### 一、 redis与memcache的区别

**1.1** redis在数据支持上要比memecache多的多，Redis除K-V之外，还支持list,set,sorted set,hash等众多数据结构，

**1.2** Redis和Memcache都是将数据存放在内存中，都是内存数据库。不过memcache还可用于缓存其他东西，例如图片、视频等等。

**1.3** 虚拟内存–Redis当物理内存用完时，可以将一些很久没用到的value 交换到磁盘 。Redis采用hash结构来做K-V存储，由于其组合式的压缩，其内存利用率会高于Memcache。

**1.4** redis支持数据持久化，redis有部份存在硬盘上，这样能保证数据的持久性；memcache不支持数据持久化，断电就要重启。

**1.5** 性能:由于Redis只使用单核,而Memcache可以使用多核，所以平均每一个核上Redis在存储小数据时比Memcache性能更高。而在100K以上的数据时，Memcache性能要高于Redis。

**1.6** 数据持久化和主从复制时，只有redis拥有这两个特性。如果目标是构建一个缓存在升级或者重启后之前的数据不会丢失的话，那也只能选择redis。

**1.7** 如果需要缓存的数据只是key-value这样简单的结构时，采用memcache，它也足够的稳定可靠。如果涉及到存储，排序等一系列复杂的操作时，毫无疑问选择redis。

### 二、redis和MongoDB区别

**2.1** MongoDB与Redis都依赖内存，但redisTPS高于MongoDB

**2.2** MongoDB有丰富的数据表达,索引;最类似于关系型数据库,支持丰富的查询语句，redis数据丰富,较少的IO

**2.3** MongoDB适合大数据量存储,依赖系统虚拟内存,采用镜像文件存储;内存占用率比较高,官方建议独立部署在64位系统，Redis2.0后支持虚拟内存特性(VM) 突破物理内存限制;数据可以设置时效性,类似于memcache

**2.4** MongoDB支持master-slave,replicatset(内部采用paxos选举算法,自动故障恢复),auto sharding机制,对客户端屏蔽了故障转移和切片机制，redis依赖客户端来实现分布式读写;主从复制时,每次从节点重新连接主节点都要依赖整个快照,无增量复制;不支持auto sharding,需要依赖程序设定一致性hash机制

**2.5** MongoDB从1.8版本后,采用binlog方式(类似Mysql) 支持持久化，redis依赖快照进行持久化;AOF增强可靠性;增强性的同时,影响访问性能

**2.6** MongoDB不支持事务,靠客户端保证，redis支持事务,比较脆,仅能保证事务中的操作按顺序执行，而且MongoDB内置数据分析功能，redis不支持数据分析，所以， MongoDB海量数据的访问效率提升，相比之下redis较小数据量的性能和运算

**2.7** 当TPS(每秒传输的事物处理个数)非常高的时候，可以用到redis中数据持久化和分布式锁的内容，通过redis数据持久化，我们可以将缓存的数据保存到本地中来。此时使用redis较好。

**2.8** 当拥有海量数据但要求有限查询时间可以采用mongodb中的分片来实现，通过mongodb的分片机制，我们可以将海量的数据查询分别负载到不同的分片服务器上面，最后将数据查询的数据结果整合到一起。基于这种情况，不管数据量有多大，我们都可以实现快速的查询功能，查询时间约等于（数据量/分片数量）。分片其实本身就是一种高可用性的方案，因为每一个分片都保留着完整的一份数据，每次插入数据的时候，先插入一个主分片中，然后同步复制到所有从分片中，即使一个分片挂了，其余分片也能自动升级为主分片，继续工作。

### 二、redis的五种数据类型

**1.字符串类型String**

1.1 字符串类型是Redis的最基本类型，它可以存储任何形式的字符串。其它的四种类型都是字符串类型的不同形式。

1.2 最基本的命令：GET、SET      语法：GET key，SET key value   value如果有空格需要双引号以示区分

1.3 整数递增：INCR           语法：INCR key    默认值为0，所以首先执行命令得到 1 ，不是整型提示错误

1.4 增加指定的整数：INCRBY      语法：INCRBY key increment

1.5 整数递减：DECR           语法：DECR key   默认值为0，所以首先执行命令得到 -1，不是整型提示错误

1.6 减少指定的整数：DECRBY      语法：DECRBY key increment

1.7增加指定浮点数：INCRBYFLOAT 语法：INCRBYFLOAT key increment  与INCR命令类似，只不过可以递增一个双精度浮点数

1.8 向尾部追加值：APPEND        语法：APPEND key value   redis客户端并不是输出追加后的字符串，而是输出字符串总长度

1.9 获取字符串长度：STRLEN       语法：STRLEN key  如果键不存在返回0，注意如果有中文时，一个中文长度是3，redis是使用UTF-8编码中文的

1.10获取多个键值：MGET         语法：MGET key [key ...]  例如：MGET key1 key2

1.11设置多个键值：MSET         语法：MSET key value [key value ...]  例如：MSET key1 1 key2 "hello redis"

1.12 二进制指定位置值：GETBIT    语法：GETBIT key offset   例如：GETBIT key1 2 ，key1为hello 返回 1，返回的值只有0或1，当key不存在或超出实际长度时为0

1.13 设置二进制位置值：SETBIT    语法：SETBIT key offset value ，返回该位置的旧值

1.14 二进制是1的个数：BITCOUNT   语法：BITCOUNT key [start end] ，start 、end为开始和结束字节

1.15 位运算：BITOP           语法：BITOP operation destkey key [key ...]  ，operation支持AND、OR、XOR、NOT

1.16偏移：BITPOS            语法：BITPOS key bit [start] [end]

1. **散列型Hash**

2.1 设置单个：HSET                 语法：HSET key field value，不存在时返回1，存在时返回0，没有更新和插入之分

2.2 设置多个：HMSET                语法：HMSET key field value [field value ...]

2.3 读取单个：HGET                  语法：HGET key field，不存在是返回nil

2.4 读取多个：HMGET                语法：HMGET key field [field ...]

2.5 读取全部：HGETALL              语法：HGETALL key，返回时字段和字段值的列表

2.6 判断字段是否存在：HEXISTS     语法：HEXISTS key field，存在返回1 ，不存在返回0

2.7 字段不存在时赋值：HSETNX      语法：HSETNX key field value，与hset命令不同，hsetnx是键不存在时设置值

2.8 增加数字：HINCRBY              语法：HINCRBY key field increment ，返回增加后的数，不是整数时会提示错误

2.9 删除字段：HDEL                   语法：HDEL key field [field ...] ，返回被删除字段的个数

2.10 只获取字段名：HKEYS           语法：HKEYS key ，返回键的所有字段名

2.11只获取字段值：HVALS            语法：HVALS key  ，返回键的所有字段值

2.12 字段数量：HLEN                  语法：HLEN key ，返回字段总数

1. **列表类型List**

3.1 添加左边元素：LPUSH               语法：LPUSH key value [value ...]  ，返回添加后的列表元素的总个数

3.2 添加右边元素：RPUSH              语法：RPUSH key value [value ...]  ，返回添加后的列表元素的总个数

3.3 移除左边第一个元素：LPOP        语法：LPOP key  ，返回被移除的元素值

3.4 移除右边第一个元素：RPOP        语法：RPOP key ，返回被移除的元素值

3.5 列表元素个数：LLEN                语法：LLEN key， 不存在时返回0，redis是直接读取现成的值，并不是统计个数

3.6 获取列表片段：LRANGE           语法：LRANGE key start stop，如果start比stop靠后时返回空列表，0 -1 返回整个列表 ，正数时：start 开始索引值，stop结束索引值（索引从0开始），负数时：例如 lrange num -2 -1，-2表示最右边第二个，-1表示最右边第一个

3.7 删除指定值：LREM                 语法：LREM key count value，返回被删除的个数 count>0，从左边开始删除前count个值为value的元素 ，count<0，从右边开始删除前|count|个值为value的元素count=0，删除所有值为value的元素

3.8 索引元素值：LINDEX               语法：LINDEX key index ，返回索引的元素值，-1表示从最右边的第一位

3.9 设置元素值：LSET                  语法：LSET key index value

3.10 保留列表片段：LTRIM              语法：LTRIM key start stop，start、top 参考lrange命令

3.11 一个列表转移另一个列表：RPOPLPUSH      语法：RPOPLPUSH source desctination ，从source列表转移到desctination列表， 该命令分两步看，首先source列表RPOP右移除，再desctination列表LPUSH

1. **集合类型Set**

4.1 添加元素：SADD               语法：SADD key member [member ...] ，向一个集合添加一个或多个元素，因为集合的唯一性，所以添加相同值时会被忽略。返回成功添加元素的数量。

4.2 删除元素：SREM                语法：SREM key member [member ...] 删除集合中一个或多个元素，返回成功删除的个数。

4.4 获取全部元素：SMEMBERS     语法：SMEMBERS key ，返回集合全部元素

4.5 值是否存在：SISMEMBER      语法：SISMEMBER key member ，如果存在返回1，不存在返回0

4.6 差运算：SDIFF                  语法：SDIFF key [key ...] ，例如：集合A和集合B，差集表示A-B，在A里有的元素B里没有，返回差集合；多个集合(A-B)-C

4.7 交运算：SINTER             　语法：SINTER key [key ...]，返回交集集合，每个集合都有的元素

4.8 并运算：SUNION　　　　　　 语法：SUNION key [key ...]，返回并集集合，所有集合的元素

4.9 集合元素个数：SCARD         语法：SCARD key ，返回集合元素个数

4.10 集合运算后存储结果            语法：SDIFFSTROE destination key [key ...] ，差运算并存储到destination新集合中, SINTERSTROE destination key [key ...]，交运算并存储到destination新集合中, SUNIONSTROE destination key [key ...]，并运算并存储到destination新集合中

4.11 随机获取元素：SRANDMEMGER 语法：SRANDMEMBER key [count]，根据count不同有不同结果，count大于元素总数时返回全部元素,　count>0 ，返回集合中count不重复的元素,count<0，返回集合中count的绝对值个元素，但元素可能会重复

4.12 弹出元素：SPOP               语法：SPOP key [count] ，因为集合是无序的，所以spop会随机弹出一个元素

1. **有序集合类型**

5.1 添加集合元素：ZADD            语法：ZADD key [NX|XX] [CH] [INCR] score member [score member ...]，不存在添加，存在更新。

5.2 获取元素分数：ZSCORE         语法：ZSCORE key member ，返回元素成员的score 分数

5.3 元素小到大：ZRANGE            语法：ZRANGE key start top [WITHSCORES] ，参考LRANGE ，加上withscores 返回带元素，即元素，分数 ,当分数一样时，按元素排序

5.4元素大到小：ZREVRANGE       语法：ZREVRANGE key start [WITHSCORES] ，与zrange区别在于zrevrange是从大到小排序

5.5指定分数范围元素：ZRANGEBYSCORE   语法：ZRANGEBYSCORE key min max [WITHSCORE] [LIMIT offest count],返回从小到大的在min和max之间的元素