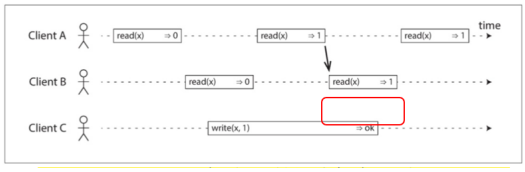
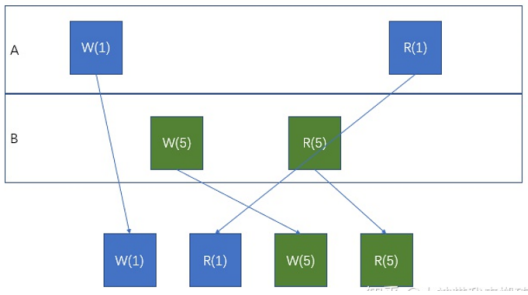
5.

学生：张帅豪 18030100101　　老师：李龙海

强一致性和弱一致性

强一致性	顺序一致性
在客户端看来，分布式存储系统的外在表现和单副本存储系统的外在表现完全一致。	任意客户端看到的所有针对分布式系统的操作(读、写等原子操作)按全局一致的顺序排列（线性化），同一个客户端发出的多个操作的顺序与该全局一致排序并不矛盾。
任意客户端看到的所有针对分布式系统的操作(读、写等原子操作)按全局一致的顺序排列（线性化），并且该排序满足多个操作在时间维度上的实际发生先后顺序。	
 一个读操作读取到了一个更新后的变量值，发生在该读操作之后的读操作只能读出相同的值或者更新版本的值，不能读出旧版本的值。	

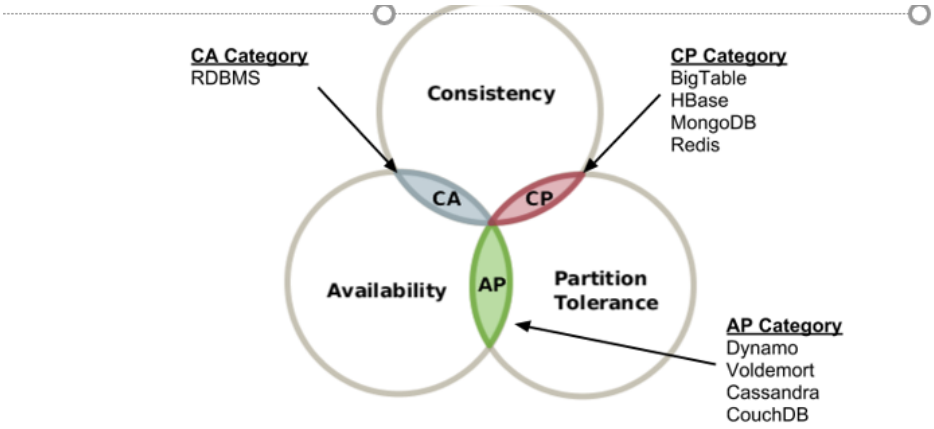
因果一致性	最终一致性
不同客户端看到的所有针对分布式系统的操作(读、写等原子操作)排序不一定一致，但该排序不违背操作发生的因果关系。	在分布式系统停止更新时，最终所有读操作都可以获得最新版本的数据。

几乎所有的分布式存储系统都支持最终一致性。
分布式系统的一致性保证越强，客户端应用程序越容易编写；一致性保证越弱，客户端程序逻辑就越复杂。
一致性保证越强，分布式系统的实现代价就越高，性能就越低。因此可以通过牺牲部分一致性获得更好的性能和容错性。

cap定理

- Consistensy(一致性)：不同节点上数据的强一致性
- Partition Tolerance(割断容忍性)：允许部分节点与其它节点断裂
- Availability(可用性)：发出的请求在规定时间段内总能返回结果（请求响应延时短，可用性高；否则可用性低）

> 在设计分布式系统时，三者只能取其二，不能三者兼得



Base定理

> 是对CAP中一致性和可用性权衡的结果，来源于对大规模互联网分布式系统实践的总结，核心思想是即使无法做到强一致性，但每个应用都可以根据自身业务特点降低部分一致性获得可用性。

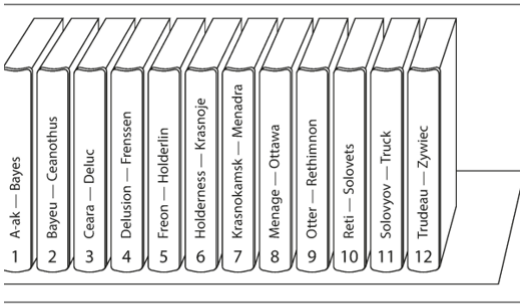
数据分区

> 目标：将数据和查询负载均匀分布在各个节点上，避免出现偏斜(skew)和热点(hot spot)问题。

根据主键范围

> - 数据集合中每个元素（块、对象、记录）都可以找到一个主键，根据主键的连续范围进行分区。
> - 各个主键范围分区一般都是非均匀分布的。
> - 不同的主键范围分区分配给不同的物理存储节点。在特定分区进行分裂或合并时会产生数据移动。

> **优点**：按主键进行连续查询很方便。
>
> **缺点**：在主键范围非均匀分布时必须建立全局索引以记录数据分区和存储节点的对应关系。一般要专门指定一个节点维护全局索引，该节点是中心节点。

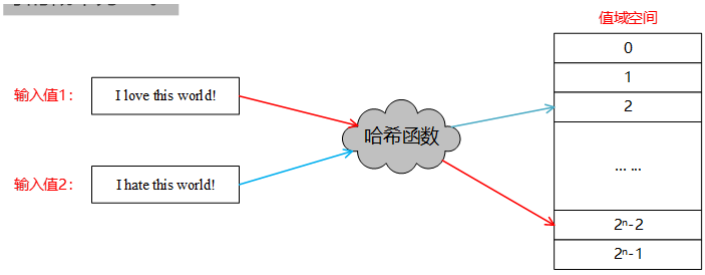


百科全书按主键范围进行分区的例子

根据主键的哈希值

哈希函数

> ①确定性：相同的输入会产生相同的输出；
>
> ②“随机性”：输入输出的对应完全是随机的。对于给定的输入字符串，函数输出“随机”落在值域空间的某个点上。两个输入值即便只相差1个bit，输出值可能相差很远。
>
> ③无碰撞性：假定函数输出n比特的整数，任取两个输入，它们的输出值相等的概率为2-n。



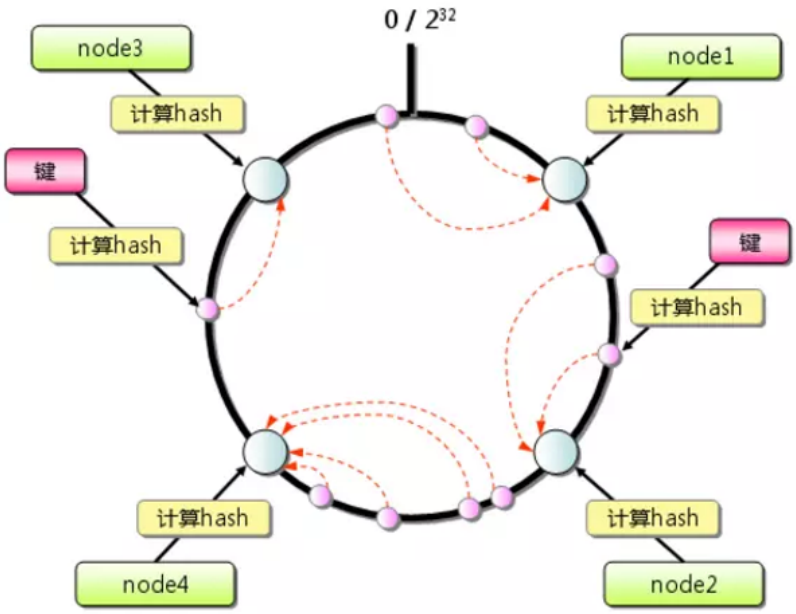
方案：

> ①将整个哈希空间均匀分成k个区间，每个存储节点负责一个哈希区间。
>
> ②计算新插入数据元素主键的哈希值，然后计算该哈希值落入了哪个哈希区间，最后将该数据元素分配给该哈希区间对应的存储节点。
>
> 在增加或删除物理节点时会产生数据移动。

优点：可以在一定程度上避免了偏斜和热点问题。 无须全局索引，因而也无须中心节点。

缺点：①基于主键进行连续范围查询效率极低。

②在物理存储节点较少时仍然会出现偏斜和热点问题。（可以采用虚拟节点的方法进行缓解）

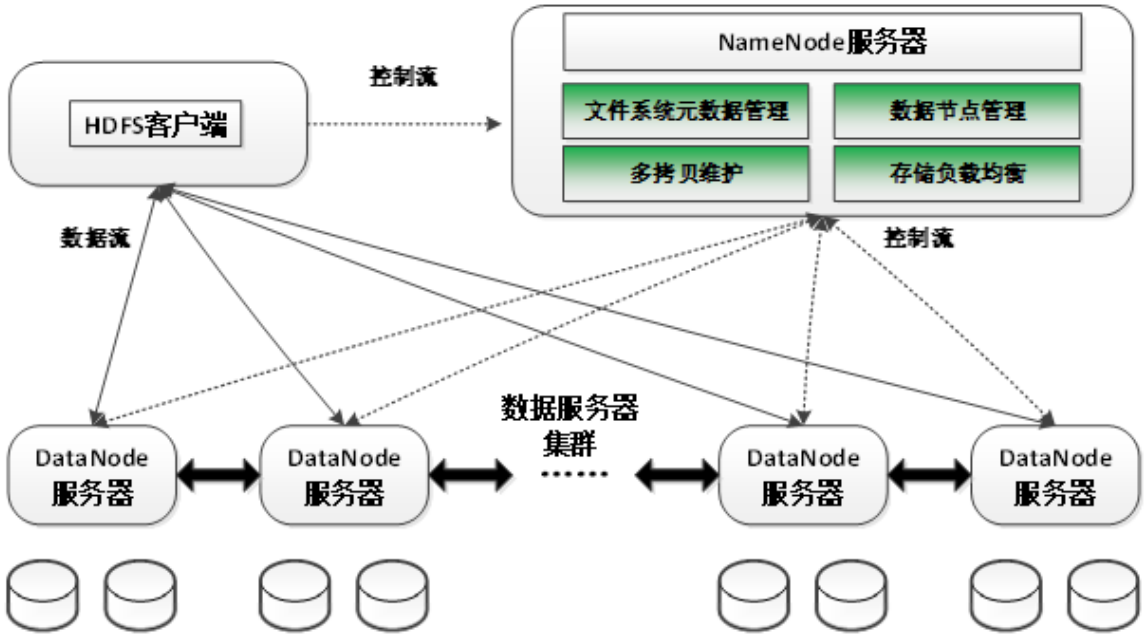


HDFS

工作原理：

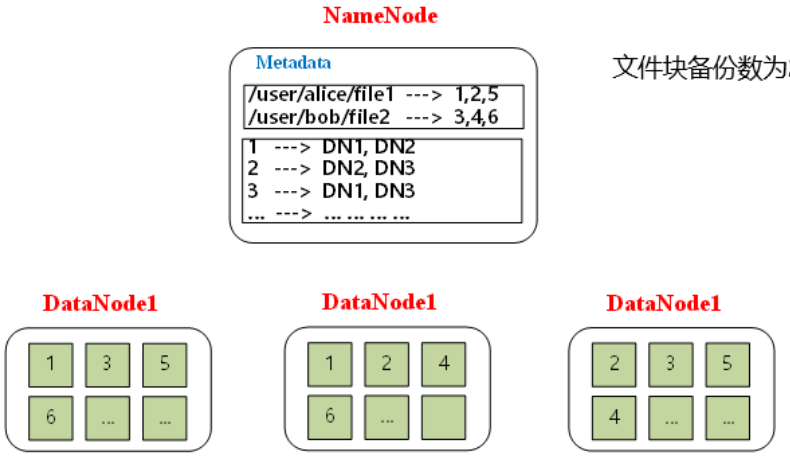
> - 客户端向 NameNode 发出写文件请求。
> - 检查是否已存在文件、检查权限。若通过检查，直接先将操作写入 EditLog ，并返回输出流对象。
> - clientclient端按端按128MB128MB的块切分文件的块切分文件。
> - clientclient将将NameNodeNameNode返回的分配的可写的返回的分配的可写的DataNodeDataNode列表和列表和DataData数据一同发送给最近的第一个数据一同发送给最近的第一个DataNodeDataNode节点，此后节点，此后clientclient端和端和NameNodeNameNode分配的多个分配的多个DataNodeDataNode构成构成pipelinepipeline管道，管道，clientclient端向输出端向输出流对象中写数据。流对象中写数据。clientclient每向第一个每向第一个DataNodeDataNode写入一个写入一个packetpacket，这个，这个packetpacket便会直接在便会直接在pipelinepipeline里传里传给第二个、第三个给第二个、第三个...DataNode...DataNode。
> - 每个每个DataNodeDataNode写完一个块后，会返回确认信息。写完一个块后，会返回确认信息。
> - 写完数据，关闭输输出流。写完数据，关闭输输出流。
> - 发送完成信号给发送完成信号给NameNodeNameNode。

Namenode 节点流程及作用：
namenode是整个文件系统 的管理节点。它维护着 1. 整个文件系统的文件目录树， 2. 文件 目录的元信息和每
个文件对应的数据块列表。 3. 接收用户的操作请求。
namenode 包含两个文件： FsImage(元数据镜像文件。存储某一时段 NameNode 内存元数据信息 和 Editlog(操
作日志文件)



NameNode 维护着 2 张表：

1. 文件与数据块 （ block ）列表的对应关系
2. 数据 块与 被存储 的 结点 的关系。



DataNode：负责存储 client 发来的数据块 block； 执行数据块的读写操作； 使用多备份策略。