1. 为何HBase速度很快?

HBase能提供实时计算服务主要原因是由其架构和底层的数据结构决定的，即由LSM-Tree(Log-Structured Merge-Tree) + HTable(region分区) + Cache决定——客户端可以直接定位到要查数据所在的HRegion server服务器，然后直接在服务器的一个region上查找要匹配的数据，并且这些数据部分是经过cache缓存的。

HBase的写入速度快是因为它其实并不是真的立即写入文件中，而是先写入内存，随后异步刷入HFile。所以在客户端看来，写入速度很快。另外，写入时候将随机写入转换成顺序写，数据写入速度也很稳定。

读取速度快是因为它使用了LSM树型结构，而不是B或B+树.HBase的存储结构导致它需要磁盘寻道时间在可预测范围内，并且读取与所要查询的rowkey连续的任意数量的记录都不会引发额外的寻道开销。而且，HBase读取首先会在缓存（BlockCache）中查找，它采用了LRU（最近最少使用算法），如果缓存中没找到，会从内存中的MemStore中查找，再去HFile查找

> https://zhuanlan.zhihu.com/p/83233850

2. hbase 实时查询的原理

实时查询，可以认为是从内存中查询，一般响应时间在 1 秒内。 HBase 的机制是数据先写入到内存中，当数据量达到一定的量（如 128M），再写入磁盘中， 在内存中，是不进行数据的更新或合并操作的，只增加数据，这使得用户的写操作只要进入内存中就可以立即返回，保证了 HBase I/O 的高性能。

3. 列簇怎么创建比较好？(<=2)

rowKey 最好要创建有规则的 rowKey，即最好是有序的。 HBase 中一张表最好只创建一到两个列族比较好，因为 HBase 不能很好的处理多个列族。

4. 描述 HBase 中 scan 和 get 的功能以及实现的异同?

HBase 的查询实现只提供两种方式：
1. 按指定 RowKey 获取唯一一条记录，get 方法（org.apache.hadoop.hbase.client.Get）Get 的方法处理分两种 : 设置了 ClosestRowBefore 和没有设置 ClosestRowBefore 的rowlock。主要是用来保证行的事务性，即每个 get 是以一个 row 来标记的。一个 row 中可以有很多 family 和 column。
2. 按指定的条件获取一批记录，scan 方法(org.apache.Hadoop.hbase.client.Scan）实现条件查询功能使用的就是 scan 方式。

> 原文链接：https://blog.csdn.net/shujuelin/article/details/89035272

5. 简述 HBase 中 compact 用途是什么，什么时候触发，分为哪两种，有什么区别?

在 hbase 中每当有 memstore 数据 flush 到磁盘之后，就形成一个 storefile，当 storeFile的数量达到一定程度后，就需要将 storefile 文件来进行 compaction 操作。
Compact 的作用：
- 合并文件
- 清除过期，多余版本的数据(删除的数据不会马上删除,只会被标记需要删除)
- 提高读写数据的效率

HBase 中实现了两种 compaction 的方式：minor and major. 这两种 compaction 方式的区别是：
1. Minor 操作只用来做部分文件的合并操作以及包括 minVersion=0 并且设置 ttl 的过期版本清理，不做任何删除数据、多版本数据的清理工作。

2. Major 操作是对 Region 下的 HStore 下的所有 StoreFile 执行合并操作，最终的结果是整理合并出一个文件。

> 需要注意的是 major 合并的时候，只会对一个列族的 HFile 进行合并。

> 原文链接：https://blog.csdn.net/shujuelin/article/details/89035272

6. HBase rowlock 有什么用

在初始化 Put 对象的时候，如果需要频繁地重复修改某些行（加锁），用户有必要创建一个 RowLock 实例来防止其他客户端访问这些行。