**Hbase优化**[](#hbase)

**高可用**[](#_1)

在HBase中Hmaster负责监控RegionServer的生命周期，均衡RegionServer的负载，如果Hmaster挂了。那么整个HBase集群将陷入不健康的状态，并且此时的工作状态并不会维持太久。。

**预分区**[](#_2)

**预分区的作用？为什么要进行预分区？**[](#_3)

每个region都维护着startRow与endRowKey，如果加入的数据符合某个region维护的rowKey范围，则该数据交给这个region维护。那么依照这个原则，我们可以将数据说要投放的分区提前大致的规划好，以提高HBase性能。

**Rowkey设计**[](#rowkey)

在HBase中一条数据的唯一标识就是rowKey，那么这条数据存储于那个分区，取决于rowkey处于哪个预分区的区间内，设计rowkey的主要目的，就是让数据均匀的分布在所有的region中，一定程度上防止 **数据倾斜**。接下来我们就谈一谈rowkey常用的设计方案：

* 生成随机数、hash、散列值
  + 遵循散列原则
* 字符串反转
  + 遵循散列原则
  + 比如将日期时间戳进行翻转：2020年01月30日，（字典序排列的）
* 字符串拼接

**比如统计一个月的订单数据，rowkey怎么设置？**[](#rowkey_1)

**将201901的月份按照分区数进行取余**。

**rowkey设计的原则**[](#rowkey_2)

* 唯一性原则：类似于关系型数据库的主键
* 长度原则：满足业务要求的情况下尽可能的短
* 散列原则：让rowkey没有规律，可以均匀的分配

**HBase存储数据时如何分布到各个节点上？**[](#hbase_1)

1. 预切分
2. 看设计rowkey的能力
3. 根本上还是业务设计的能力
4. 我不是针对其他回答，我看了看答非所问，想起了一句霸气的话，在座的都是垃圾