**Redis**

***1. redis支持的5种数据类型***

String, Hashs(散列值，可以存储对象)，List，Sets，Sorted Set(zset,是set的升级版本，可以对元素排序，每次添加新元素后，都会对元素进行重新排序)

***2. 速度快***

Redis使用标准C语言编写，将数据加载到内存当中，所以存取速度非常快；官方数据表明，普通的Linux机器上Redis的读写速度分别达到8W和10W每秒

***3. 数据持久化***

Redis提供策略可以将数据异步的持久化到磁盘当中。appendfsync everysec --每秒执行一次，Redis默认开启这个

***4. 主从复制***

非凡之星就是用到主从复制，没有用到集群；官方提供的一个数据：Slave在21秒内完成10G的key set同步

***5. 集群***

可以通过自动发现将数据分散在各个节点中。比较常用的好像是Ruby语言编写的工具实现集群。

***6. String***

value不仅仅是String，也可以是int，因为Redis内部实现是单线程的，所以可以认为get,set,incr,decr等操作是线程安全的。

***7. redis有三种存储介质***

内存存储，磁盘存储和log文件存储。

因为redis是使用异步的方式将数据持久化到磁盘的，所以如果考虑到机器断电造成数据丢失。可以使用AOF方式，具体看AOF的相关描述

appendonly yes:表示每次更新后进行日志记录，而redis本身数据同步是根据这个日志文件上的save条件来同步的。

***8. redis两种持久化方式***

快照(Snapshotting):他是默认的持久化方式，将内存中的数据以快照的方式写入二进制文件中(dump.db)，可以配置持久化策略：redis在几秒内如果好过多少个key被修改，那么就生成快照。

save 900 1 #900秒内如果超过1个key被修改，则发起快照保存

AOF方式：比快照方式更可靠些，redis每收到一个写命令时，都会通过write函数追加到appendonly.aof文件中。

appendonly yes:表示每次更新后进行日志记录，而redis本身数据同步是根据这个日志文件上的save条件来同步的。

***9. 主从复制***

1个master可以有多个Slave

Slave可以跟Slave相连接

其实也可以这么做，禁用master的持久化，使用slave做持久化。

同步过程:master会开启一个后台线程，将数据库快照保存到文件中。master将文件发送给slave，slave将文件保存到磁盘上，并且恢复快照内容到本db上

来自资料:Slave初始化的时候会使用全量复制。(redis2.8之后才增加增量同步机制)

1. 从服务器初始化是向master发送sync命令，master生成快照文件，并向所有的从库发送这个文件。
2. 此后master执行所有的写命令.master快照发送完毕之后开始想从库缓冲区发送写命令
3. 存库会将丢弃原有的数据，并根据master发过来的快照文件生成数据。
4. 从库开始从缓冲区执行master发送过来的写命令

从库从新开启后将重新同步所有的master数据。

10. 关于Redis主从复制

因为为了机器宕机考虑，并且本身数据还没有形成一定的规模，所以，master我使用的是AOF模式。

因为主备复制是基于快照的方式的，而且不是增量的，他将整个库的快照发送给从库。所以这种情况就是master同时使用快照和AOF模式.

10.