Hbase

- 部署
  - 安装hdfs
    - https://www.alexjf.net/blog/distributed-systems/hadoop-yarn-installation-definitive-guide/
    - 只需要启动hdfs（namenode和datanode），不需要用到yarn和MapReduce
  - hbase配置属性列表
    - https://hbase.apache.org/book.html#hbase_default_configurations
  - Web UI
    - 127.0.0.1:8088 -> resourcemaneger
    - http://127.0.0.1:16010 -> hbase
    - http://127.0.0.1:50070 -> namenode
  - Service port
    - 127.0.0.1:8020 -> namenode
- 基础
  - HBase中为什么要有Column Family
    - https://blog.csdn.net/liuj2511981/article/details/8599614
  - HBase架构
    - http://www.blogjava.net/DLevin/archive/2015/08/22/426877.html
    - http://www.blogjava.net/DLevin/archive/2015/08/22/426950.html
  - WAL
    - https://zh.wikipedia.org/wiki/%E9%A2%84%E5%86%99%E5%BC%8F%E6%97%A5%E5%BF%97
- 技巧和优化注意点
  - 1）减少调整
    - 根据你的RowKey设计来进行预建分区，减少region的动态分裂
    - 估计项目数据量大小，给HFile设定一个合适的值
  - 2）减少启停
    - 关闭Compaction，在闲时进行手动Compaction
    - 批量数据写入时采用BulkLoad（https://www.jianshu.com/p/8bd9862f8567）
  - 3）减少数据量
    - 开启BloomFilter，BloomFilter是列族级别的过滤，在生成一个StoreFile同时会生成一个MetaBlock，用于查询时过滤数据，但是不精确
    - 使用压缩：一般推荐使用Snappy和LZO压缩
  - 4）RowKey设计
    - 散列性：散列性能够保证相同相似的rowkey聚合，相异的rowkey分散，有利于查询
    - 简短性：rowkey作为key的一部分存储在HFile中，如果为了可读性将rowKey设计得过长，那么将会增加存储压力
    - 唯一性：rowKey必须具备明显的区别性
    - 业务性：举些例子
      - 假如我的查询条件比较多，而且不是针对列的条件，那么rowKey的设计就应该支持多条件查

询
- 如果我的查询要求是最近插入的数据优先，那么rowKey则可以采用叫上Long.Max-时间戳的方式，这样rowKey就是递减排列

- 5）列族的设计
  - 多列族设计的优劣
    - 优势：HBase中数据时按列进行存储的，那么查询某一列族的某一列时就不需要全盘扫描，只需要扫描某一列族，减少了读I/O；其实多列族设计对减少的作用不是很明显，适用于读多写少的场景
    - 劣势：降低了写的I/O性能。原因如下：数据写到store以后是先缓存在memstore中，同一个region中存在多个列族则存在多个store，每个store都有一个memstore，当memstore进行flush时，属于同一个region的store中的memstore都会进行flush，增加I/O开销

- HBase简单读写流程
  - 读
    - 找到要读取数据的region所在的RegionServer，然后按照以下顺序进行读取：先去BlockCache读取，若BlockCache没有，则到Memstore读取，若MemStore中没有，则到HFile中读取。
  - 写
    - 找到要写入数据的region所在的RegionServer，然后将数据先写到WAL中，然后再将数据写到MemStore等待刷新，回复客户端写入完成。
- HBase首次读写流程
  - Client从ZooKeeper中读取hbase:meta表
  - Client从hbase:meta中获取想要操作的region的位置信息，并且将hbase:meta缓存在Client端，用于后续的操作
  - 当一个RegionServer宕机而执行重定位之后，Client需要重新获取新的hase:meta信息进行缓存
- HBase和mysql的区别。
  - HBase基于列存储，查询中的选中规则是通过列来定义，因此整个数据库是自动索引化的。HBase无需考虑分库、分表，它可以对存储的数据自动切分数据，并支持高并发读写操作，使得海量数据存储自动具有更强的扩展性。但是HBase不包含事务，没有表与表之间关联查询，mysql基于行存储，mysql的innodb引擎带事务控制，表之间的join比较方便；伸缩性比较差。
- HBase二级索引的设计
  - https://www.cnblogs.com/MOBIN/p/5579088.html
  - Phoenix
- 预分区
  - https://blog.csdn.net/javajxz008/article/details/51913471
- Hregion如何实现自动切分
  - https://blog.csdn.net/HaixWang/article/details/79514983
- 最佳实践
  - https://zhuanlan.zhihu.com/p/27800787
  - https://www.tuicool.com/articles/EbimM3Q
  - 阿里云（https://help.aliyun.com/document_detail/59373.html?spm=a2c4g.11186623.6.573.q2EP6C）
  - 滴滴（https://cloud.tencent.com/developer/article/1042669）
    - 早期版本需要考虑rowkey排序的问题，rowkey需要reverse timestamp的设计来方便获取最新的数据（因为hbase是字典顺序排序，最新插入的时间会是最小的值，排序后就在最前面），现在有number of version机制，默认返回的数据是最新版本的数据，如果需要scan之前的数据，需要带

上version属性

- 剖析HBase负载均衡和性能指标
  - http://www.cnblogs.com/smartloli/p/9249259.html
- ScanApi
  - https://www.jianshu.com/p/5411bfb4abd6
- rowkey的设计
  - https://zhuanlan.zhihu.com/p/30074408