Redis

Redis是一个支持网络、基于内存、可选持久性的NoSql数据库，目前在很多的系统中都使用了Redis，尤其是在实现缓存功能的时候应用的尤其广泛（缓存功能也是很多人对Redis的认识），那么Redis到底有哪些优点和缺点，为什么会被广泛应用呢？

Redis的优点

Redis的第一个优点就是速度快，Redis使用C语言实现，基于内存，数据的读写效率非常的高，这也是为什么很多系统的缓存功能使用Redis来实现，但是需要明确的是Redis是一个数据库，缓存只是它的一项应用而已。

Redis的第二个优点是单线程模型，所谓单线程模型就是每一个请求都会有一个全新的线程来进行处理，这一点类似于Struts2，每一个请求都会有一个新的线程来进行处理。这样做的好处就是避免了线程频繁切换带来的系统开销，同时也避免了让人头疼的多线程问题。

Redis的第三个优点就是使用了非阻塞I/O，不在网络上浪费时间，进一步提高了效率。
Redis的第四个优点就是支持多种的数据类型，并且每一种数据类型都提供了丰富的操作命令，适用于很多特殊的场景，并且支持自定义命令创建个性化的操作命令。

Redis的数据类型

数据类型	可以存储的值	操作
STRING	字符串、整数或者浮点数	对整个字符串或者字符串的其中一部分执行操作 对整数和浮点数执行自增或者自减操作
LIST	列表	从两端压入或者弹出元素 读取单个或者多个元素 进行修剪，只保留一个范围内的元素
SET	无序集合	添加、获取、移除单个元素 检查一个元素是否存在于集合中 计算交集、并集、差集 从集合里面随机获取元素
HASH	包含键值对的无序散列表	添加、获取、移除单个键值对 获取所有键值对 检查某个键是否存在
ZSET	有序集合	添加、获取、删除元素 根据分值范围或者成员来获取元素 计算一个键的排名

String（字符串）

String是简单的 key-value 键值对，value 不仅可以是 String，也可以是数字。String在redis内部存储默认就是一个字符串，被redisObject所引用，当遇到incr,decr等操作时会转成数值型进行计算，此时redisObject的encoding字段为int。

String在redis内部存储默认就是一个字符串，被redisObject所引用，当遇到incr,decr等操作时会转成数值型进行计算，此时redisObject的encoding字段为int。

List（列表）

Redis列表是简单的字符串列表，简单的说就是一个链表或者说是一个队列。可以从头部或尾部向Redis列表添加元素。列表的最大长度为2^32 - 1，也即每个列表支持超过40亿个元素。

Redis list的实现为一个双向链表，即可以支持反向查找和遍历，更方便操作，不过带来了部分额外的内存开销，Redis内部的很多实现，包括发送缓冲队列等也都是用的这个数据结构。

Hash（字典，哈希表）

Redis Hash对应Value内部实际就是一个HashMap，实际这里会有2种不同实现，这个Hash的成员比较少时Redis为了节省内存会采用类似一维数组的方式来紧凑存储，而不会采用真正的HashMap结构，对应的value redisObject的encoding为zipmap,当成员数量增大时会自动转成真正的HashMap。

Set（集合）

可以理解为一堆值不重复的列表，类似数学领域中的集合概念，且Redis也提供了针对集合的求交集、并集、差集等操作。

set 的内部实现是一个 value永远为null的HashMap，实际就是通过计算hash的方式来快速排重的，这也是set能提供判断一个成员是否在集合内的原因。

Sorted Set（有序集合）

Redis有序集合类似Redis集合，不同的是增加了一个功能，即集合是有序的。一个有序集合的每个成员带有分数，用于进行排序。

Redis有序集合添加、删除和测试的时间复杂度均为O(1)(固定时间，无论里面包含的元素集合的数量)。列表的最大长度为2^32- 1元素(4294967295，超过40亿每个元素的集合)。

Redis sorted set的内部使用HashMap和跳跃表(SkipList)来保证数据的存储和有序，HashMap里放的是成员到score的映射，而跳跃表里存放的是所有的成员，排序依据是HashMap里存的score,使用跳跃表的结构可以获得比较高的查找效率，并且在实现上比较简单。

Redis能够做什么

缓存

缓存功能就具体解释了，Redis大部分的应用场景就是缓存，合理的使用能够提升服务器性能。

排行榜

利用Redis的SortSet数据结构

计算器/限速器

利用Redis中原子性的自增操作，我们可以统计类似用户点赞数、用户访问数等，这类操作如果用MySQL，频繁的读写会带来相当大的压力

限速器比较典型的使用场景是限制某个用户访问某个API的频率，常用的有抢购时，防止用户疯狂点击带来不必要的压力

好友关系

利用集合的一些命令，比如求交集、并集、差集等。可以方便搞定一些共同好友、共同爱好之类的功能

简单消息队列

Redis自身的发布/订阅模式

也可以利用List来实现一个队列机制实现到货通知、邮件发送之类的需求，不需要高可靠，但是会带来非常大的DB压力，完全可以用List来完成异步解耦

Session共享

可以用在Session共享服务器。

在我看来，Redis适合以下场景：

- 共享Cache ，不怕丢数据，丢了可以从DB中reload；
- 共享Session ，不怕丢数据，丢了可以重新登录；
- batch job的中间结果。不怕丢数据，丢了重新跑job就可以了；
- 一些简单数据的存储，低频改动，但是会被频繁读取。比如首页推荐的产品列表。但此时必须增加HA的防护，sentinel、cluster或者自定义的机制都可以；
- 一些更加复杂存储的building block，比如分布式锁，此时需要多节点来实现一个简单的quorum

其他场景，往往有更好的、更成熟的方案。
特别注意，不要用Redis存储任何需要“认真对待”的数据，请用支持ACID事务的数据库。

缓存介绍

1）定义

缓存就是在内存中存储的数据备份，当数据没有发生本质变化的时候，我们避免数据的查询操作直接连接数据库，而是去 内容中读取数据，这样就大大降低了数据库的读写次数，而且从内存中读数据的速度要比从数据库查询要快很多。

2）缓存的形式

页面缓存（smarty静态化技术）：页面缓存经常用在CMS（content manage system）内存管理系统里面。

数据缓存：经常会用在页面的具体数据里面。

Redis介绍

Redis是Remote Dictionary Server(远程数据服务)的缩写，由意大利人antirez(Salvatore Sanfilippo)开发的一款内存高速缓存数据库，该软件使用C语言编写，它的数据模型为key-value。

它支持丰富的数据结构(类型)，比如String/List/Hash/Set/Sorted Set。

可持久化(一边运行，一边把数据往硬盘中备份一份，防止断电等情况导致数据丢失，等断电情况恢复之后，Redis再把硬盘中的数据恢复到内存中)，保证了数据的安全。

4、为什么选择Redis

1）Redis不仅仅支持简单的k/v类型的数据，同时还提供list，set，zset，hash等数据结构的存储。

2）Redis支持master-slave(主-从)模式应用

3）Redis支持数据持久化，可以将内存中的数据保持在磁盘中，重启的时候可以再次加载进行使用。

4）Redis单个value的最大限制是1GB，memcached只能保存1MB的数据。