BIG DATA

4.NoSQL数据库

NoSQL数据库

- 4.1　NoSQL概述
- 4.2　NoSQL的特点与问题
- 4.3　NoSQL的主要存储方式
- 4.4　常用的NoSQL数据库

4.1 NoSQL概述

- NoSQL不是某个产品或某种单一的技术，而是一族产品及一系列不同类却有时相互关联的数据存储与处理的技术的集合

4.1.1 非结构化问题

可计算性问题是指可以使用计算机来解决的实际问题，而计算机(冯诺依曼机)本身的优势在于数值计算，很多非数值问题(比如，文字识别、图像处理等)都通过转化成为数值问题后由计算机处理。一个可以使用计算机解决的问题被定义为可以在有限步骤内被解决的问题，哥德巴赫猜想这样的问题就不属于可计算问题，因为计算机没有办法给出数学意义上的证明，所以也没有任何理由期待计算机能解决世界上所有的问题。分析某个问题的可计算性意义重大，它使得人们将资源集中在可以解决的问题集中，而不必浪费时间去解决不可能解决的问题，可以尽早转向使用计算机以外的更加有效的手段去解决。

由于计算机所能处理的问题是离散数值问题，对于连续性非线性复杂问题，需要用有限元等计算数学工具，把问题映射成计算机能够处理的离散数值计算类问题。随着计算机功能的强大，计算机进入以数据、信息处理为基础的广泛应用领域，最成功的是结构化数据、信息处理的应用。结构化计算要求每个信息和数据都可以分解为多个有意义的字段，计算机处理的不是单纯的数值，而是结构数据和信息。

随着计算机处理能力的扩大，也为处理复杂的非结构化信息提供了可能的技术。由于结构数据与信息所占份额比例小，大多数的数据都是非结构数据，对非结构数据计算的研究备受重视并已成为研究热点。

4.1.2 NoSQL 的产生

- NOSQL一词最早出现于1998年，是Carlo Strozzi开发的一个轻量、开源、不提供SQL功能的关系数据库。
- 2009年，Last.fm的Johan Oskarsson发起了一次关于分布式开源数据库的讨论，来自Rackspace的Eric Evans再次提出了NOSQL的概念，这时的NOSQL主要指非关系型、分布式、不提供ACID的数据库设计模式。
- 2009年在亚特兰大举行的"no:sql(east)"讨论会是一个里程碑，其口号是"select fun, profit from real_world where relational=false;"。因此，对NOSQL最普遍的解释是“非关联型的”，强调键-值存储和面向文档数据库的优点，而不是单纯的反对RDBMS。
- 基于2014年的收入，NOSQL市场领先企业是MarkLogic，MongoDB和Datastax。基于2015年的人气排名，最受欢迎的NOSQL数据库是MongoDB，Apache Cassandra和Redis。

Not Only SQL

NoSQL 是 Not Only SQL 的英文简写，其含义是不仅是结构化查询，所以它是不同于传统的关系型数据库的数据库管理系统的统称。NoSQL 与 SQL 显著的区别是 NoSQL 不使用 SQL 作为查询语言。其数据存储不需要固定的表格模式，也避免使用 SQL 的JOIN操作，具有横向可扩展性的特征。NoSQL 的实现具有两个重要点，一个是主要使用硬盘作为存储载体，另一个是尽可能把随机存储器当作存储载体。

关系型数据库中的表存储格式化的数据结构，每个元组字段的组成都相同，即使每个元组不是都需要所有的字段，但数据库还是为每个元组分配所有的字段，这样的结构便于表与表之间进行连接等操作，但却出现了冗余，从这一点考虑它也是关系型数据库性能瓶颈的一个因素。而非关系型数据库以键值对存储，其结构不固定，每一个元组可以有不一样的字段，而且每个元组可以根据需要增减自己的键值对，这样就不局限于固定的结构，可以减少时间和空间的开销。

关系型数据库面临的问题

1. 高并发读写的需求

Web 2.0 网站以及普通的 BBS 网站，需要高并发写请求。Web 2.0 网站要根据用户个性化信息来实时生成动态页面和提供动态信息，因为数据库并发负载非常高，往往要达到每秒上万次读写请求，所以基本上无法使用动态页面静态化技术。关系型数据库可以响应上万次 SQL 查询，但是响应上万次 SQL 写数据请求，硬盘 I/O 无法完成。这是由于对硬盘的写过程速度比读过程的速度慢的原因。

关系型数据库面临的问题

2. 高效率存储和访问的需求

对于大型的社交网站，每天产生大量的用户动态信息。对于关系数据库，在一张存储 2.5 亿条记录的表中进行 SQL 查询，效率极低，甚至不可忍受。此外，大型 Web 网站的用户登录系统，面对数以亿计的账号，关系数据库困难重重。

关系型数据库面临的问题

在基于 Web 的架构中，数据库进行横向扩展困难。当一个应用系统的用户量和访问量与日俱增时，数据库却没有办法像 Web Server 和 APPServer 那样简单地通过添加更多的硬件和服务节点来扩展性能和负载能力。对于很多需要提供 24 小时不间断服务的网站，对数据库系统进行升级和扩展困难，通常需要停机维护和数据迁移，可是停机维护随之引起效率降低。

对于 Web 2.0 网站，关系数据库的很多主要特性无法发挥。例如，很多 Web 实时系统并不要求严格的数据库事务一致性，对读一致性的要求很低，对写一致性要求也不高。因此数据库事务管理成了数据库高负载下一个沉重的负担。对关系数据库，插入一条数据之后立刻查询，虽然可以读出这个数据，但是对于多数 Web 应用，并不要求这么高的实时性。对复杂的 SQL 查询，特别是多表关联查询的需求。任何大数据量的 Web 系统都避免多个大的关联查询，以及复杂报表查询。尤其是社交类型的网站，从需求以及产品设计角度，就需要避免产生这种情况。更多地采用单表的主键查询，以及单表的简单条件分页查询，极大地弱化了 SQL 的功能，因此，关系数据库在越来越多的应用场景下已不合适，为了解决这类问题的 NoSQL 数据库应运而生。NoSQL 是非关系型数据库。NoSQL 数据存储不需要固定的表结构，也不存在连接操作。在大数据存取方面，NoSQL 具有关系型数据库无法比拟的优秀性能。

NoSQL 是非关系型的、分布式的、开源的横向可扩展的数据库。横向扩展可以将多个服务器从逻辑上看成一个实体。NoSQL 主要用于 Web 大规模的非关系型数据存储。具有模式自由、支持简易复制、简单的 API、最终的一致性(非 ACID)、大容量数据等特性。NoSQL 数据库应用最多的是键-值存储方式，还包括文档型存储方式、列存储方式、图形存储方式、面向对象存储方式与 xml 等存储方式等。

4.2 NoSQL的特点与问题

CAP 理论、BASE 模型和最终一致性是 NoSQL 数据库存在的三大基石。NoSQL 数据库满足了数据存储的横向伸缩性的需求。一些开源的 NoSQL 体系，如 Facebook 的 Cassandra、Apache 的 HBase 等也得到了广泛应用。

4.2.1 NoSQL的特点

(1)运行在 PC 服务器集群上。PC 集群扩充方便并且成本很低，避免了传统商业数据库共享操作的复杂性和成本。

(2)突破了性能瓶颈。NoSQL 可以省去将 Web 或 Java 应用和数据转换成 SQL 格式的时间，执行速度变得更快。

(3)没有过多的操作。NoSQL 能够满足企业的具体需求，没有过多的操作。

(4)支持者源于社区。因为 NoSQL 项目都是开源的，所以它们缺乏供应商提供的正式支持，得到了社区支持。

(5)弹性扩展。随着负载的增加将数据库分不到多个不同的主机上。随着每秒事务数与可用性需求的提高，以及数据库往云或虚拟环境的迁移。NoSQL 数据库从设计之初就考虑了透明横向扩展。

(6)大数据量。NoSQL 系统可以存储与处理大数据量，比如，Hadoop，超过目前最大的 RDBMS 可以管理的数据规模。

(7)灵活的数据模型。NoSQL 的键值存储与文档数据库允许应用在一个数据单元中存入任何结构。即使是定义更加严格的基于 BigTable 的 NoSQL 数据库，通常也允许创建新的字段而不致带来麻烦。

4.2.2 NoSQL 问题

1. 成熟度

与 NoSQL 相比，RDBMS 系统更加稳定，而且功能也更加丰富，但是 NoSQL 数据库正在快速发展中，许多关键性的功能还有待实现与完善。对于大多数开发者，最经常使用的是最成熟的技术而不是最新的技术。

2. 支持

用户希望能得到产品维护的支持与保证，但大多数的 NoSQL 系统都是开源项目，虽然对于每个 NoSQL 数据库，通常也有一个或多个小型公司对它们提供支持，但是在支持的范围与资源或可信度方面，不能与 Oracle，Microsoft 或 IBM 等大型公司相比较。

4.2.2 NoSQL 问题

3. 分析和商务智能化

NoSQL 数据库现在已经可以满足现代的 Web2.0 应用程序的高度的可扩展性的要求。这就导致大多数功能都是面向这些应用程序而设计。但是，在一个应用程序中，具有商业价值的数据已经超出了一个标准的 Web 应用程序需要的“插入—读取—更新—删除”的范畴。在数据库中进行商业信息的挖掘可以提高企业的效率和竞争力，所以商务智能始终是一个至关重要的 IT 问题。NoSQL 数据库几乎没有提供专用的查询和分析工具。即使是一个简单的查询，也要求操作者具有很高超的编程技术，而且，常用的商务智能工具无法与 NoSQL 连接。

像 Hadoop 中的 HIVE 或 PIG 一些解决方案能够提供帮助，可以使访问 Hadoop 集群中的数据变得更加容易。

4. 管理

NoSQL 的设计目标是提供一个零管理的解决方案，目前还远没有达到这个目标。安装 NoSQL 需要很多技巧，维护也需要付出很多的努力。

5. 专业知识

SQL 开发者对 RDBMS 的概念和编程方法很熟悉，相反地，几乎每一个 NoSQL 开发者都正处于学习状态中。虽然这种情况会随着时间的推移而改变，但是现在找到一些有经验的一个 NoSQL 程序员或管理员要比找到 RDBMS 专家更为困难。

4.3 NoSQL的主要存储方式

- 文档式存储
- 列式存储
- 键值式存储
- 对象式存储
- 图形式存储
- XML存储

4.3.1 键值存储方式

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）

4.3.2 文档存储方式

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）

4.3.3 列存储方式

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）

4.3.4 图形存储方式

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）

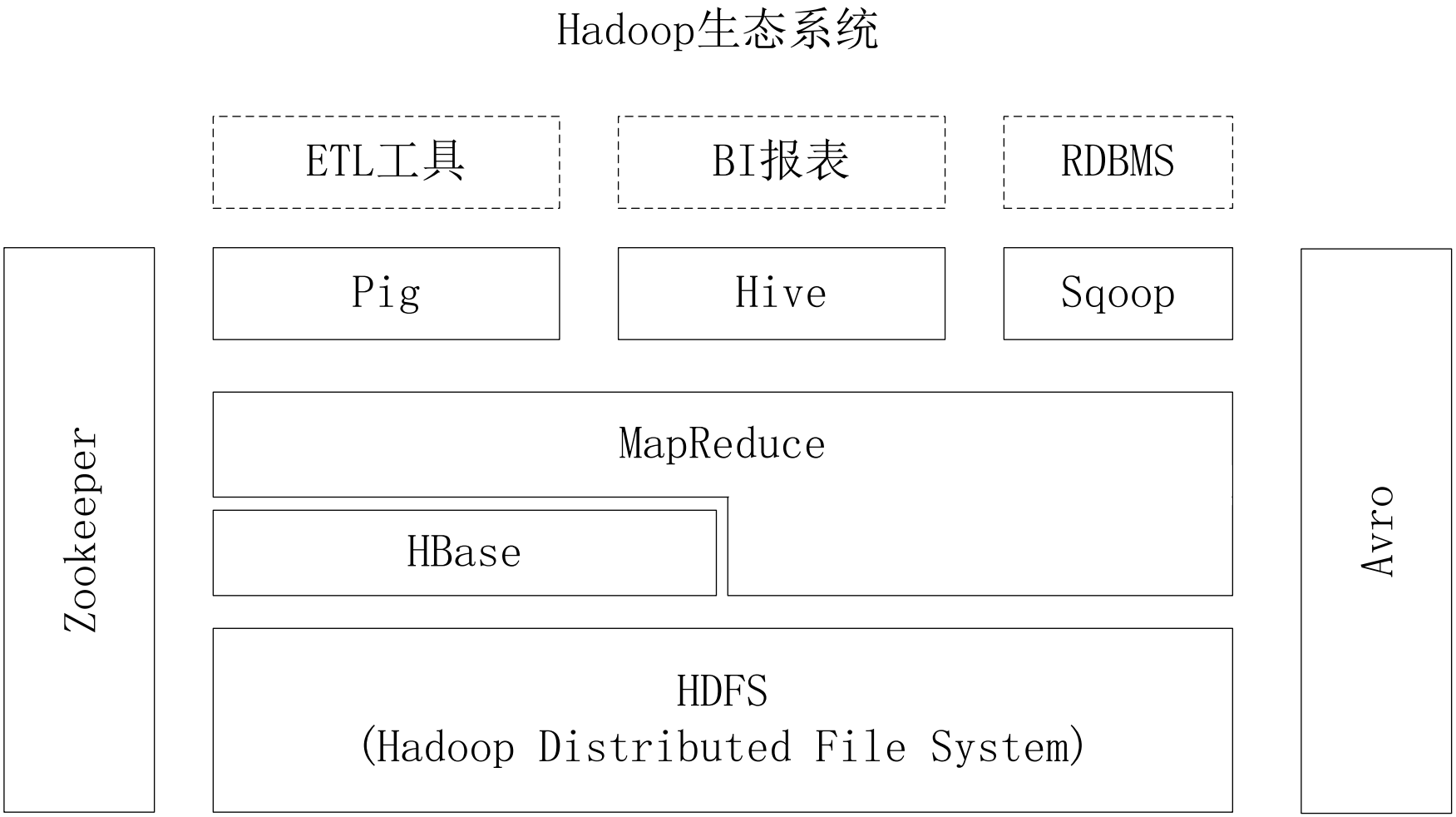
4.3.5 各种典型的存储方式所对应的NoSQL 数据库

存储方式	典型的 NoSQL 数据库
文档存储方式	MongoDB
列存储方式	Hbase Cassandra Hypertable
键-值存储方式	Redis Tokyo Cabinet/ Tokyo Tyrant Flae
对象存储方式	Db4o Versant
图形存储方式	Neo4J FlockDB
XML 存储方式	Berkeley DB XML BaseX

数据库主要分为 BigTable(Google)和 Dynamo(Amazon)两种系列，下面以 NoSQL 中目前最受欢迎的 MongoDB 为例进行说明。

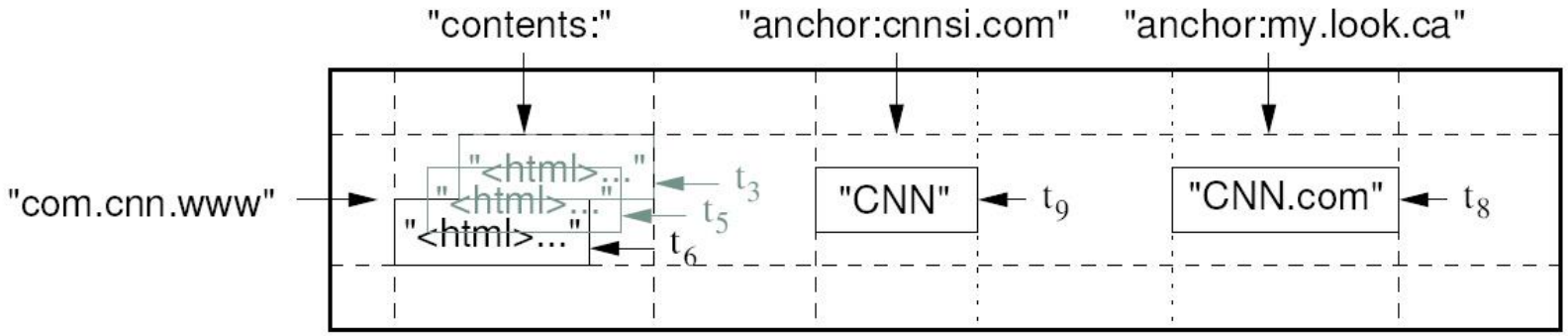
可伸缩性是分布式系统的一个重要标志。分布式系统设计遵循 CAP 定理，从 EJB 到 MySQL，都试图实现 CAP 三个核心需求的完美结合，关系数据库主要满足一致性和可用性，特别是要求极其严格的一致性，主要是通过数据库锁或 JTA/JDBC 事务实现，过于苛刻的一致性要求，很难实现分区错性。而 NoSQL 数据库，则分为满足一致性和分区错性(例如，MongoDB 等 NoSQL 数据库)及可用性和分区错性(例如，Dynamo Cassandra 等 NoSQL 数据库)两种。

HBase简介


Hadoop生态系统
ETL工具
BI报表
RDBMS
Pig
Hive
Sqoop
Zookeeper
MapReduce
HBase
Avro
HDFS
(Hadoop Distributed File System)

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 用户发起网络搜索请求
  - 4 网络搜索应用查询建立好的索引，从BigTable得到网页
  - 5 搜索结果提交给用户



网页在BigTable中的存储样例

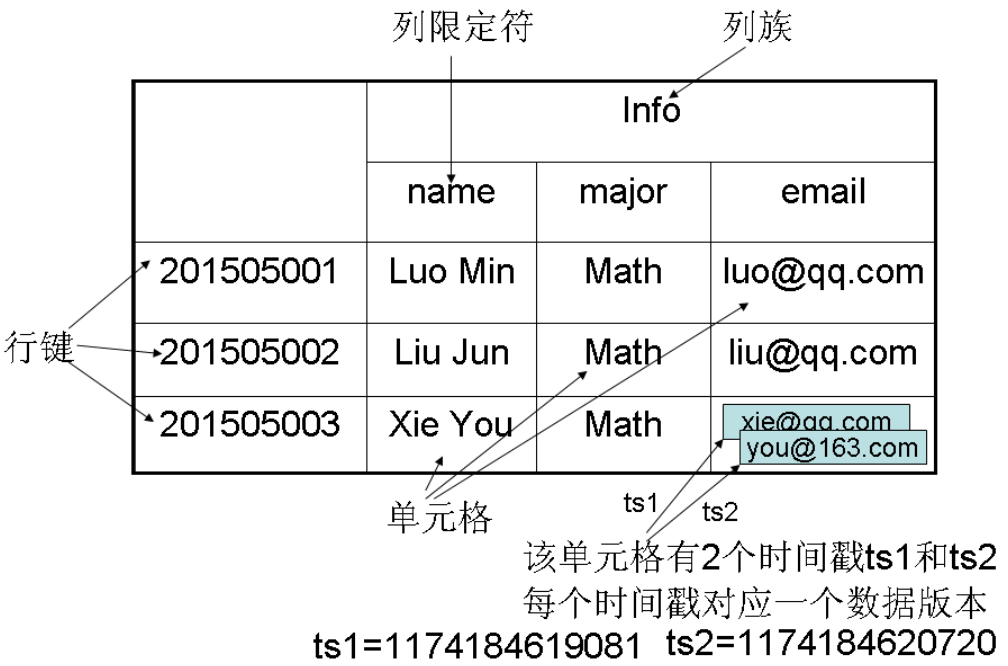
HBase和BigTable的关系

• 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

	BigTable	**HBase**
文件存储系统	GFS	HDFS
海量数据处理	MapReduce	Hadoop MapReduce
协同服务管理	Chubby	Zookeeper

数据模型相关概念

- 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

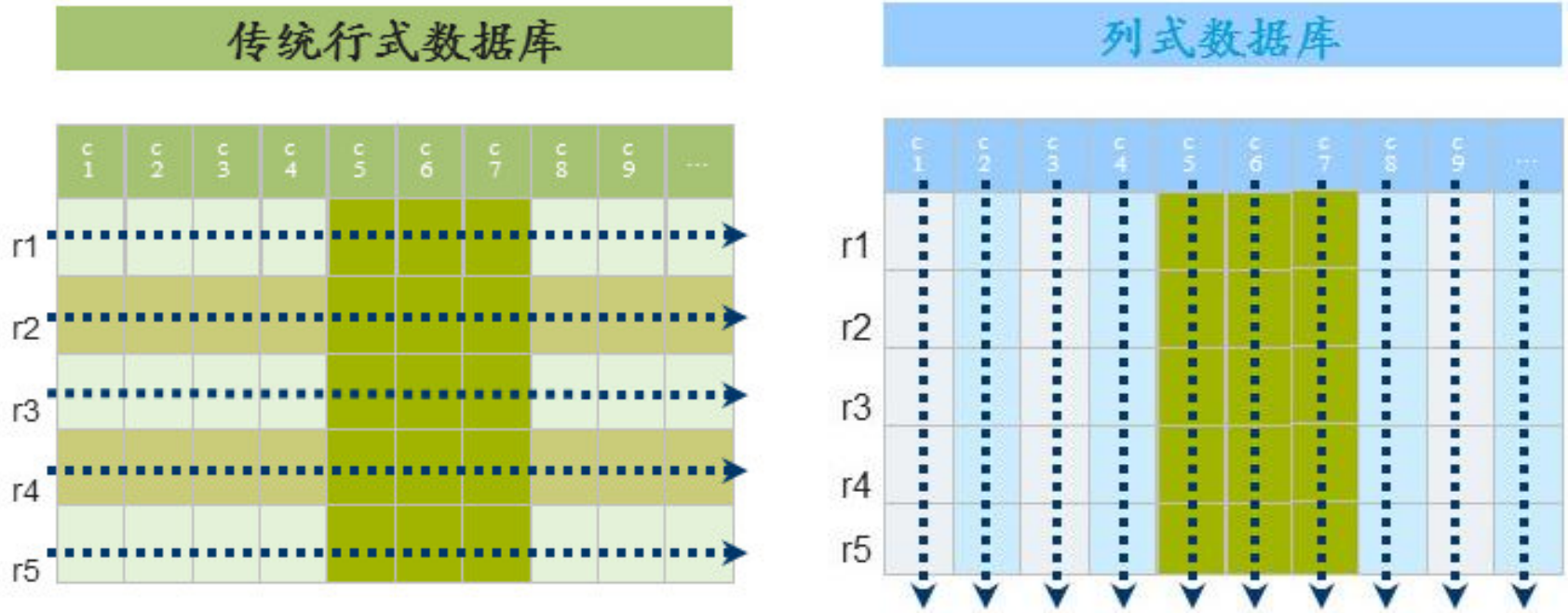


数据模型相关概念

- HBase中需要根据行键、列族、列限定符和时间戳来确定一个单元格，因此，可以视为一个“四维坐标”，即[行键, 列族, 列限定符, 时间戳]

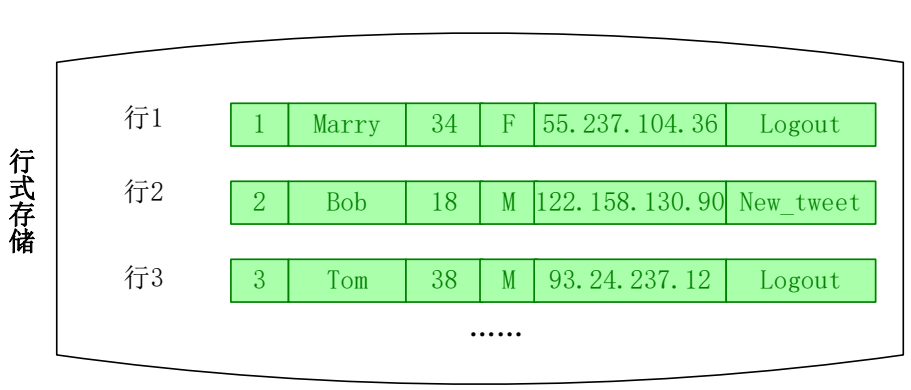
键	值
[“201505003”, “Info”, “email”, 1174184619081]	“xie@qq.com”
[“201505003”, “Info”, “email”, 1174184620720]	“you@163.com”

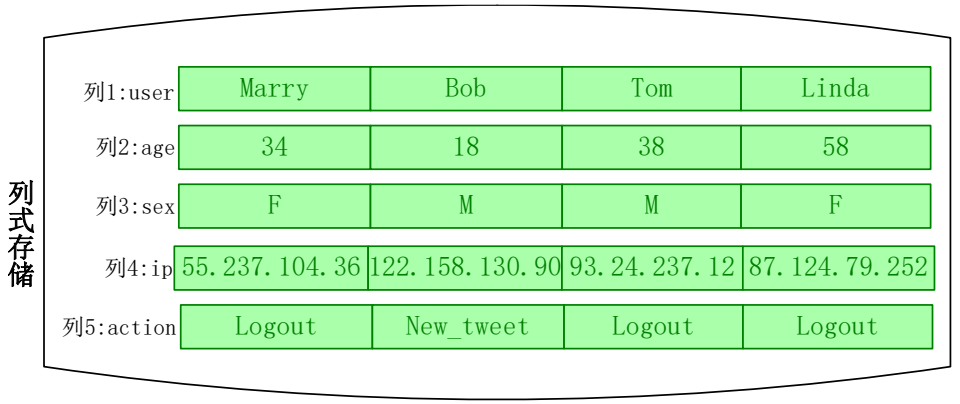
面向列的存储


传统行式数据库
c 1
c 2
c 3
c 4
c 5
c 6
c 7
c 8
c 9
...
r1
r2
r3
r4
r5
列式数据库
c 1
c 2
c 3
c 4
c 5
c 6
c 7
c 8
c 9
...
r1
r2
r3
r4
r5

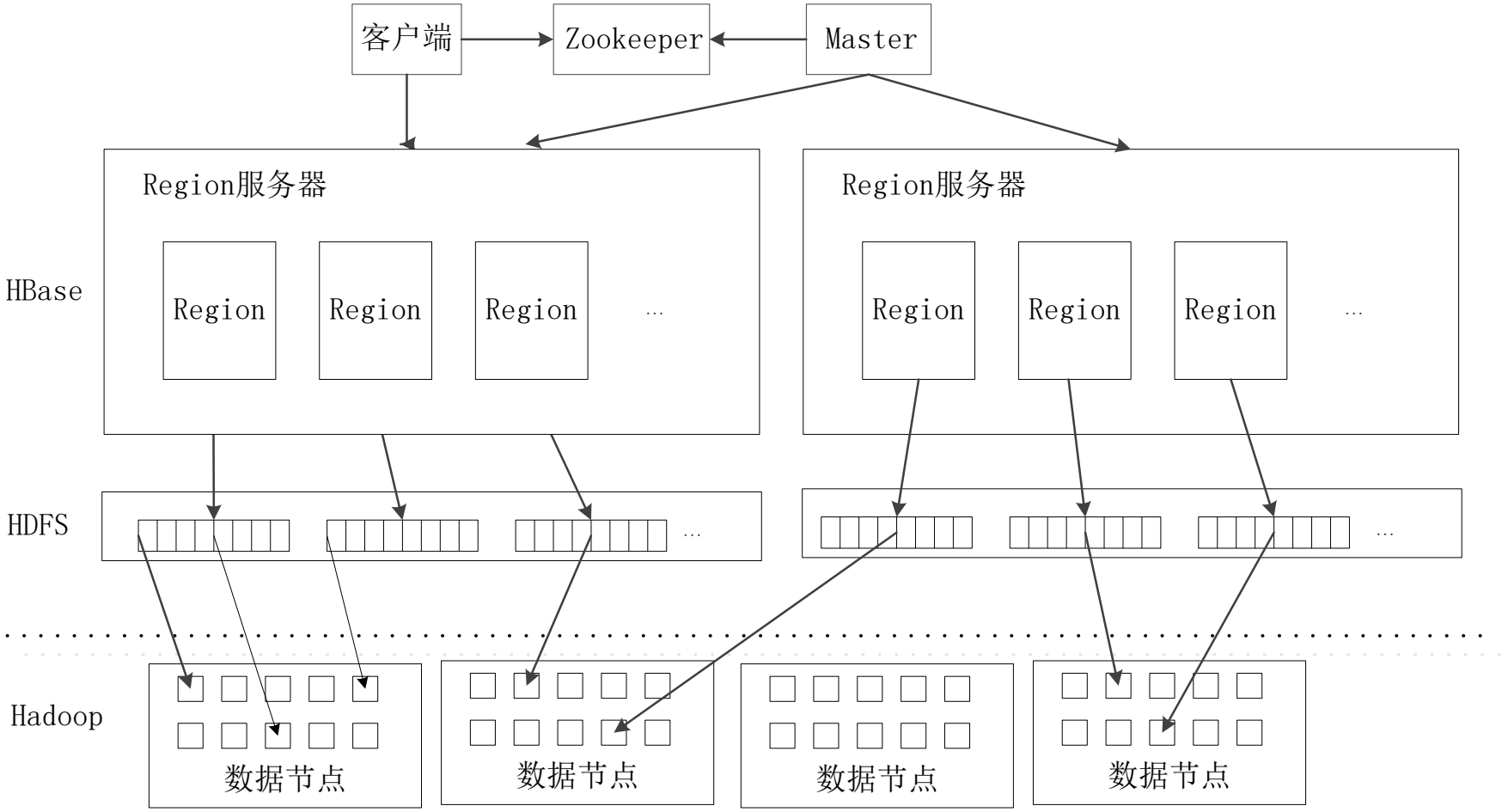
面向列的存储

Log					
Log_id	user	age	sex	ip	action
1	Marry	34	F	55.237.104.36	Logout
2	Bob	18	M	122.158.130.90	New_tweet
3	Tom	38	M	93.24.237.12	Logout
4	Linda	58	F	87.124.79.252	Logout





HBase系统架构


客户端
Zookeeper
Master
Region服务器
Region服务器
HBase
Region
Region
Region
...
Region
Region
Region
...
HDFS
...
...
Hadoop
数据节点
数据节点
数据节点
数据节点

HBase的安装与配置

- 选择合适的版本
- 在linux上先完成配置Hadoop
- 在linux上配置zookeeper
- 在linux上配置hbase
- 启动关闭Hadoop和HBase的顺序一定是：
  - 启动zookeeper —>启动Hadoop—>启动HBase—>关闭HBase—>关闭Hadoop —>关闭zookeeper

常用Shell命令

- create：创建表
- list：列出HBase中所有的表信息
- put：向表、行、列指定的单元格添加数据
- 一次只能为一个表的一行数据的一个列添加一个数据
- scan：浏览表的相关信息
- get：通过表名、行、列、时间戳、时间范围和版本号来获得相应单元格的值

MongoDB简介

- MongoDB 是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。
- 在高负载的情况下，添加更多的节点，可以保证服务器性能。
- MongoDB 旨在为WEB应用提供可扩展的高性能数据存储解决方案。
- MongoDB 将数据存储为一个文档，数据结构由键值(key=>value)对组成。MongoDB 文档类似于 JSON 对象。字段值可以包含其他文档，数组及文档数组。

MongoDB主要特点

- 提供了一个面向文档存储，操作起来比较简单和容易
- 可以设置任何属性的索引来实现更快的排序
- 具有较好的水平可扩展性
- 支持丰富的查询表达式，可轻易查询文档中内嵌的对象及数组
- 可以实现替换完成的文档（数据）或者一些指定的数据字段
- MongoDB中的Map/Reduce主要是用来对数据进行批量处理和聚合操作
- 支持各种编程语言:RUBY，PYTHON，JAVA，C++，PHP，C#等语言
- MongoDB安装简单

MongoDB概念解析

- 在mongodb中基本的概念是文档、集合、数据库

SQL术语/概念	MongoDB术语/概念	解释/说明
database	database	数据库
table	collection	数据库表/集合
row	document	数据记录行/文档
column	field	数据字段/域
index	index	索引
table joins		表连接,MongoDB不支持
primary key	primary key	主键,MongoDB自动将_id字段设置为主键

MongoDB概念解析

id	user_name	email	age	city
1	Mark Hanks	mark@abc.com	25	Los Angeles
2	Richard Peter	richard@abc.com	31	Dallas

```
{
        "_id": ObjectId("5146bb52d852427006000lf3"),
        "age": 25,
        "city": "Los Angeles",
        "email": "mark@abc.com",
       "user_name": "Mark Hanks "
}
{          "_id": ObjectId("5146bb52d852427006000lf2"),
           "age": 31,
           "city": "Dallas",
           "email": "richard@abc.com",
           "user_name": "Richard Peter "
}
```

MongoDB概念解析

• 举例2：在一个关系型数据库中，一篇博客（包含文章内容、评论、评论的投票）会被打散在多张数据表中。在文档数据库MongoDB中，能用一个文档来表示一篇博客， 评论与投票作为文档数组，放在正文主文档中。这样数据更易于管理，消除了传统关系型数据库中影响性能和水平扩展性的“JOIN”操作。

author:

pid	tid	name
1	1	Jane

blogposts:

tid	cid	title
1	1	"MyFirstPost"
1	2	"MyFirstPost"

comments:

cid	by	text
1	"Abe"	"First"
2	"Ada"	"Good post"

MongoDB概念解析

- 关系数据库中的其中一条记录，在文档数据库MongoDB中的存储方式类似如下：

```
{
“id”:1,
“author”:”Jane”,
“blogposts”:{
    “tile”: ”MyFirstPost”,
    “comment”: {
        “by”: ”Ada”,
        ”text”: ”Good post”
        }
    }
}
```

MongoDB概念解析

- 数据库
  - 一个mongodb中可以建立多个数据库。
  - MongoDB的默认数据库为"db"，该数据库存储在data目录中。
  - MongoDB的单个实例可以容纳多个独立的数据库，每一个都有自己的集合和权限，不同的数据库也放置在不同的文件中。
- 文档
  - 文档是一个键值(key-value)对(即BSON)。MongoDB 的文档不需要设置相同的字段，并且相同的字段不需要相同的数据类型，这与关系型数据库有很大的区别，也是 MongoDB 非常突出的特点。

MongoDB概念解析

- 下表列出了 RDBMS 与 MongoDB 对应的术语：

RDBMS	MongoDB
数据库	数据库
表格	集合
行	文档
列	字段
表联合	嵌入文档
主键	主键 (MongoDB 提供了 key 为 _id)

数据库服务和客户端	
Mysqld/Oracle	mongod
mysql/sqlplus	mongo

集合

- 集合就是 MongoDB 文档组，类似于 RDBMS （关系数据库管理系统：Relational Database Management System)中的表格。
- 集合存在于数据库中，集合没有固定的结构，这意味着你在对集合可以插入不同格式和类型的数据，但通常情况下我们插入集合的数据都会有一定的关联性。
- 比如，我们可以将以下不同数据结构的文档插入到集合中：

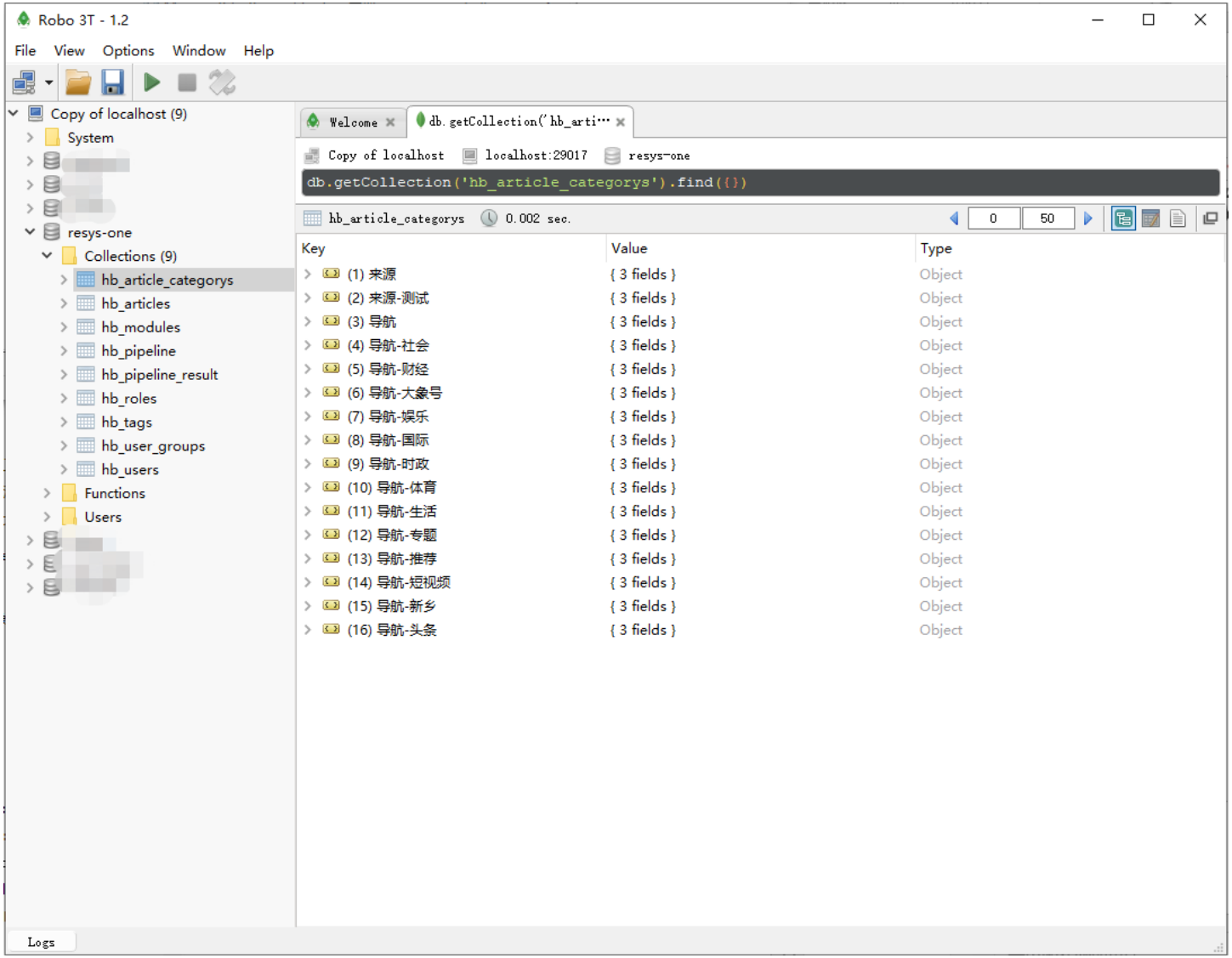
MongoDB 数据类型

数据类型	描述
String	字符串。存储数据常用的数据类型。在 MongoDB 中，UTF-8 编码的字符串才是合法的。
Integer	整型数值。用于存储数值。根据你所采用的服务器，可分为 32 位或 64 位。
Boolean	布尔值。用于存储布尔值（真/假）。
Double	双精度浮点值。用于存储浮点值。
Min/Max keys	将一个值与 BSON（二进制的 JSON）元素的最低值和最高值相对比。
Arrays	用于将数组或列表或多个值存储为一个键。
Timestamp	时间戳。记录文档修改或添加的具体时间。
Object	用于内嵌文档。
Null	用于创建空值。
Symbol	符号。该数据类型基本上等同于字符串类型，但不同的是，它一般用于采用特殊符号类型的语言。
Date	日期时间。用 UNIX 时间格式来存储当前日期或时间。你可以指定自己的日期时间：创建 Date 对象，传入年月日信息。
Object ID	对象 ID。用于创建文档的 ID。
Binary Data	二进制数据。用于存储二进制数据。
Code	代码类型。用于在文档中存储 JavaScript 代码。
Regular expression	正则表达式类型。用于存储正则表达式。

安装MongoDB

- MongoDB提供了可用于32位和64位系统的预编译二进制包，你可以从MongoDB官网下载安装，MongoDB预编译二进制包下载地址：http://www.mongodb.org/downloads
- 启动 MongoDB服务：在MongoDB安装目录的bin目录下执行'mongod'

MongoDB应用展示



增删改查

- 插入文档

```
>db.col.insert({username:"zhangsan", passwd:"123456" })
```

- 查找文档

```
>db.col.find({username: "zhangsan" })
```

- 更新文档

```
>db.col.update({username: "zhangsan" },
{$set:{'title':'lisi'}})
```

- 删除文档

```
>db.col.remove({username: "zhangsan" })
```

4.4 常用的NoSQL数据库

- Cassandra
- Lucene/Solr
- Riak
- CouchDB
- Neo4J
- racle 的NoSQL
- Hadoop 的HBase
- Bigtable/ Accumulo/ Hypertable
- DynamoDB
- MongoDB

4.4.1 Cassandra

Cassandra最初由 Facebook 开发，后来成了 Apache 开源项目，它是一个网络社交云计算方面理想的数据库。它集成了其他的流行工具，如 Solr 现在已经成为一个完全成熟的大型数据存储工具。Cassandra 是一个混合型的非关系的数据库，类似于 Google 的 BigTable。其主要功能比 Dynomite(分布式的 Key-Value 存储系统)更丰富，但支持度却不如文档存储 MongoDB。Cassandra 的主要特点就是它不是一个数据库，而是由一堆数据库节点共同构成的一个分布式网络服务，对 Cassandra 的一个写操作，会被复制到其他节点上去，而对 Cassandra 的读操作，也会被路由到某个节点上面去读取。在最近的一次测试中，Netflix 建立了一个 288 个节点的集群。

4.4.2 Lucene/Solr

Lucene是 Apache 软件基金会 4 jakarta 项目组的一个子项目，这是一个开放源代码的全文检索引擎工具包，就是它不是一个完整的全文检索引擎，而是一个全文检索引擎的架构。主要使用 Lucene 来检索大量的文本块，它采用了与其他 NoSQL 数据存储相似的模型。如果说查询并不是仅仅局限于精确的匹配，而是寻找出那些出现在块中的字或者字段的话， Lucene 是最好的查询方式。

4.4.3 Riak

Riak是由技术公司开发的一个类似 Dynamo 的分布式键值系统。采用了分布式水平扩展性与高容错性。可以使用 JavaScript 或者 Erlang 来进行 Map/Reduce 查询，它们将查询每个节点，收集结果，而且可以重复，如果需要使用的结果进行重新进行搜寻。该系统还提供全文索引，同时还提供一个控制面板，可以查看集群的信息。

4.4.4 CouchDB

CouchDB是用 Erlang 开发的面向文档的数据库系统，它不是一个传统的关系数据库，而是面向文档的数据库，其数据存储方式类似 lucene 的 index 文件格式，CouchDB 最大的特点是一个面向 Web 应用的新一代存储系统。作为一个分布式的数据库，CouchDB 可以把存储系统分布到 *n* 台物理的节点上，并且很好地协调和同步节点之间的数据读写一致性。CouchDB 支持 REST API，可以让用户使用 JavaScript 来操作 CouchDB 数据库，也可以用 JavaScript 编写查询语句，如果利用 AJAX 技术结合 CouchDB 开发出来的 CMS 系统将更为简单和方便。

CouchDB 是 Couchbase 的子集，它可以提供缓存功能，更好的分片，增量查询，更好的索引和一些其他的功能。其实 Couchbase 与 CouchDB 也紧密相关，Couchbase 产品包含了 CouchDB 的一个副本。

CouchDB 是用 Erlang 开发的面向文档的数据库系统，其数据存储方式类似 Lucene 的 Index 文件格式。

4.4.5 Neo4J

大多数的 NoSQL 数据库只是键和值的一个灵活的捆绑，不过Neo4J存储的是对象之间的关系，或者说这种结构就是数学中的图。Neo4J 是一个面向网络(图)的数据库，也就是说，它是一个嵌入式的、基于磁盘的、具备完全的事务特性的 Java 持久化引擎，但是它将结构化数据存储在网络上，而不是表中，当然也可以把 Neo4J 看成一个高性能的图引擎，该引擎具有成熟和健壮的数据库的所有特性。该工具包括很多有关搜索和分析的关系的算法，它能够帮助寻找谁是我的朋友，或者寻找朋友的朋友。图的遍历算法，可以节省很多指针查询的麻烦。

4.4.6 Oracle 的NoSQL

Oracle 的NoSQL Database是 Oracle 开发的产品，将键/值对拆分在整个节点集上，这样的优势在于提供了一个灵活的事务保护措施，进而可以确保从数据在节点上等待存储开始，到通过网络被成功备份结束。

Oracle 的NoSQL Database是 Big Data Appliance 的其中一个组件，Big Data Appliance 是集成了 Hadoop、NoSQL Database、Oracle 数据库 Hadoop 适配器、Oracle 数据库 Hadoop 装载器及 R 语言的系统。

4.4.7 Hadoop 的HBase

HBase(Hadoop Database)是一个高可靠性、高性能、面向列的、可伸缩的分布式存储系统，利用 HBase 可在廉价 PC 服务器上搭建起大规模结构化存储集群。HBase 是 Google Bigtable 的开源实现，类似 Google Bigtable 利用 GFS 作为其文件存储系统，HBase 利用 Hadoop HDFS 作为其文件存储系统。Google 运行 MapReduce 来处理 Bigtable 中的海量数据，HBase 同样利用 Hadoop MapReduce 来处理 HBase 中的海量数据。

虽然 Hadoop 及其所有的工具构成了分布计算开发平台，但 Hadoop 也包括数据库，在 HBase 中也是通过节点来传播数据。Hadoop 的 Map /Reduce 的架构非常适合于复杂的计算任务或查询工作。

4.4.8 Bigtable/ Accumulo/ Hypertable

Bigtable 是非关系的数据库，是一个稀疏的、分布式的、持久化存储的多维度排序映射。Bigtable 的设计目的是可靠的处理 PB 级别的数据，并且能够部署到上千台机器上。Bigtable 已经实现了适用性广泛、可扩展、高性能和高可用性。Bigtable 已经在超过 60 个 Google 的产品和项目上得到了应用，包括 Google Analytics、GoogleFinance、Orkut、Personalized Search、Writely 和 GoogleEarth。

4.4.9 DynamoDB

DynamoDB是亚马逊的键值存储方式的存储平台，具有良好的可用性和扩展性。读写访问中 99.9%的响应时间都在 300ms 内。DynamoDB 的 NoSQL 解决方案，也是使用键值对存储的方式，平且通过服务器把所有的数据存储在 SSD 上的三个不同的区域。如果有更高的传输需求，DynamoDB 也可以在后台添加更多的服务器。

4.4.10 MongoDB

MongoDB 是一个介于关系数据库和非关系数据库之间的产品，是功能最丰富、最像关系数据库的非关系数据库。它支持的数据结构非常松散，可以存储比较复杂的数据类型。MongoDB 是一个高性能、开源、无模式自由的文档型数据库，使用 C++开发，是当前 NoSQL 数据库中最常用的一种数据库。在许多情况下，可以替代传统的关系型数据库，采用了键值存储方式。

MongoDB 用 C++开发，主要解决的是海量数据的访问效率问题。当数据量达到 50GB 以上的时，MongoDB 的数据库访问速度是 MySQL 的 10 倍以上。MongoDB 的并发读写效率不是特别出色，性能测试表明，大约每秒可以处理 0.5 万到 1.5 万次读写请求。MongoDB 还自带了一个分布式文件系统 GridFS，可以支持海量的数据存储。MongoDB 也有一个 MongoMapper 项目，是模仿 Merb 的 DataMapper 编写的 MongoDB 接口，使用起来非常简单，几乎和 DataMapper 一样，功能非常强大。

本 章 小 结

面对越来越多的非结构化数据的存储与管理，NoSQL 数据库应运而生。在大数据中，85%以上是非结构化数据，因此，对于大数据应用，NoSQL 数据库不可缺少。本章主要介绍了非结构化问题、NoSQL 的产生的背景、特点与优势，在此基础之上，还介绍了 NoSQL 的主要类型，尤其对键值存储方式、文档式存储方式、列存储方式等作了较详细的说明。最后，概括性介绍了常用的 NoSQL 数据库。