1. redis入门

一、什么是BSD？

BSD全称：Berkeley Software Distribution license: BSD开源协议是一个给于使用者很大自

由的协议。可以自由的使用，修改源代码，也可以将修改后的代码作为开源或者专有软件再发布。当你发布使用了BSD协议的代码，或者以BSD协议代码为基础做二次开发自己的产品时，需要满足三个条件：

1、如果再发布的产品中包含源代码，则在源代码中必须带有原来代码中的BSD协议。

2、如果再发布的只是二进制类库/软件，则需要在类库/软件的文档和版权声明中包含原来代

码中的BSD协议。

3、不可以用开源代码的作者/机构名字和原来产品的名字做市场推广。

BSD代码鼓励代码共享，但需要尊重代码作者的著作权。BSD由于允许使用者修改和重新发布代码，也允许使用或在BSD代码上开发商业软件发布和销 售，因此是对商业集成很友好的协议。很多的公司企业在选用开源产品的时候都首选BSD协议，因为可以完全控制这些第三方的代码，在必要的时候可以修改或者 二次开发.

二、什么是Redis？

REmote DIctionary Server(Redis) 是一个由Salvatore Sanfilippo写的开源免费、key-value存储系统。Redis是一个开源的使用ANSI C语言编写、遵守BSD协议、支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API。它通常被称为数据结构服务器，因为值（value）可以是 字符串(String), 哈希(Map), 列表(list), 集合(sets) 和 有序集合(sorted sets)等类型。

三、Redis 优势

性能极高:Redis能读的速度是110000次/s,写的速度是81000次/s 。

丰富的数据类型:Redis支持二进制案例的 Strings, Lists, Hashes, Sets 及 Ordered Sets 数据类型操作。

原子:Redis的所有操作都是原子性的，同时Redis还支持对几个操作全并后的原子性执行。

丰富的特性:Redis还支持 publish/subscribe, 通知, key过期等等特性。

四、Redis缺点：

（1）单线程

（2）耗内存

适用于数据变化快且数据库大小可遇见（适合内存容量）的应用程序。

Redis与其他key-value存储有什么不同？

Redis有着更为复杂的数据结构并且提供对他们的原子性操作，这是一个不同于其他数据库的进化路径。Redis的数据类型都是基于基本数据结构的同时对程序员透明，无需进行额外的抽象。

Redis运行在内存中但是可以持久化到磁盘，所以在对不同数据集进行高速读写时需要权衡内存，因为数据量不能大于硬件内存。

在内存数据库方面的另一个优点是，相比在磁盘上相同的复杂的数据结构，在内存中操作起来非常简单，这样Redis可以做很多内部复杂性很强的事情。同时，在磁盘格式方面他们是紧凑的以追加的方式产生的，因为他们并不需要进行随机访问。

五、几种noSQL数据库的比较

# [几款主流 NoSql 数据库](http://www.cnblogs.com/vajoy/p/5471308.html)：HBase、Redis、MongoDB、Couchbase、LevelDB等。

5.1、HBase

HBase 是 Apache Hadoop 中的一个子项目，属于 bigtable 的开源版本，所实现的语言为Java（故依赖 Java SDK）。HBase 依托于 Hadoop 的 HDFS（分布式文件系统）作为最基本存储基础单元。

优势

1. 存储容量大，一个表可以容纳上亿行，上百万列；

2. 可通过版本进行检索，能搜到所需的历史版本数据；

3. 负载高时，可通过简单的添加机器来实现水平切分扩展，跟Hadoop的无缝集成保障了其数据可靠性（HDFS）和海量数据分析的高性能（MapReduce）；

4. 在第3点的基础上可有效避免单点故障的发生。

缺点

1. 基于Java语言实现及Hadoop架构意味着其API更适用于Java项目；

2. node开发环境下所需依赖项较多、配置麻烦（或不知如何配置，如持久化配置），缺乏文档；

3. 占用内存很大，且鉴于建立在为批量分析而优化的HDFS上，导致读取性能不高；

4. API相比其它 NoSql 的相对笨拙。

适用场景

1. bigtable类型的数据存储；

2. 对数据有版本查询需求；

3. 应对超大数据量要求扩展简单的需求

1.2、MongoDB

MongoDB 是一个高性能，开源，无模式的文档型数据库，开发语言是C++。它在许多场景下可用于替代传统的关系型数据库或键/值存储方式。

优势

1、强大的自动化 shading 功能；

2、全索引支持，查询非常高效；

3、面向文档（BSON）存储，数据模式简单而强大。

4、支持动态查询，查询指令也使用JSON形式的标记，可轻易查询文档中内嵌的对象及数组。

5、支持 javascript 表达式查询，可在服务器端执行任意的 javascript函数。

缺点

1. 单个文档大小限制为16M，32位系统上，不支持大于2.5G的数据；

2. 对内存要求比较大，至少要保证热数据（索引，数据及系统其它开销）都能装进内存；

3. 非事务机制，无法保证事件的原子性。

适用场景

1. 适用于实时的插入、更新与查询的需求，并具备应用程序实时数据存储所需的复制及高度伸缩性；

2. 非常适合文档化格式的存储及查询；

3. 高伸缩性的场景：MongoDB 非常适合由数十或者数百台服务器组成的数据库。

4. 对性能的关注超过对功能的要求。

## 六、Memcached和Redis的比较

Redis的作者Salvatore Sanfilippo曾经对这两种基于内存的数据存储系统进行过比较，总体来看还是比较客观的，现总结如下：

1）性能对比：由于Redis只使用单核，而Memcached可以使用多核，所以平均每一个核上Redis在存储小数据时比Memcached性能更高。而在100k以上的数据中，Memcached性能要高于Redis，虽然Redis最近也在存储大数据的性能上进行优化，但是比起 Memcached，还是稍有逊色。

2）内存使用效率对比：使用简单的key-value存储的话，Memcached的内存利用率更高，而如果Redis采用hash结构来做key-value存储，由于其组合式的压缩，其内存利用率会高于Memcached。

3）Redis支持服务器端的数据操作：Redis相比Memcached来说，拥有更多的数据结构和并支持更丰富的数据操作，通常在Memcached 里，你需要将数据拿到客户端来进行类似的修改再set回去。这大大增加了网络IO的次数和数据体积。在Redis中，这些复杂的操作通常和一般的 GET/SET一样高效。所以，如果需要缓存能够支持更复杂的结构和操作，那么Redis会是不错的选择。

4) 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。

5) Redis支持数据的备份，即master-slave模式的数据备份。

6) Redis支持数据的持久化，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

7)使用场合的比较

1.Redis使用最佳方式是全部数据in-memory。

2.Redis更多场景是作为Memcached的替代者来使用。

3.当需要除key/value之外的更多数据类型支持时，使用Redis更合适。

4.当存储的数据不能被剔除时，使用Redis更合适。

附加：

master-slave模式： Master就是主盘的意思,Slave就是从盘的意思  主人；奴隶  
一般来说用在数据库集群比较多，主要是实现读写分离。对于数据库应用而言基本上是读大

于写，因此由 Master 服务器负责增、删、改操作，由 Slave 负责读操作（也就是 SELECT）

Master 一般只有一台，而 Slave 可以有好多台。Slave 与 Master 之间会有心跳数据包（一般数

据库服务器会提供配置）。当 Master 有数据写入时 Master 会将数据同步至各 Slave 上。如果数

据库采用 Master-Slave 的话，那对于应用程序来说是透明的，也就是说，不管用不

用 Master-Slave，我们的代码不需要进行修改。