大数据离线阶段

01

一、 课程计划

目录

一、 课程计划.....2
二、 大数据课程导论.....4
1. 数据与数据分析.....4
2. 数据分析作用.....5
2.1. 现状分析（分析当下的数据）.....5
2.2. 原因分析（分析过去的数据）.....6
2.3. 预测分析（结合数据预测未来）.....6
3. 数据分析基本步骤.....7
3.1. 明确分析目的和思路.....7
3.2. 数据收集.....9
3.3. 数据处理.....10
3.4. 数据分析.....11
3.5. 数据展现.....12
3.6. 报告撰写.....13
4. 大数据时代.....14
4.1. 概述.....14
4.2. 海量数据的挑战.....15
4.3. 大数据的特点（5V 特征）.....16
4.4. 大数据的应用场景.....17
5. 分布式技术.....21
5.1. 什么是分布式.....21
5.2. 常用分布式方案.....23
5.3. 分布式、集群.....24
三、 Apache ZooKeeper.....25
1. Zookeeper 基本知识.....25
1.1. ZooKeeper 概述.....25
1.2. ZooKeeper 特性.....26
1.3. ZooKeeper 集群角色.....27
1.4. ZooKeeper 集群搭建.....28
2. ZooKeeper shell.....29
2.1. 客户端连接.....29
2.2. shell 基本操作.....29
3. ZooKeeper 数据模型.....32
3.1. 数据结构图.....33

3.2. 节点类型.....34
3.3. 节点属性.....35
4. ZooKeeper Watcher（监听机制）.....36
5. Zookeeper 典型应用.....39
5.1. 数据发布/订阅.....39
5.2. 提供集群选举.....40
5.3. 分布式锁.....40

二、 大数据课程导论

1. 数据与数据分析

数据分析离不开数据。百科对数据（data）的定义：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，比如声音、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。

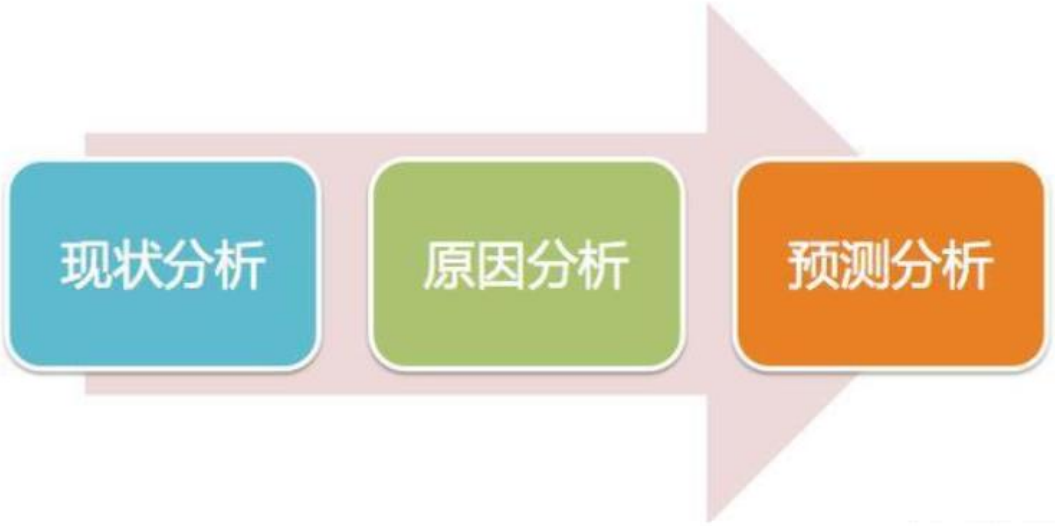
数据分析是指用适当的统计分析方法对收集来的数据进行分析，将它们加以汇总和理解并消化，以求最大化地开发数据的功能，发挥数据的作用。

商业领域中，数据分析能够给帮助企业进行判断和决策，以便采取相应的策略与行动。例如，企业高层希望通过市场分析和研究，把握当前产品的市场动向，从而指定合理的产品研发和销售计划，这就必须依赖数据分析才能完成。生活中最著名的例子便是天气专家通过对气象数据进行分析，并且制作出天气预报，根据预报，我们会做出相应的策略，是带伞还是加件毛衣。

2． 数据分析作用

数据分析的目的是把隐藏在数据背后的信息集中和提炼出来，总结出所研究对象的内在规律，帮助管理者进行有效的判断和决策。

数据分析在企业日常经营分析中主要有三大作用：



2.1． 现状分析（分析当下的数据）

简单来说就是告诉你当前的状况，具体体现在:

第一，告诉你企业现阶段的整体运营情况，通过各个指标的完成情况来衡量企业的运营状态，以说明企业整天运营是好了还是坏了，好的程度如何，坏的程度又到哪里。

第二，告诉你企业各项业务的构成，让你了解企业各项业务的发展以及变动情况，对企业运营状况有更深入的了解。

2.2. 原因分析（分析过去的数据）

简单来说就是告诉你某一现状为什么发生。

经过现状分析，我们对企业的运营情况有了基本了解，但不知道运营情况具体好在哪里，差在哪里，是什么原因引起的。这时就需要开展原因分析，以进一步确定业务变动的具体原因。

例如 2020 年 2 月运营收入下降 50%，是什么原因导致的呢，是各项业务收入都出现下降，还是个别业务收入下降引起的，是各个地区业务收入都出现下降，还是个别地区业务收入下降引起的。这就需要我们开展原因分析，进一步确定收入下降的具体原因，对运营策略做出调整与优化。

2.3. 预测分析（结合数据预测未来）

简单来说就是告诉你将来会发生什么。

在了解企业运营现状后，有时还需要对企业未来发展趋势做出预测，为制订企业运营目标及策略提供有效的参考与决策依据，以保证企业的可持续健康发展。预测分析一般通过专题分析来完成，通常在制订企业季度、年度等计划时进行，其开展的频率没有现状分析及原因分析高。

3. 数据分析基本步骤

张文霖在《数据分析六步曲》说，典型的数据分析应该包含以下几个步骤：

3.1. 明确分析目的和思路

首先明白本次的目的，梳理分析思路，并搭建整体分析框架，把分析目的分解，化为若干的点，清晰明了，即分析的目的，用户什么样的，如何具体开展数据分析，需要从哪几个角度进行分析，采用哪些分析指标（各类分析指标需合理搭配使用）。

同时，确保分析框架的体系化和逻辑性，简单来说就是先分析什么，后分析什么，使得各个分析点之间具有逻辑联系。避免不知从哪方面入手以及分析的内容和指标被质疑是否合理、完整。所以体系化就是为了让你的分析框架具有说服力。

要想使分析框架体系化，就需要一些营销、管理等理论为指导，结合着实际的业务情况进行构建，这样才能保证分析维度的完整性，分析结果的有效性以及正确性。比如以用户行为理论为指导，搭建的互联网网站分析指标框架如下：

把跟数据分析相关的营销、管理等理论统称为数据分析方法论。比如用户行为理论、PEST 分析法、5W2H 分析法等等。

3.2. 数据收集

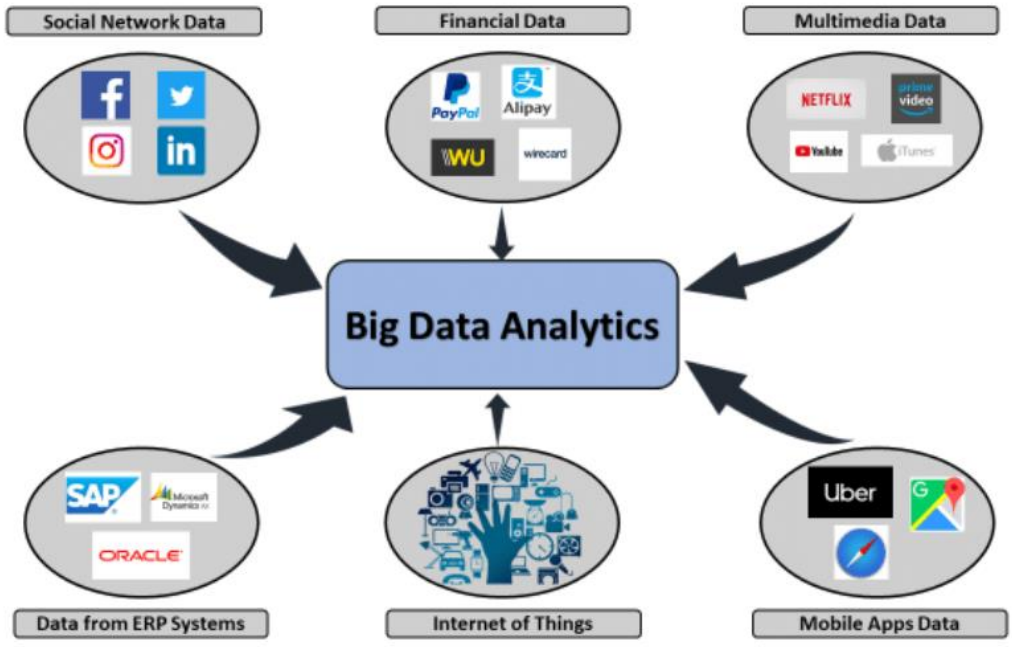
一般数据来源主要有以下几种方式：

数据库：每个公司都有自己的业务数据库，存放从公司成立以来产生的相关业务数据。这个业务数据库就是一个庞大的数据资源，需要有效地利用起来。

公开出版物：可以用于收集数据的公开出版物包括《中国统计年鉴》《中国社会统计年鉴》《中国人口统计年鉴》《世界经济年鉴》《世界发展报告》等统计年鉴或报告。

互联网：随着互联网的发展，网络上发布的数据越来越多，特别是搜索引擎可以帮助我们快速找到所需要的数据，例如国家及地方统计局网站、行业组织网站、政府机构网站、传播媒体网站、大型综合门户网站等上面都可能有我们需要的数据。

市场调查：进行数据分析时，需要了解用户的想法与需求，但是通过以上三种方式获得此类数据会比较困难，因此可以尝试使用市场调查的方法收集用户的想法和需求数据。

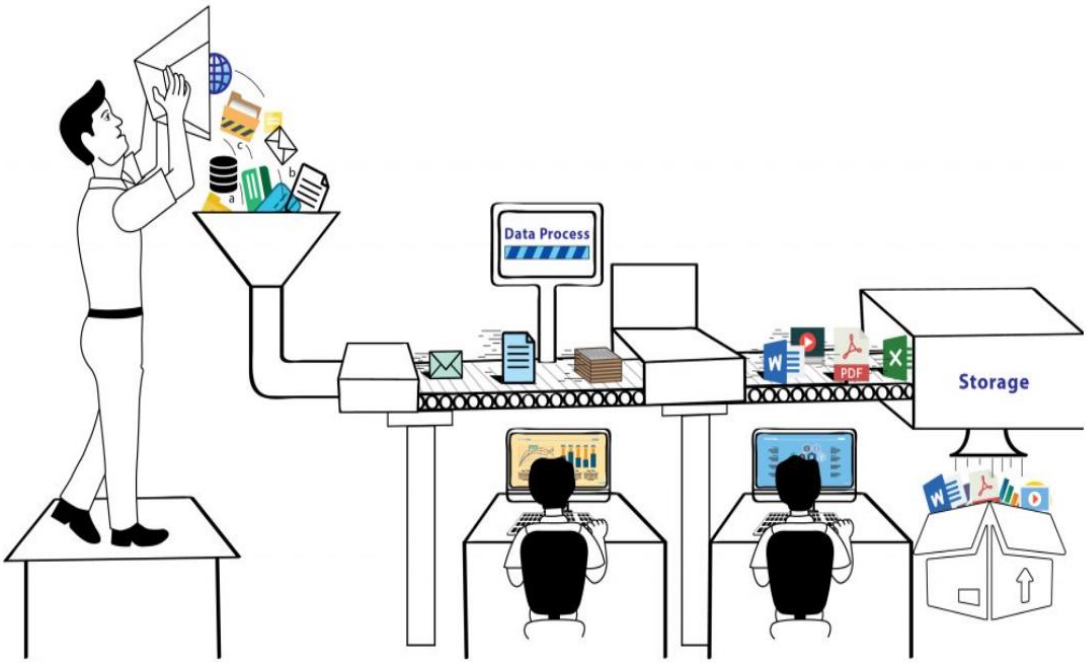


3.3. 数据处理

数据处理是指对收集到的数据进行加工整理，形成适合数据分析的样式，它是数据分析前必不可少的阶段。数据处理的基本目的是从大量的、杂乱无章、难以理解的数据中，抽取并推导出对解决问题有价值、有意义的数据。

数据处理主要包括**数据清洗、数据转化、数据提取、数据计算**等处理方法。一般拿到手的数据都需要进行一定的处理才能用于后续的数据分析工作，即使再“干净”’的原始数据也需要先进行一定的处理才能使用。

数据处理是数据分析的基础。通过数据处理，将收集到的原始数据转换为可以分析的形式，并且保证数据的一致性和有效性。



3.4. 数据分析

数据分析是指用适当的分析方法及工具，对处理过的数据进行分析，提取有价值的信息，形成有效结论的过程。由于数据分析多是通过软件来完成的，这就要求数据分析师不仅要掌握各种数据分析方法，还要熟悉数据分析软件的操作。

数据挖掘其实是一种高级的数据分析方法，就是从大量的数据中挖掘出有用的信息，它是根据用户的特定要求，从浩如烟海的数据中找出所需的信息，以满足用户的特定需求。数据挖掘技术是人们长期对数据库技术进行研究和开发的结果。一般来说，数据挖掘侧重解决四类数据分析问题:分类、聚类、关联和预测，重点在寻找模式和规律。数据分析与数据挖掘的本质是一样的，都是从数据里面发现关于业务的知识。

3.5. 数据展现

一般情况下，数据是通过表格和图形的方式来呈现的，我们常说用图表说话就是这个意思。常用的数据图表包括饼图、柱形图、条形图、折线图、散点图、雷达图等，当然可以对这些图表进一步整理加工，使之变为我们所需要的图形，例如金字塔图、矩阵图、漏斗图等。

大多数情况下，人们更愿意接受图形这种数据展现方式，因为它能更加有效、直观地传递出分析所要表达的观点。记位，一般情况不，能用图说明问题的就不用表格，能用表格说明问题的就不要用文字。

3.6. 报告撰写

数据分析报告其实是对整个数据分析过程的一个总结与呈现。通过报告，把数据分析的起因、过程、结果及建议完整地呈现出来，供决策者参考。

一份好的数据分析报告，首先需要有一个好的分析框架，并且图文并茂，层次明晰，能够让阅读者一目了然。结构清晰、主次分明可以使阅读者正确理解报告内容;图文并茂，可以令数据更加生动活泼 ,提供视觉冲击力，有助于阅读者更形象、直观地看清楚问题和结论，从而产生思考。

另外，数据分析报告需要有明确的结论，没有明确结论的分析称不上分析，同时也失去了报告的意义，因为我们最初就是为寻找或者求证一个结论才进行分析的，所以千万不要舍本求末。

最后，好的分析报告一定要有建议或解决方案。作为决策者，需要的不仅仅是找出问题，更重要的是建议或解决方案，以便他们做决策时作参考。所以，数据分析师不仅需要掌握数据分析方法，而且还要了解和熟悉业务，这样才能根据发现的业务问题，提出具有可行性的建议或解决方案。

2019年12月份BB霜分析报告

时间：2019年12月　报告类型：电商报告　行业：个护化妆　☆ 收藏

一、BB霜行业概况

随着互联网普及率的不断提高，电商行业市场规模迅速攀升。各行各业持续发力，通过创新营销模式来增加消费欲望。

从行业整体研究数据来看，BB霜行业12月份的整体销量为8753.69万件，销售额为￥152.58亿元，行业产品的成交均价为￥174.31元

BB霜行业12月销售情况数据	
本月销售额	152.58亿元
本月销售量	8753.69万件
成交均价	174.31元

4. 大数据时代

4.1. 概述

最早提出“大数据”时代到来的是全球知名咨询公司麦肯锡，麦肯锡称：“数据，已经渗透到当今每一个行业和业务职能领域，成为重要的生产因素。人们对于海量数据的挖掘和运用，预示着新一波生产率增长和消费者盈余浪潮的到来。”

进入 2012 年，大数据（big data）一词越来越多地被提及，人们用它来描述和定义信息爆炸时代产生的海量数据，并命名与之相关的技术发展与创新。

CCTV 纪录片《大数据时代》，是国内首部大数据产业题材纪录片，节目细致而生动地讲述了大数据技术在政府治理、民生服务、数据安全、工业转型、未来生活等方面给我们带来的改变和影响。

4.2. 海量数据的挑战

公开数据显示，互联网搜索巨头百度 2013 年拥有数据量接近 EB 级别。阿里、腾讯都声明自己存储的数据总量都达到了百 PB 以上。此外，电信、医疗、金融、公共安全、交通、气象等各个方面保存的数据量也都达到数十或者上百 PB 级别。全球数据量以每两年翻倍的速度增长,在 2010 年已经正式进入 ZB 时代，2020 年全球数据总量达到 44ZB。

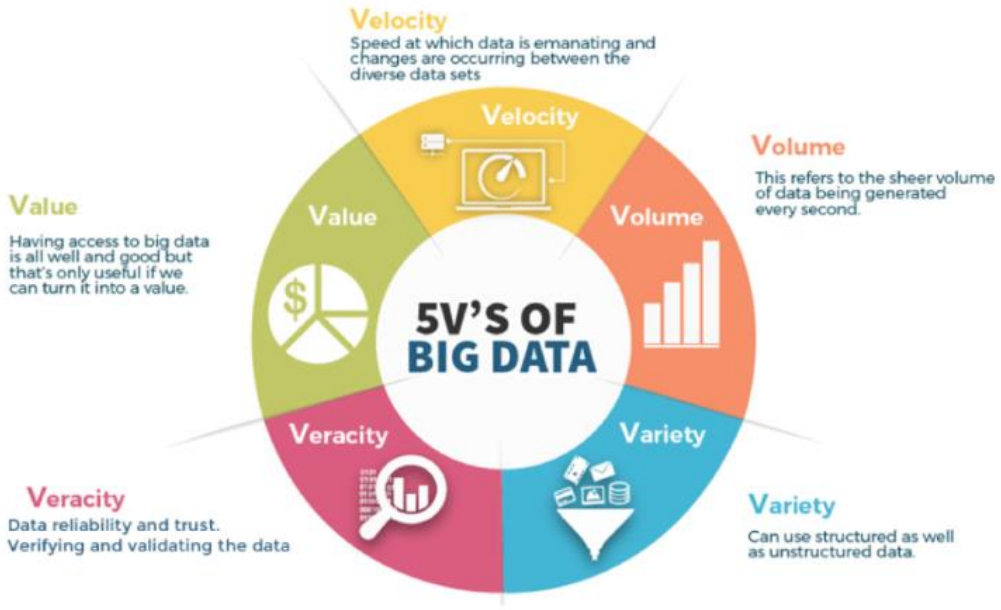
```
1KB (Kilobyte 千)=1024B,
1MB (Megabyte 兆)=1024KB,
1GB (Gigabyte 吉)=1024MB,
1TB (Trillionbyte 太)=1024GB,
1PB（Petabyte 拍）=1024TB,
1EB（Exabyte 艾）=1024PB,
1ZB (Zettabyte 泽)= 1024 EB,
1YB (Yottabyte 尧)= 1024 ZB,
1BB (Brontobyte 布)= 1024 YB.
```

究竟怎么去存储庞大的数据，是企业面临的首要问题。传统的数据存储模式存储容量是有大小限制或者空间限制的，怎么去设计出一个可以支撑大量数据的存储方案是开展数据分析的首要前提。

解决了海量数据的存储问题，接下来面临的海量数据的计算问题也是比较让人头疼，因为企业不仅追求可以计算，还会追求计算的速度、效率。

以目前互联网行业产生的数据量级别，要处理这些数据，就需要一个更好、更便捷的分析计算方式了。传统的数据处理方式显然力不从心，而且效率也会非常低下。这正是传统数据分析领域面临的另一个挑战，如何去分析、计算海量数据。

4.3. 大数据的特点（5V 特征）



Volume：数据量大，包括采集、存储和计算的量都非常大；

Variety：种类和来源多样化。包括结构化、半结构化和非结构化数据；

Value：数据价值密度相对较低，或者说是浪里淘沙却又弥足珍贵；

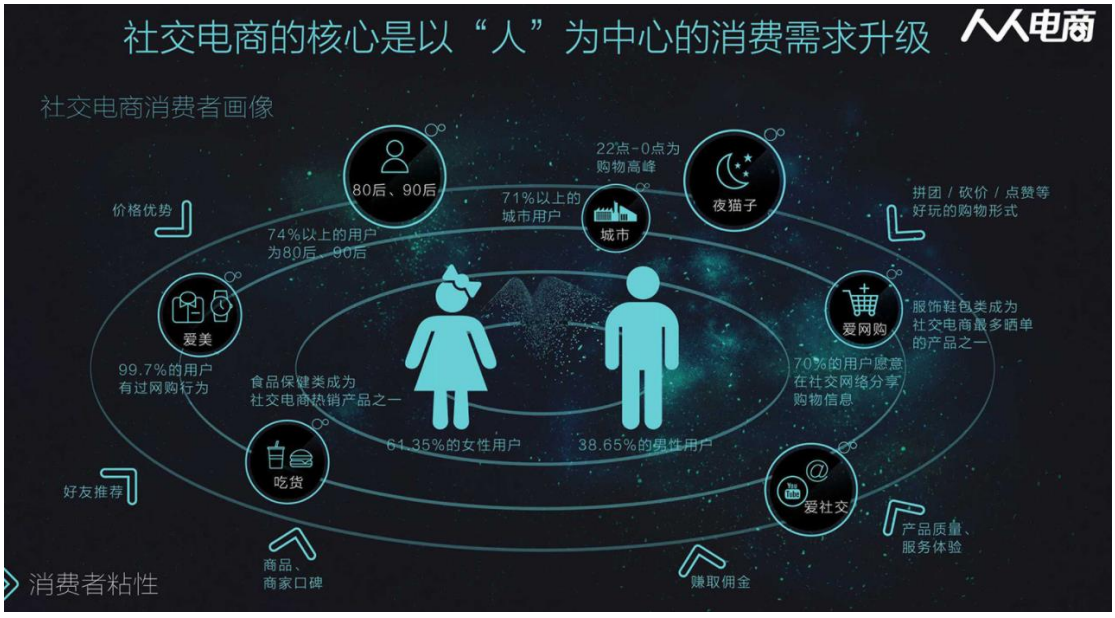
Velocity：数据增长速度快，处理速度也快，时效性要求高；

Veracity：数据的准确性和可信赖度，即数据的质量。

4.4. 大数据的应用场景

电商方面：精准广告位，通过对用户的浏览行为，点击行为等进行大数据采集，分析，挖掘用户的二层三层喜欢，扩大产出。

传媒方面：猜你喜欢，通过对受众人群机型大数据分析，结合对应算法，对受众喜欢的进行交互推荐。



金融方面：理财投资，通过对个人的信用评估，风险承担能力评估，集合众多理财产品、推荐响应的投资理财产品。

交通方面：目前，交通的大数据应用主要在两个方面:一方面通过对车流量等海量数据的收集，估算，预测该路段一定时间内的车流量情况，给用户提供便利，合理进行道路规划；另一方面可以利用大数据来实现即时信号灯调度，提高已有线路通行能力。

电信方面：智慧营业厅，通过对用户当前的行为习惯、偏好，节假日的相应数据变化，调节自身业务结构，做到按需分配。

安防方面：人脸识别，通过人脸识别，匹配，存储用户数据，结合人工智能，分析及甄别用户行为，预防犯罪行为发生。

医疗方面：智慧医疗，通过对海量病例大数据的存储，匹配、检索、结合用户的饮食、行为等习惯，搭建智慧医疗体系。

5. 分布式技术

5.1. 什么是分布式

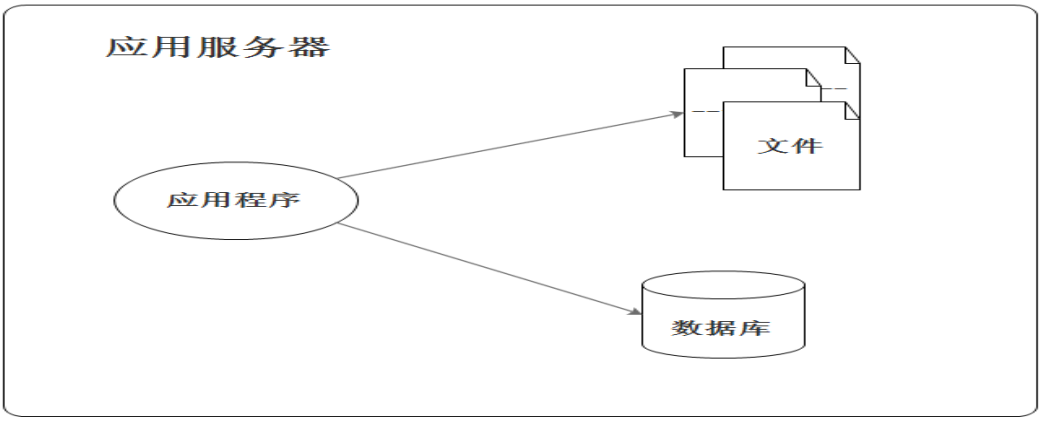
分布式系统是指：一个硬件或软件，其组件会分布在不同的计算机上，彼此之间仅仅通过网络消息传递进行通信和协调的系统。

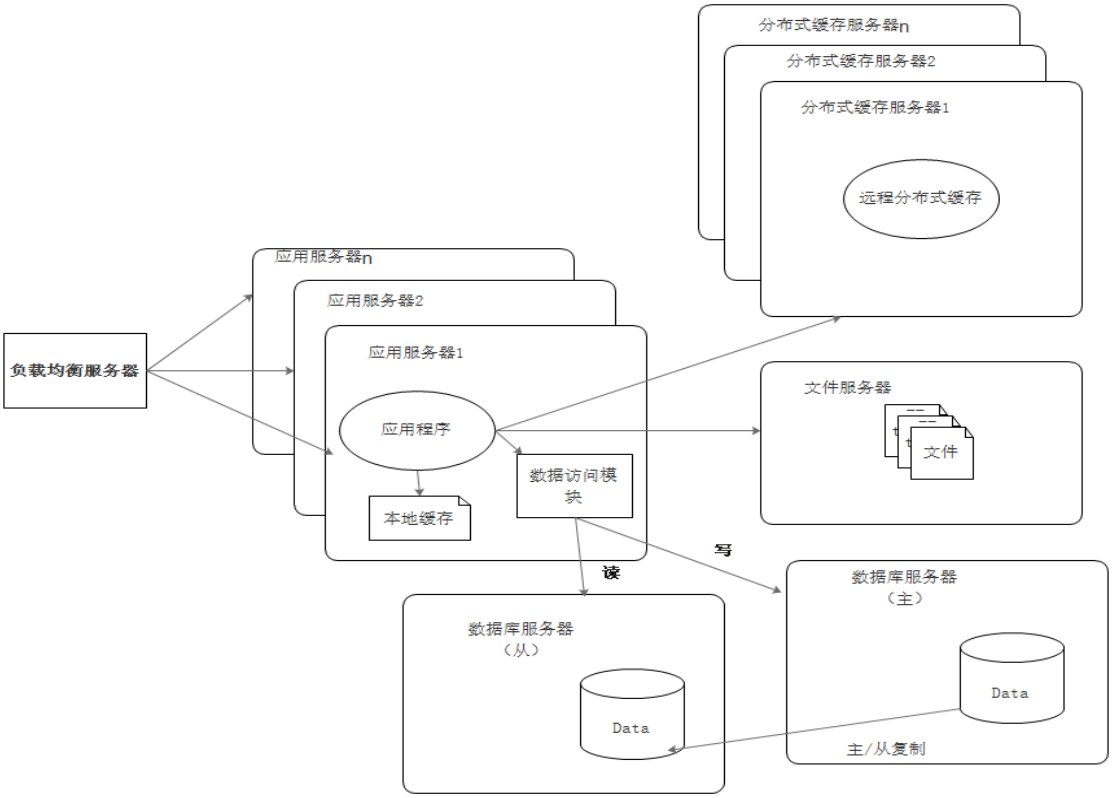
简单来说就是一群独立计算机集合起来共同对外提供服务，但是对于系统的用户来说，就像是一台计算机在提供服务一样。

分布式意味着可以采用更多的普通计算机（相对于昂贵的大型机）组成分布式集群对外提供服务。计算机越多，CPU、内存、存储资源等也就越多，能够处理的并发访问量也就越大。

下面以网站架构变迁来举例说明。

初代的 web 服务网站架构往往比较简单，应用程序、数据库、文件等所有的资源都在一台服务器上。





图：现在互联网网站常用的架构

从分布式系统的概念中我们知道，各个主机之间通信和协调主要通过网络进行，所以，分布式系统中的计算机在空间上几乎没有任何限制，这些计算机可能被放在不同的机柜上，也可能被部署在不同的机房中，还可能在不同的城市中，对于大型的网站甚至可能分布在不同的国家和地区。

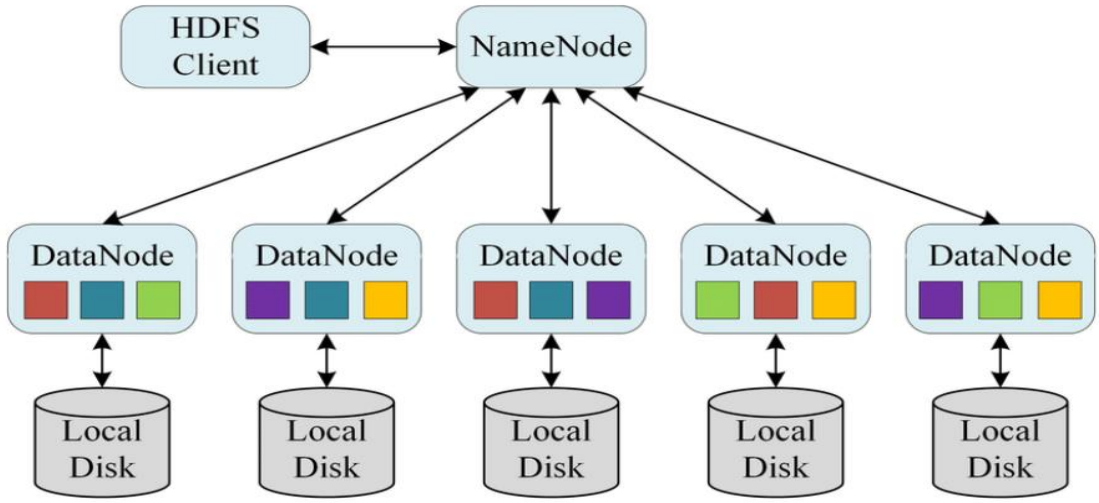
5.2. 常用分布式方案

分布式应用和服务

将应用和服务进行分层和分割，然后将应用和服务模块进行分布式部署。这样做不仅可以提高并发访问能力、减少数据库连接和资源消耗，还能使不同应用复用共同的服务，使业务易于扩展。比如：分布式服务框架 Dubbo。

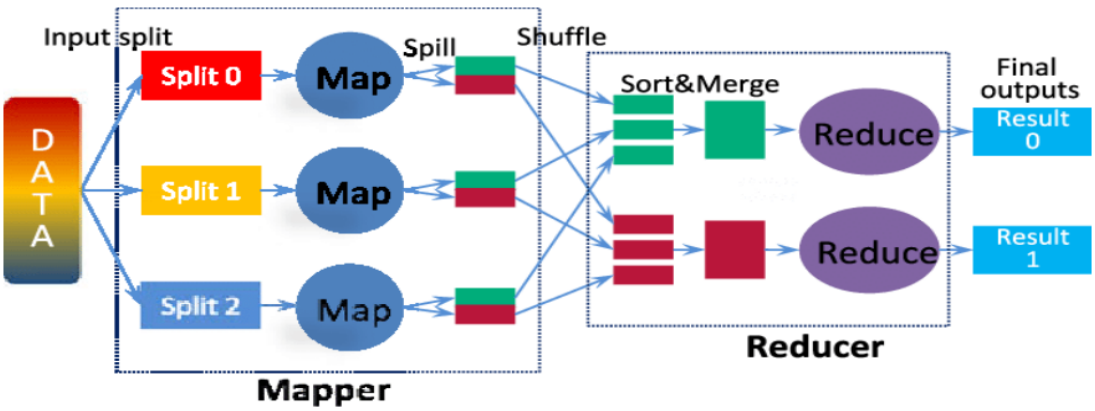
分布式数据存储

大型网站常常需要处理海量数据，单台计算机往往无法提供足够的内存空间，可以对这些数据进行分布式存储。比如 Apache Hadoop HDFS。



分布式计算

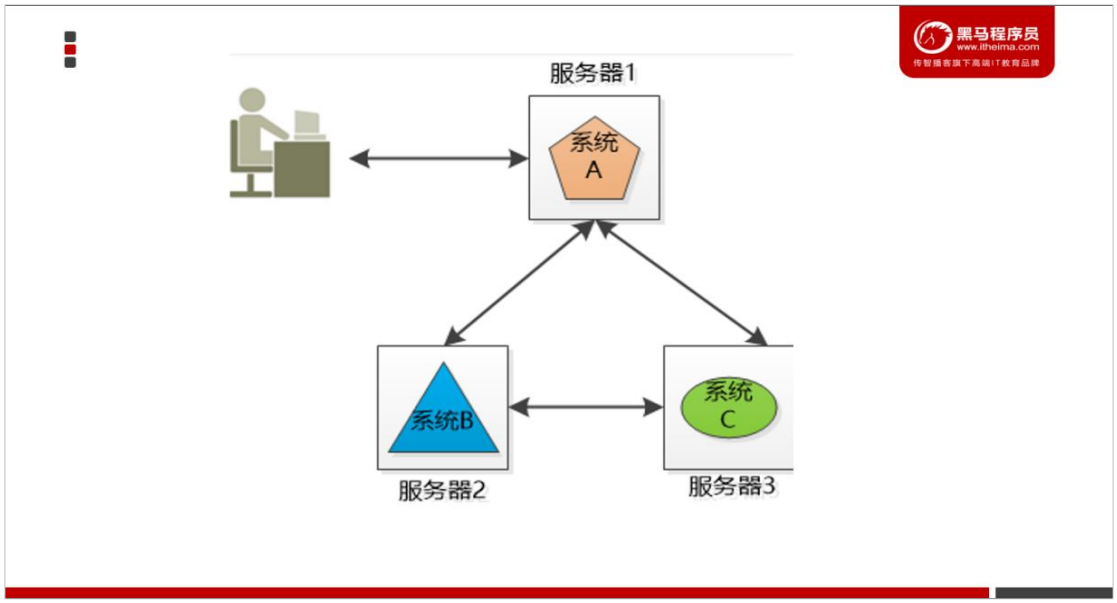
随着计算技术的发展，有些应用需要非常巨大的计算能力才能完成，如果采用集中式计算，需要耗费相当长的时间来完成。分布式计算将该应用分解成许多小的部分，分配给多台计算机进行处理。这样可以节约整体计算时间，大大提高计算效率。比如 Apache Hadoop MapReduce。



5.3. 分布式、集群

分布式（distributed）是指在多台不同的服务器中部署不同的服务模块，通过远程调用协同工作，对外提供服务。

集群（cluster）是指在多台不同的服务器中部署相同应用或服务模块，构成一个集群，通过负载均衡设备对外提供服务。



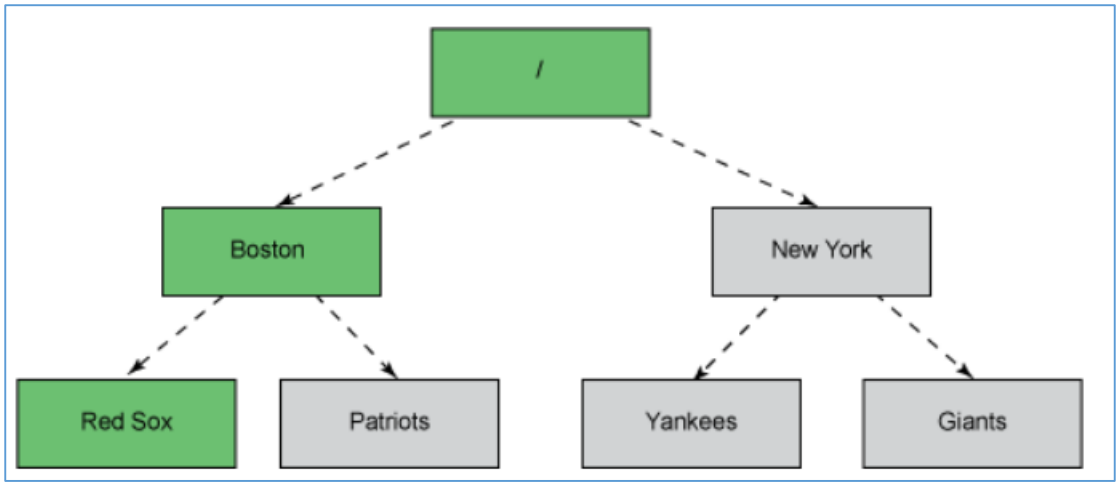
三、 Apache ZooKeeper

1. Zookeeper 基本知识

1.1. ZooKeeper 概述

Zookeeper 是一个分布式协调服务的开源框架。主要用来解决分布式集群中应用系统的一致性问题。

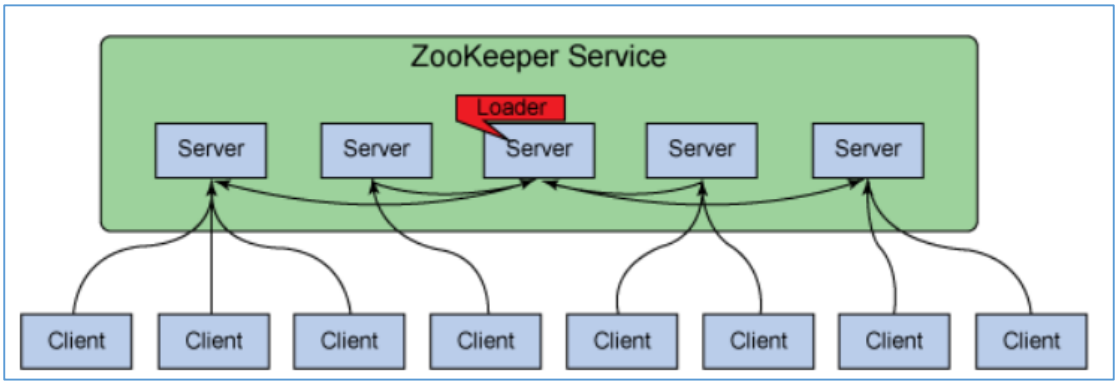
ZooKeeper 本质上是一个分布式的小文件存储系统。提供基于类似于文件系统的目录树方式的数据存储，并且可以对树中的节点进行有效管理。从而用来维护和监控你存储的数据的状态变化。通过监控这些数据状态的变化，从而可以达到基于数据的集群管理。



1.2. ZooKeeper 特性

1. 全局数据一致：集群中每个服务器保存一份相同的数据副本，client 无论连接到哪个服务器，展示的数据都是一致的，这是最重要的特征；
2. 可靠性：如果消息被其中一台服务器接受，那么将被所有的服务器接受。
3. 顺序性：包括全局有序和偏序两种：全局有序是指如果在一台服务器上消息 a 在消息 b 前发布，则在所有 Server 上消息 a 都将在消息 b 前被发布；偏序是指如果一个消息 b 在消息 a 后被同一个发送者发布，a 必将排在 b 前面。
4. 数据更新原子性：一次数据更新要么成功（半数以上节点成功），要么失败，不存在中间状态；
5. 实时性：Zookeeper 保证客户端将在一个时间间隔范围内获得服务器的更新信息，或者服务器失效的信息。

1.3. ZooKeeper 集群角色



Leader:

Zookeeper 集群工作的核心

事务请求（写操作）的唯一调度和处理者，保证集群事务处理的顺序性；

集群内部各个服务器的调度者。

对于 create，setData，delete 等有写操作的请求，则需要统一转发给 leader 处理，leader 需要决定编号、执行操作，这个过程称为一个事务。

Follower:

处理客户端非事务（读操作）请求，转发事务请求给 Leader；

参与集群 Leader 选举投票。

此外，针对访问量比较大的 zookeeper 集群，还可新增观察者角色。

Observer:

观察者角色，观察 Zookeeper 集群的最新状态变化并将这些状态同步过来，其对于非事务请求可以进行独立处理，对于事务请求，则会转发给 Leader 服务器进行处理。

不会参与任何形式的投票只提供非事务服务，通常用于在不影响集群事务处理能力的前提下提升集群的非事务处理能力。

1.4. ZooKeeper 集群搭建

Zookeeper 集群搭建指的是 ZooKeeper 分布式模式安装。通常由 2n+1 台 servers 组成。这是因为为了保证 Leader 选举（基于 Paxos 算法的实现）能过得到多数的支持，所以 ZooKeeper 集群的数量一般为奇数。

Zookeeper 运行需要 java 环境，所以需要提前安装 jdk。对于安装 leader+follower 模式的集群，大致过程如下：

- 配置主机名称到 IP 地址映射配置
- 修改 ZooKeeper 配置文件
- 远程复制分发安装文件
- 设置 myid
- 启动 ZooKeeper 集群

如果要想使用 Observer 模式，可在对应节点的配置文件添加如下配置：

```
peerType=observer
```

其次，必须在配置文件指定哪些节点被指定为 Observer，如：

```
server.1:node1:2181:3181:observer
```

详细步骤请参考附件安装资料。

2. ZooKeeper shell

2.1. 客户端连接

运行 `zkCli.sh -server ip` 进入命令行工具。

输入 help，输出 zk shell 提示：

```
ZooKeeper -server host:port cmd args
        stat path [watch]
        set path data [version]
        ls path [watch]
        delquota [-n|-b] path
        ls2 path [watch]
        setAcl path acl
        setquota -n|-b val path
        history
        redo cmdno
        printwatches on|off
        delete path [version]
        sync path
        listquota path
        rmr path
        get path [watch]
        create [-s] [-e] path data acl
        addauth scheme auth
        quit
        getAcl path
        close
        connect host:port
```

2.2. shell 基本操作

创建节点

```
create [-s] [-e] path data acl
```

其中，-s 或-e 分别指定节点特性，顺序或临时节点，若不指定，则表示持久节点；acl 用来进行权限控制。

创建顺序节点：

```
[zk: node-22(CONNECTED) 4] create -s /test 123
Created /test0000000003
```

创建临时节点：

```
[zk: node-22(CONNECTED) 5] create -e /test-temp 123temp
Created /test-temp
```

创建永久节点：

```
[zk: node-22(CONNECTED) 1] create /test-p 123p
Created /test-p
```

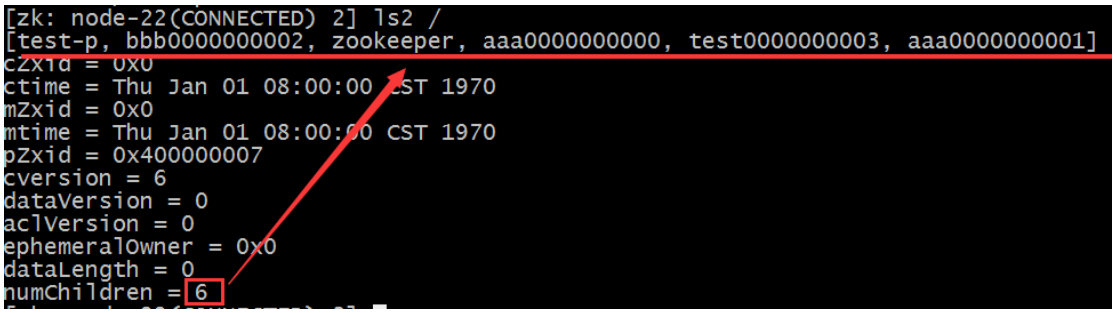
读取节点

与读取相关的命令有 ls 命令和 get 命令，ls 命令可以列出 Zookeeper 指定节点下的所有子节点，只能查看指定节点下的第一级的所有子节点；get 命令可以获取 Zookeeper 指定节点的数据内容和属性信息。

ls path [watch]

get path [watch]

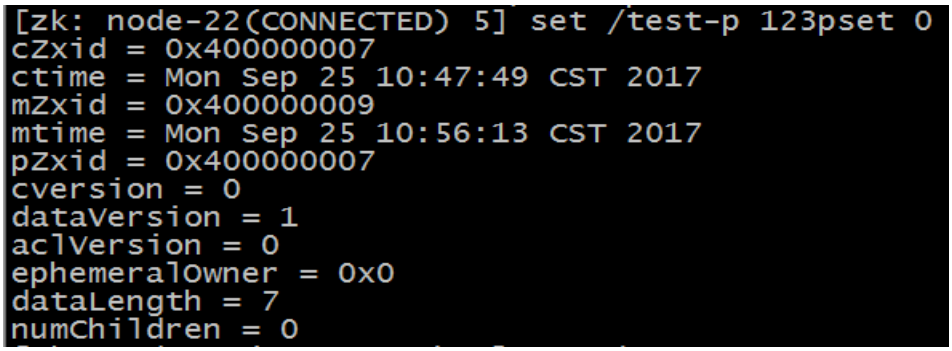
ls2 path [watch]



更新节点

set path data [version]

data 就是要更新的新内容，version 表示数据版本。



现在 dataVersion 已经变为 1 了，表示进行了更新。

删除节点

delete path [version]

若删除节点存在子节点，那么无法删除该节点，必须先删除子节点，再删除父节点。

Rmr path

可以递归删除节点。

quota

setquota -n|-b val path 对节点增加限制。

n:表示子节点的最大个数

b:表示数据值的最大长度

val:子节点最大个数或数据值的最大长度

path:节点路径

```
[zk: node-22(CONNECTED) 13] setquota -n 2 /quota
Comment: the parts are option -n val 2 path /quota
```

listquota path 列出指定节点的 quota

```
[zk: node-22(CONNECTED) 14] listquota /quota
absolute path is /zookeeper/quota/quota/zookeeper_limits
Output quota for /quota count=2,bytes=-1
Output stat for /quota count=1,bytes=1
```

子节点个数为 2,数据长度-1 表示没限制

delquota [-n|-b] path 删除 quota

其他命令

history ： 列出命令历史

```
[zk: node-22(CONNECTED) 16] history
6 - get /test-p
7 - delete /test-p
8 - ls
9 - ls /
10 - stat /
11 - ls /
12 - create /quota 1
13 - setquota -n 2 /quota
14 - listquota /quota
15 - delquota -n /quota
16 - history
```

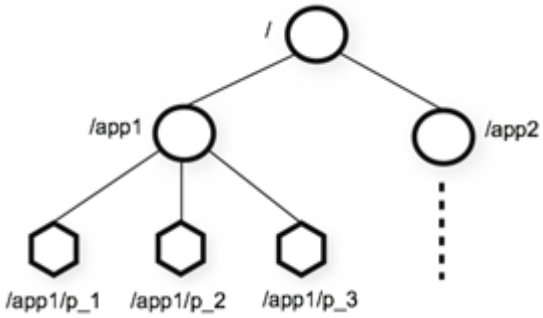
redo：该命令可以重新执行指定命令编号的历史命令,命令编号可以通过 history 查看

3. ZooKeeper 数据模型

ZooKeeper 的数据模型，在结构上和标准文件系统的非常相似，拥有一个层次的命名空间，都是采用树形层次结构，ZooKeeper 树中的每个节点被称为—Znode。和文件系统的目录树一样，ZooKeeper 树中的每个节点可以拥有子节点。但也有不同之处：

1. Znode 兼具文件和目录两种特点。既像文件一样维护着数据、元信息、ACL、时间戳等数据结构，又像目录一样可以作为路径标识的一部分，并可以具有子 Znode。用户对 Znode 具有增、删、改、查等操作（权限允许的情况下）。
2. Znode 具有原子性操作，读操作将获取与节点相关的所有数据，写操作也将替换掉节点的所有数据。另外，每一个节点都拥有自己的 ACL(访问控制列表)，这个列表规定了用户的权限，即限定了特定用户对目标节点可以执行的操作。
3. Znode 存储数据大小有限制。ZooKeeper 虽然可以关联一些数据，但并没有被设计为常规的数据库或者大数据存储，相反的是，它用来管理调度数据，比如分布式应用中的配置文件信息、状态信息、汇集位置等等。这些数据的共同特性就是它们都是很小的数据，通常以 KB 为大小单位。ZooKeeper 的服务器和客户端都被设计为严格检查并限制每个 Znode 的数据大小至多 1M，当时常规使用中应该远小于此值。
4. Znode 通过路径引用，如同 Unix 中的文件路径。路径必须是绝对的，因此他们必须由斜杠字符来开头。除此以外，他们必须是唯一的，也就是说每一个路径只有一个表示，因此这些路径不能改变。在 ZooKeeper 中，路径由 Unicode 字符串组成，并且有一些限制。字符串"/zookeeper"用以保存管理信息，比如关键配额信息。

3.1. 数据结构图



图中的每个节点称为一个 Znode。 每个 Znode 由 3 部分组成:

① stat：此为状态信息，描述该 Znode 的版本，权限等信息

② data：与该 Znode 关联的数据

③ children：该 Znode 下的子节点

3.2. 节点类型

Znode 有两种，分别为临时节点和永久节点。

节点的类型在创建时即被确定，并且不能改变。

临时节点：该节点的生命周期依赖于创建它们的会话。一旦会话结束，临时节点将被自动删除，当然可以也可以手动删除。临时节点不允许拥有子节点。

永久节点：该节点的生命周期不依赖于会话，并且只有在客户端显示执行删除操作的时候，他们才能被删除。

Znode 还有一个序列化的特性，如果创建的时候指定的话，该 Znode 的名字后面会自动追加一个不断增加的序列号。序列号对于此节点的父节点来说是唯一的，这样便会记录每个子节点创建的先后顺序。它的格式为“%10d”(10 位数字，没有数值的数位用 0 补充，例如“0000000001”)。

```
[zk: localhost:2181(CONNECTED) 4] create -s /aaa helloallen
Created /aaa0000000000
[zk: localhost:2181(CONNECTED) 5] create -s /aaa helloallen
Created /aaa0000000001
[zk: localhost:2181(CONNECTED) 6] get /aaa
Node does not exist: /aaa
[zk: localhost:2181(CONNECTED) 7] ls /
[zookeeper, aaa0000000000, aaa0000000001]
[zk: localhost:2181(CONNECTED) 8] create -s /bbb helloallen
Created /bbb0000000002
[zk: localhost:2181(CONNECTED) 9] ls /
[bbb0000000002, zookeeper, aaa0000000000, aaa0000000001]
[zk: localhost:2181(CONNECTED) 10]
```

这样便会存在四种类型的 Znode 节点，分别对应：

PERSISTENT：永久节点

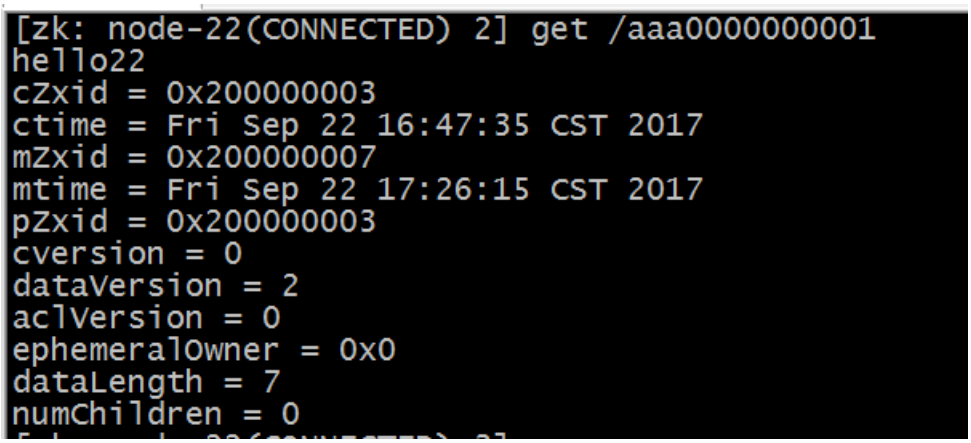
EPHEMERAL：临时节点

PERSISTENT_SEQUENTIAL：永久节点、序列化

EPHEMERAL_SEQUENTIAL：临时节点、序列化

3.3. 节点属性

每个 znode 都包含了一系列的属性，通过命令 get，可以获得节点的属性。



dataVersion：数据版本号，每次对节点进行 set 操作，dataVersion 的值都会增加 1（即使设置的是相同的数据），可有效避免了数据更新时出现的先后顺序问题。

cversion ：子节点的版本号。当 znode 的子节点有变化时，cversion 的值就会增加 1。

cZxid ：Znode 创建的事务 id。

mZxid ：Znode 被修改的事务 id，即每次对 znode 的修改都会更新 mZxid。

对于 zk 来说，每次的变化都会产生一个唯一的事务 id，zxid（ZooKeeper Transaction Id）。通过 zxid，可以确定更新操作的先后顺序。例如，如果 zxid1 小于 zxid2，说明 zxid1 操作先于 zxid2 发生，zxid 对于整个 zk 都是唯一的，即使操作的是不同的 znode。

ctime：节点创建时的时间戳.

mtime：节点最新一次更新发生时的时间戳.

ephemeralOwner:如果该节点为临时节点，ephemeralOwner 值表示与该节点绑定的 session id. 如果不是，ephemeralOwner 值为 0.

在 client 和 server 通信之前,首先需要建立连接,该连接称为 session。连接建立后,如果发生连接超时、授权失败,或者显式关闭连接,连接便处于 CLOSED 状态，此时 session 结束。

4. ZooKeeper Watcher（监听机制）

ZooKeeper 提供了分布式数据发布/订阅功能，一个典型的发布/订阅模型系统定义了一种一对多的订阅关系，能让多个订阅者同时监听某一个主题对象，当这个主题对象自身状态变化时，会通知所有订阅者，使他们能够做出相应的处理。

ZooKeeper 中，引入了 Watcher 机制来实现这种分布式的通知功能。ZooKeeper 允许客户端向服务端注册一个 Watcher 监听，当服务端的一些事件触发了这个 Watcher，那么就会向指定客户端发送一个事件通知来实现分布式的通知功能。

触发事件种类很多，如：节点创建，节点删除，节点改变，子节点改变等。

总的来说可以概括 Watcher 为以下三个过程：客户端向服务端注册 Watcher、服务端事件发生触发 Watcher、客户端回调 Watcher 得到触发事件情况

Watch 机制特点

一次性触发

事件发生触发监听，一个 watcher event 就会被发送到设置监听的客户端，这种效果是一次性的，后续再次发生同样的事件，不会再次触发。

事件封装

ZooKeeper 使用 WatchedEvent 对象来封装服务端事件并传递。

WatchedEvent 包含了每一个事件的三个基本属性：

通知状态（keeperState），事件类型（EventType）和节点路径（path）

event 异步发送

watcher 的通知事件从服务端发送到客户端是异步的。

先注册再触发

Zookeeper 中的 watch 机制，必须客户端先去服务端注册监听，这样事件发送才会触发监听，通知给客户端。

通知状态和事件类型

同一个事件类型在不同的通知状态中代表的含义有所不同，下表列举了常见的通知状态和事件类型。

KeeperState	EventType	触发条件	说明
	None (-1)	客户端与服务端成功建立连接	
SyncConnected (0)	NodeCreated (1)	Watcher监听的对应数据节点被创建	
	NodeDeleted (2)	Watcher监听的对应数据节点被删除	此时客户端和服务器处于连接状态
	NodeDataChanged (3)	Watcher监听的对应数据节点的数据内容发生变更	
	NodeChildChanged (4)	Wather监听的对应数据节点的子节点列表发生变更	
Disconnected (0)	None (-1)	客户端与ZooKeeper服务器断开连接	此时客户端和服务器处于断开连接状态
Expired (-112)	Node (-1)	会话超时	此时客户端会话失效，通常同时也会受到SessionExpiredException异常
AuthFailed (4)	None (-1)	通常有两种情况，1：使用错误的schema进行权限检查 2：SASL权限检查失败	通常同时也会收到AuthFailedException异常

其中连接状态事件(type=None, path=null)不需要客户端注册，客户端只要有需要直接处理就行了。

Shell 客户端设置 watcher

设置节点数据变动监听：

```
[zk: localhost:2181(CONNECTED) 12] get /aaa0000000001 watch
helloallen
```

通过另一个客户端更改节点数据：

```
[zk: localhost:2181(CONNECTED) 0] set /aaa0000000001 hello22
```

此时设置监听的节点收到通知：

```
[zk: localhost:2181(CONNECTED) 13]
WATCHER::

WatchedEvent state:SyncConnected type:NodeDataChanged path:/aaa0000000001
get /aaa0000000001 watch
hello22
```

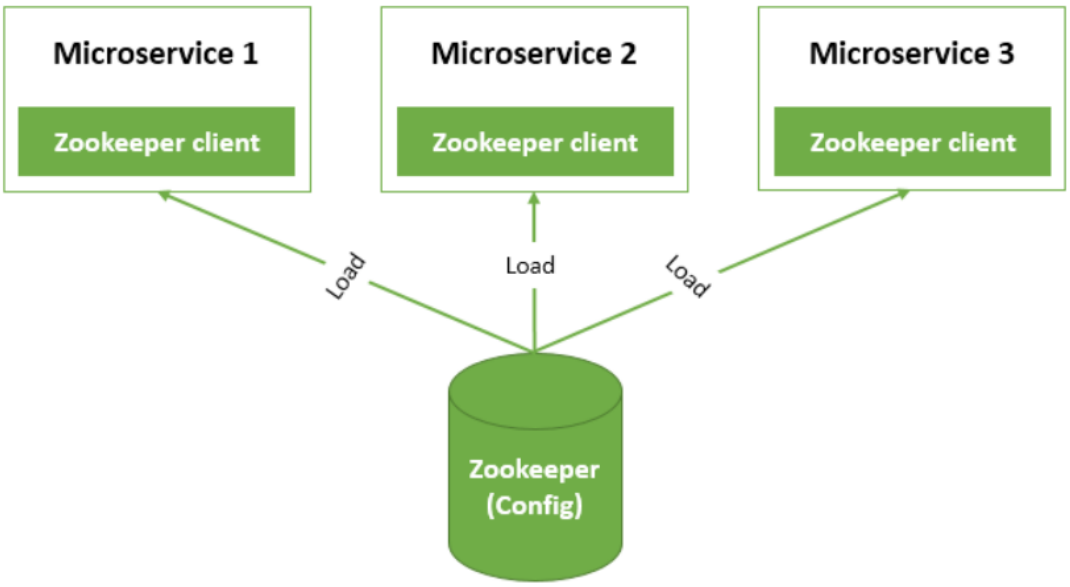
5. Zookeeper 典型应用

5.1. 数据发布/订阅

数据发布/订阅系统即所谓的配置中心，也就是发布者将数据发布到 ZooKeeper 的一个节点上，提供订阅者进行数据订阅，从而实现动态更新数据的目的，实现配置信息的集中式管理和数据的动态更新。

ZooKeeper 采用的是推拉相结合的方式：客户端向服务器注册自己需要关注的节点，一旦该节点的数据发生改变，那么服务端就会向相应的客户端发送 Watcher 事件通知，客户端接收到消息通知后，需要主动到服务端获取最新的数据。

主要用到了：监听机制。

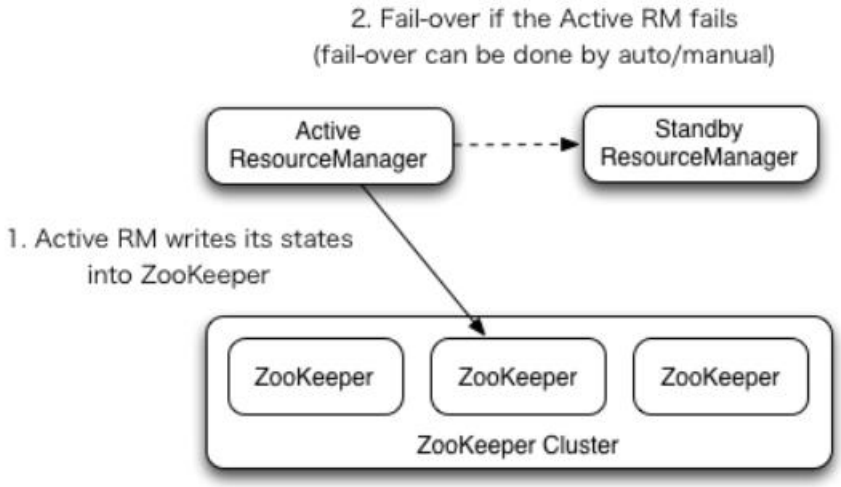


5.2. 提供集群选举

在分布式环境下，不管是主从架构集群，还是主备架构集群，要求在服务的时候有且有一个正常的对外提供服务，我们称之为 master。

当 master 出现故障之后，需要重新选举出的新的 master。保证服务的连续可用性。zookeeper 可以提供这样的功能服务。

主要用到了：znode 唯一性、临时节点短暂性、监听机制。



5.3. 分布式锁

ZooKeeper 通过数据节点表示一个锁，例如/itcast/lock 节点就可以定义一个锁，所有客户端都会调用 create()接口，试图在/itcast 下创建 lock 子节点，但是 ZooKeeper 的强一致性会保证所有客户端最终只有一个客户创建成功。也就可以认为获得了锁，其它线程 Watcher 监听子节点变化（等待释放锁，竞争获取资源）。

此外也可以通过 znode 的序列化特性，给创建 znode 的客户端自动编号，从而实现所谓的顺序锁的功能。