存储系统

利用谷歌提出的MapReduce分布式并行计算模型来处理海量数据

使用谷歌分布式文件系统GFS作为底层数据存储

采用Chubby提供协同服务管理

可以扩展到PB级别的数据和上千台机器，具备广泛应用性、可扩展性、高性能和高可用性等特点

谷歌的许多项目都存储在BigTable中，包括搜索、地图、财经、打印、社交网站Orkut、视频共享网站YouTube和博客网站Blogger等

6.7.2 HBase简介

HBase是一个高可靠、高性能、面向列、可伸缩的分布式数据库，是谷歌BigTable的开源实现，主要用来存储非结构化和半结构化的松散数据。HBase的目标是处理非常庞大的表，可以通过水平扩展的方式，利用廉价计算机集群处理由超过10亿行数据和数百万列元素组成的数据表

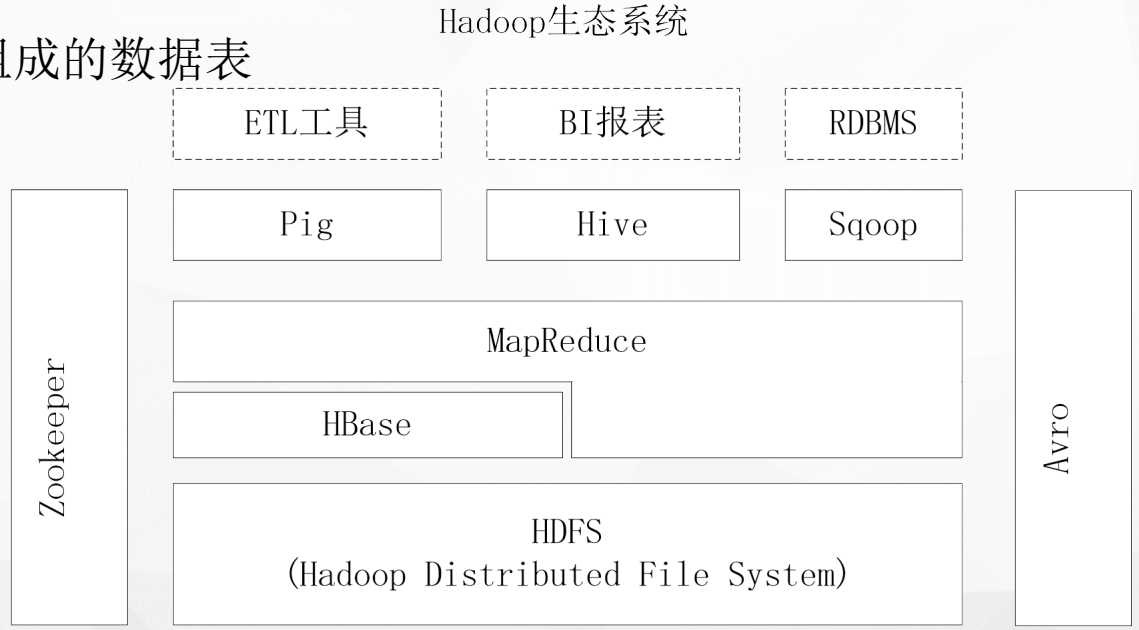


图 Hadoop生态系统中HBase与其他部分的关系

6.7.3 HBase数据模型

表：HBase采用表来组织数据，表由行和列组成，列划分为若干个列族

行：每个HBase表都由若干行组成，每个行由行键（row key）来标识。

列族：一个HBase表被分组成许多“列族”（Column Family）的集合，它是基本的访问控制单元

列限定符：列族里的数据通过列限定符（或列）来定位

单元格：在HBase表中，通过行、列族和列限定符确定一个“单元格”（cell），单元格中存储的数据没有数据类型，总被视为字节数组byte[]

时间戳：每个单元格都保存着同一份数据的多个版本，这些版本采用时间戳进行索引

6.7.4 HBase系统架构

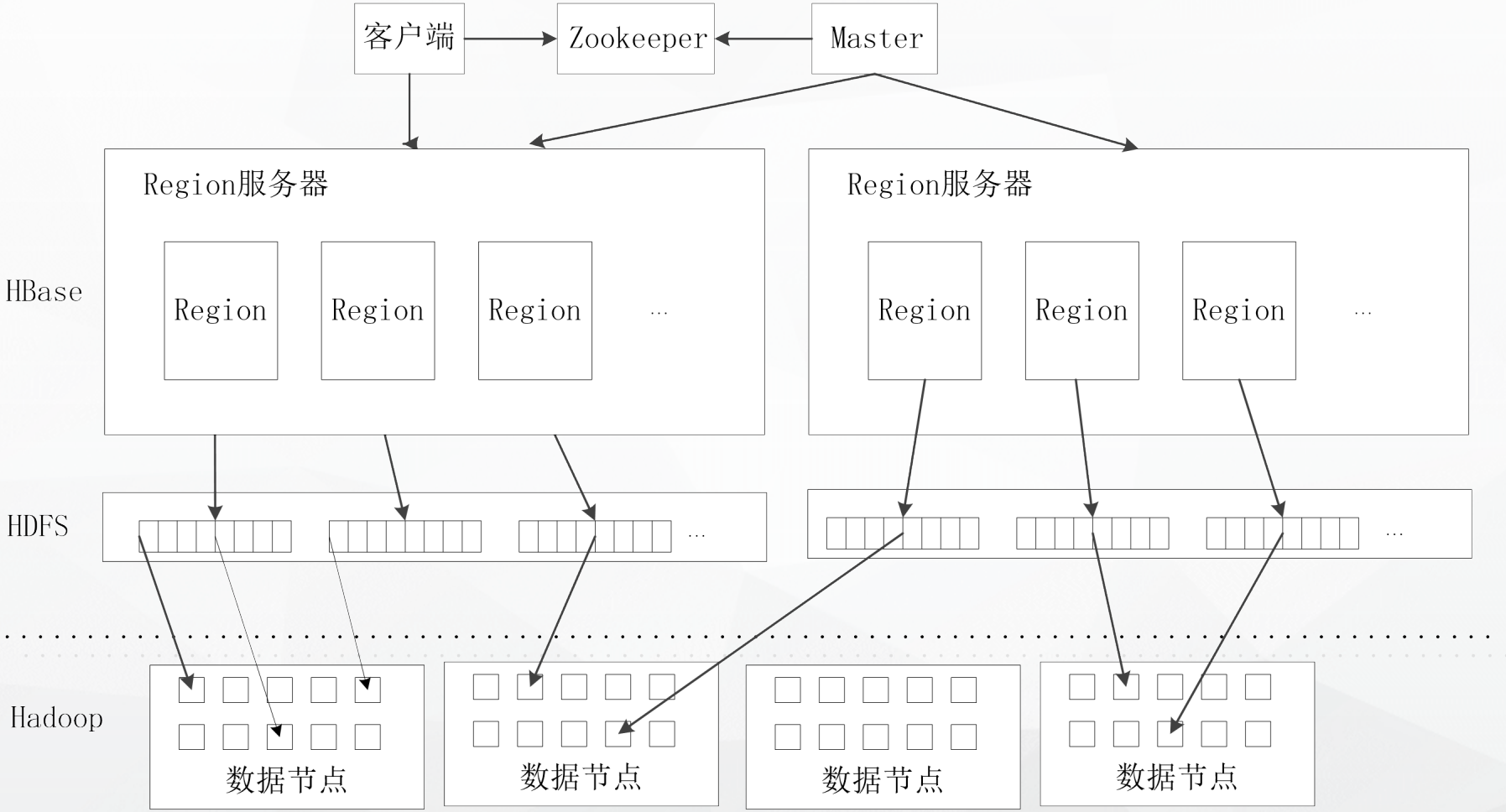


图 HBase的系统架构

6.8

Google Spanner

6.8 Google Spanner

Spanner是一个可扩展的、全球分布式的数据库，是由谷歌公司设计、开发和部署的。在最高抽象层面，Spanner就是一个数据库，把数据分片存储在许多Paxos状态机上，这些机器位于遍布全球的数据中心内。复制技术可以用来服务于全球可用性和地理局部性。客户端会自动在副本之间进行失败恢复。随着数据的变化和服务器的变化，Spanner会自动把数据进行重新分片，从而有效应对负载变化和处理失败。Spanner被设计成可以扩展到几百万个机器节点，跨越成百上千个数据中心，具备几万亿数据库行的规模。应用可以借助于Spanner来实现高可用性，通过在一个地区的内部和跨越不同的地区之间复制数据，保证即使面对大范围的自然灾害时数据依然可用。

6.8 Google Spanner

作为一个全球分布式数据库，Spanner提供了很好的特性：

第一，在数据的副本配置方面，应用可以在一个很细的粒度上进行动态控制。应用可以详细规定，哪些数据中心包含哪些数据，数据距离用户有多远（控制用户读取数据的延迟），不同数据副本之间距离有多远（控制写操作的延迟），以及需要维护多少个副本（控制可用性和读操作性能）。数据也可以被动态和透明地在数据中心之间进行移动，从而平衡不同数据中心内资源的使用。

第二，Spanner提供了读和写操作的外部一致性，以及在一个时间戳下面的跨越数据库的全球一致性的读操作。这些特性使得Spanner可以支持一致的备份、一致的MapReduce执行和原子模式变更，所有都是在全球范围内实现，即使存在正在处理中的事务也可以。

6.8 Google Spanner

下图显示了在一个Spanner的Universe中的服务器。一个Zone包括一个Zonemaster，和一百至几千个Spanserver。Zonemaster把数据分配给Spanserver，Spanserver把数据提供给客户端。客户端使用每个Zone上面的Locationproxy来定位可以为自己提供数据的Spanserver。Universemaster是一个控制台，它显示了关于Zone的各种状态信息，可以用于相互之间的调试。Placementdriver会周期性地与Spanserver进行交互，来发现那些需要被转移的数据，或者是为了满足新的副本约束条件，或者是为了进行负载均衡。

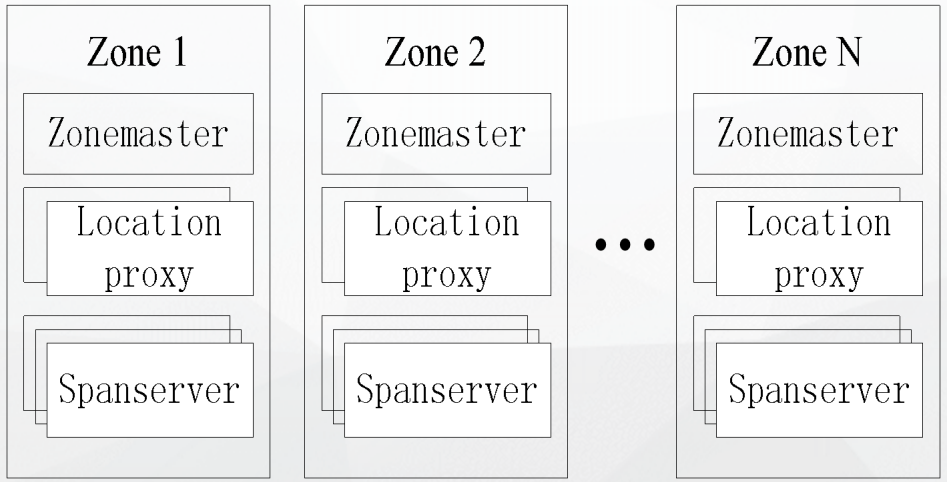


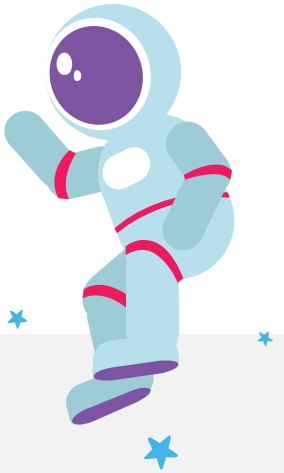
图 Spanner服务器的组织方式

6.9 本章小结

随着计算机技术的发展，数据存储与管理经历了人工管理、文件系统、数据库系统三个发展阶段。在数据库方面，经历了网状数据库、层次数据库、关系数据库、NoSQL数据库的发展过程；在文件系统方面，则由单机文件系统发展到了现在的分布式文件系统。数据存储与管理技术的不断发展，使得人类能够管理的数据越来越多，效率越来越高，对后续的大数据处理分析环节起到了很好的支撑作用。

本章内容首先介绍了文件系统、关系数据库、数据仓库、并行数据库等传统的数据存储与管理技术，然后，分布式文件和分布式数据库等大数据时代的数据存储和管理技术。需要说明的是，虽然大数据时代的新技术大行其道，但是，一些传统的数据存储与管理技术仍然在发挥着“余热”，不会立即退出大数据的“江湖”。

大数据与统计系 | 大数据系列课程



大数据

大数据与统计系数字教师网
http://hssj.wxcgi.com/