几种常见的数据分区方法

我们使用数据库可以快速访问业务数据，但是随着时间的推移，数据库会不断增长，提取信息所需的时间也会更长，数据操作成为瓶颈。这时候我们就需要对数据进行分区（partition）了。分区是将数据库或其组成元素划分为不同的独立部分。数据库分区通常是出于可管理性、性能或可用性或负载平衡的原因而进行的。在分布式数据库管理系统中分区是很流行，其中每个分区可以分布在多个节点上，节点上的用户在分区上执行本地事务。由于数据的分区，使得系统的整体性能得以提升。

数据分区方法

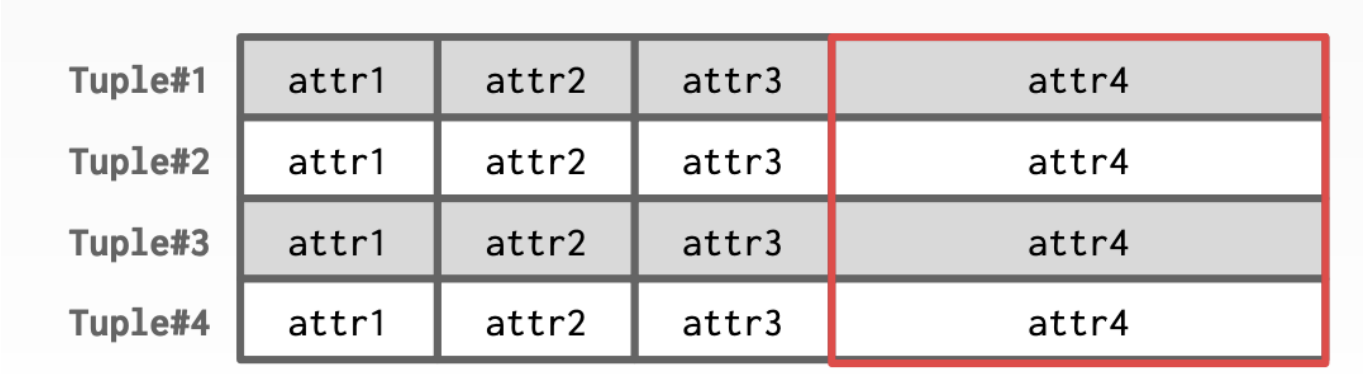
数据的分区方法（Partitioning methods）大概有以下几种：

- 垂直分区（Vertical partitioning）
- 水平分区（Horizontal partitioning）
- 混合分区（Hybrid partitioning）

垂直分区（Vertical partitioning）

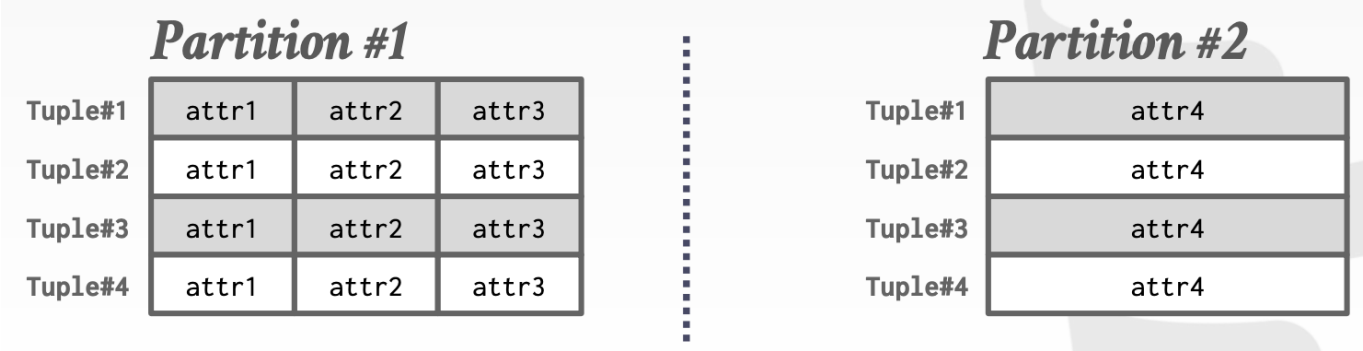
垂直分区需要创建一些较少列的表，每张表存储源表的部分列，以此达到数据的分区。比如我们有一张名为 iteblog 表，如下：

```
CREATE TABLE iteblog (
 attr1 INT,
 attr2 INT,
 attr3 INT,
 attr4 TEXT
);
```



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

使用垂直分区，可以将这张表拆分成以下形式：

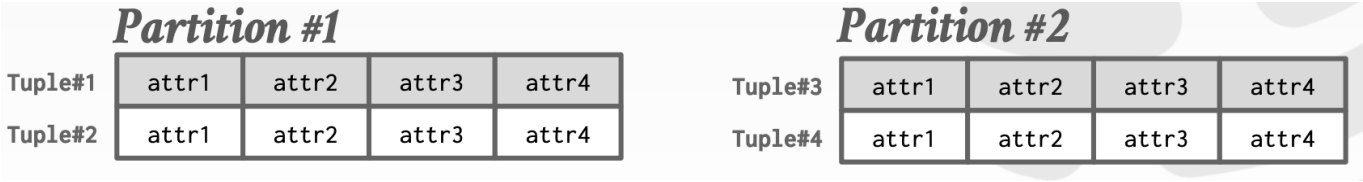


如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

这个在大数据数据仓库很常见，比如我们将一些数据量小，但是经常查询的数据放到 ES 中，数据量比较大的部分，但是不经常被查到放到 HBase 中。这种方法还可以根据说的访问频率，把不同的列数据存放到不同的存储介质中，以此节省存储成本。

水平分区（Horizontal partitioning）

水平分区分区也称为分片（sharding），其根据不同的分区算法将不同行的数据存储到不同的表中（比如关系型数据库中的分库分表）。例如，邮政编码小于50000 的客户存储在 CustomersEast 表中，而邮政编码大于或等于 50000 的客户存储在 CustomerWest 表中，所以分区表就是 CustomersEast 和 CustomersWest，这两张表加起来对外提供一个完整的视图。



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

分区算法

水平分区一般会选择表中的某列或某些列调用分区算法，计算其分区之后已经分到那张表中，这些被选中的列也称为 partitioning key，比较常见的分区算法有：

- Range partitioning:通过确定分区键是否在某个范围内来选择分区。比如 zipcode 列的值在 0 到 1000 之间属于分区 A；值在 1001 到 2000 之间属于分区 B；值在 2001 到 3000 之间属于分区 C；以此类推。我们熟悉的 HBase 表中 Region 的分区就是用这种方法进行的。
- Hash partitioning：这种分区算法也很常见。就是对选择的 partitioning key

计算其哈希值，得到的哈希值就是对应的分区。我们熟悉的 Kafka Topic 计算分区就是用这种分区算法的。这种分区算法理论上会将数据均匀分散到不同分区中。

- Round-robin partitioning ：这是最简单的分区算法，比如有3个分区，第一条数据放到第一个分区；第二条放到第二个分区；第三条数据放到第三个分区；第四条放到第一个分区；计算规则是 (i mod n)，其中 n 代表分区数，i 代表第几条数据，得到的模就是对应的分区。
- List partitioning: 为分区分配一个值列表。如果分区键具有这些值中的一个，则选择分区。例如，“国家/地区”列为“冰岛”，“挪威”，“瑞典”，“芬兰”或“丹麦”的所有行都可以选择北欧国家/地区的分区。
- Composite partitioning ： 允许上述分区模式的特定组合，例如，首先应用范围分区，然后应用哈希分区。一致性哈希（Consistent hashing）可以被认为是哈希（Hash partitioning）和列表分区（List partitioning）的组合。
-

混合分区（Hybrid partitioning）

这种分区结合了垂直和水平分区。比如我们有一个保存不同类型数据的大型数据，那么我们我们可以水平地对客户信息进行分区，然后再利用垂直将图片存储在Blob 存储中，比如下图所示：