**几种NOSQL基本介绍的汇总笔记**

# 常见数据库

1.key-value 数据库；例如：redius，memcache，等

2.宽表/高表设计，非结构化分布式存储；例如：hbase等

3.支持快速搜索的文档类型数据库，例如：elasticsearch，solr等

4.图形结构关系数据库；例如：neo4j等

5.数仓计算类数据库：hive，pig等

# 一、Redis 教程



REmote DIctionary Server(Redis) 是一个由Salvatore Sanfilippo写的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。

**Redis 简介**

Redis 是完全开源免费的，遵守BSD协议，是一个高性能的key-value数据库。

Redis 与其他 key - value 缓存产品有以下三个特点：

* Redis支持数据的持久化，可以将内存中的数据保存在磁盘中，重启的时候可以再次加载进行使用。
* Redis不仅仅支持简单的key-value类型的数据，同时还提供list，set，zset，hash等数据结构的存储。
* Redis支持数据的备份，即master-slave模式的数据备份。

**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，意思就是要么成功执行要么失败完全不执行。单个操作是原子性的。多个操作也支持事务，即原子性，通过MULTI和EXEC指令包起来。
* 丰富的特性 – Redis还支持 publish/subscribe, 通知, key 过期等等特性。

**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。

**二、Memcached**

Memcached是一个自由开源的，高性能，分布式内存对象缓存系统。

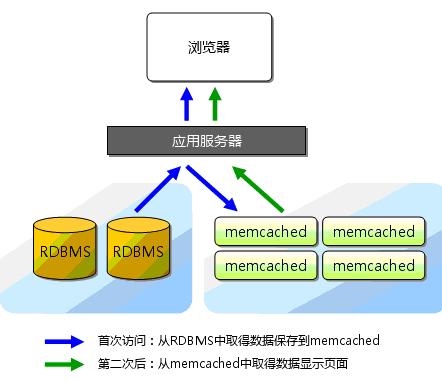
Memcached是以LiveJournal旗下Danga Interactive公司的Brad Fitzpatric为首开发的一款软件。现在已成为mixi、hatena、Facebook、Vox、LiveJournal等众多服务中提高Web应用扩展性的重要因素。

Memcached是一种基于内存的key-value存储，用来存储小块的任意数据（字符串、对象）。这些数据可以是数据库调用、API调用或者是页面渲染的结果。

Memcached简洁而强大。它的简洁设计便于快速开发，减轻开发难度，解决了大数据量缓存的很多问题。它的API兼容大部分流行的开发语言。

本质上，它是一个简洁的key-value存储系统。

一般的使用目的是，通过缓存数据库查询结果，减少数据库访问次数，以提高动态Web应用的速度、提高可扩展性。



Memcached 官网：[http://memcached.org/](http://memcached.org/" \t "_blank)。

**特征**

memcached作为高速运行的分布式缓存服务器，具有以下的特点。

* 协议简单
* 基于libevent的事件处理
* 内置内存存储方式
* memcached不互相通信的分布式

**支持的语言**

许多语言都实现了连接memcached的客户端，其中以Perl、PHP为主。仅仅memcached网站上列出的有：

* Perl
* PHP
* Python
* Ruby
* C#
* C/C++
* Lua
* 等等

**Memcached 用户**

* LiveJournal
* Wikipedia
* Flickr
* Bebo
* Twitter
* Typepad
* Yellowbot
* Youtube
* WordPress.com
* Craigslist
* Mixi

# 三、MongoDB 教程



MongoDB 是一个基于分布式文件存储的数据库。由 C++ 语言编写。旨在为 WEB 应用提供可扩展的高性能数据存储解决方案。

MongoDB 是一个介于关系数据库和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的。

## 什么是MongoDB ?

MongoDB 是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。

在高负载的情况下，添加更多的节点，可以保证服务器性能。

MongoDB 旨在为WEB应用提供可扩展的高性能数据存储解决方案。

MongoDB 将数据存储为一个文档，数据结构由键值(key=>value)对组成。MongoDB 文档类似于 JSON 对象。字段值可以包含其他文档，数组及文档数组。



## 主要特点

* MongoDB 是一个面向文档存储的数据库，操作起来比较简单和容易。
* 你可以在MongoDB记录中设置任何属性的索引 (如：FirstName="Sameer",Address="8 Gandhi Road")来实现更快的排序。
* 你可以通过本地或者网络创建数据镜像，这使得MongoDB有更强的扩展性。
* 如果负载的增加（需要更多的存储空间和更强的处理能力） ，它可以分布在计算机网络中的其他节点上这就是所谓的分片。
* Mongo支持丰富的查询表达式。查询指令使用JSON形式的标记，可轻易查询文档中内嵌的对象及数组。
* MongoDb 使用update()命令可以实现替换完成的文档（数据）或者一些指定的数据字段 。
* Mongodb中的Map/reduce主要是用来对数据进行批量处理和聚合操作。
* Map和Reduce。Map函数调用emit(key,value)遍历集合中所有的记录，将key与value传给Reduce函数进行处理。
* Map函数和Reduce函数是使用Javascript编写的，并可以通过db.runCommand或mapreduce命令来执行MapReduce操作。
* GridFS是MongoDB中的一个内置功能，可以用于存放大量小文件。
* MongoDB允许在服务端执行脚本，可以用Javascript编写某个函数，直接在服务端执行，也可以把函数的定义存储在服务端，下次直接调用即可。
* MongoDB支持各种编程语言:RUBY，PYTHON，JAVA，C++，PHP，C#等多种语言。
* MongoDB安装简单。

## 历史

* 2007年10月，MongoDB由10gen团队所发展。2009年2月首度推出。
* 2012年05月23日，MongoDB2.1 开发分支发布了! 该版本采用全新架构，包含诸多增强。
* 2012年06月06日，MongoDB 2.0.6 发布，分布式文档数据库。
* 2013年04月23日，MongoDB 2.4.3 发布，此版本包括了一些性能优化，功能增强以及bug修复。
* 2013年08月20日，MongoDB 2.4.6 发布。
* 2013年11月01日，MongoDB 2.4.8 发布。
* ……

## MongoDB 下载

你可以在mongodb官网下载该安装包，地址为：<https://www.mongodb.com/download-center#community>。MonggoDB支持以下平台:

* OS X 32-bit
* OS X 64-bit
* Linux 32-bit
* Linux 64-bit
* Windows 32-bit
* Windows 64-bit
* Solaris i86pc
* Solaris 64

## 语言支持

MongoDB有官方的驱动如下：

* [C](http://github.com/mongodb/mongo-c-driver)
* [C++](http://github.com/mongodb/mongo)
* [C# / .NET](http://www.mongodb.org/display/DOCS/CSharp+Language+Center)
* [Erlang](https://github.com/TonyGen/mongodb-erlang)
* [Haskell](http://hackage.haskell.org/package/mongoDB)
* [Java](http://github.com/mongodb/mongo-java-driver)
* [JavaScript](http://www.mongodb.org/display/DOCS/Javascript+Language+Center)
* [Lisp](https://github.com/fons/cl-mongo)
* [node.JS](http://github.com/mongodb/node-mongodb-native)
* [Perl](http://github.com/mongodb/mongo-perl-driver)
* [PHP](http://github.com/mongodb/mongo-php-driver)
* [Python](http://github.com/mongodb/mongo-python-driver)
* [Ruby](http://github.com/mongodb/mongo-ruby-driver)
* [Scala](https://github.com/mongodb/casbah)

## MongoDB 工具

有几种可用于MongoDB的管理工具。

### 监控

MongoDB提供了网络和系统监控工具Munin，它作为一个插件应用于MongoDB中。

Gangila是MongoDB高性能的系统监视的工具，它作为一个插件应用于MongoDB中。

基于图形界面的开源工具 Cacti, 用于查看CPU负载, 网络带宽利用率,它也提供了一个应用于监控 MongoDB 的插件。

### GUI

* Fang of Mongo – 网页式,由Django和jQuery所构成。
* Futon4Mongo – 一个CouchDB Futon web的mongodb山寨版。
* Mongo3 – Ruby写成。
* MongoHub – 适用于OSX的应用程序。
* Opricot – 一个基于浏览器的MongoDB控制台, 由PHP撰写而成。
* Database Master — Windows的mongodb管理工具
* RockMongo — 最好的PHP语言的MongoDB管理工具，轻量级, 支持多国语言.

## MongoDB 应用案例

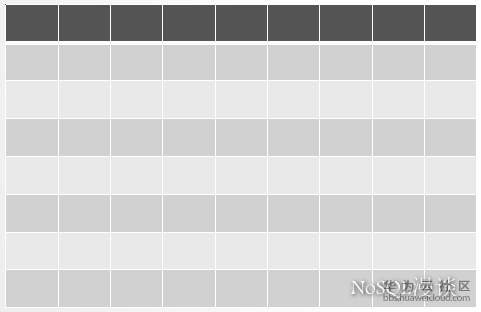
下面列举一些公司MongoDB的实际应用：

* Craiglist上使用MongoDB的存档数十亿条记录。
* FourSquare，基于位置的社交网站，在Amazon EC2的服务器上使用MongoDB分享数据。
* Shutterfly，以互联网为基础的社会和个人出版服务，使用MongoDB的各种持久性数据存储的要求。
* bit.ly, 一个基于Web的网址缩短服务，使用MongoDB的存储自己的数据。
* spike.com，一个MTV网络的联营公司， spike.com使用MongoDB的。
* Intuit公司，一个为小企业和个人的软件和服务提供商，为小型企业使用MongoDB的跟踪用户的数据。
* sourceforge.net，资源网站查找，创建和发布开源软件免费，使用MongoDB的后端存储。
* etsy.com ，一个购买和出售手工制作物品网站，使用MongoDB。
* 纽约时报，领先的在线新闻门户网站之一，使用MongoDB。
* CERN，著名的粒子物理研究所，欧洲核子研究中心大型强子对撞机的数据使用MongoDB。

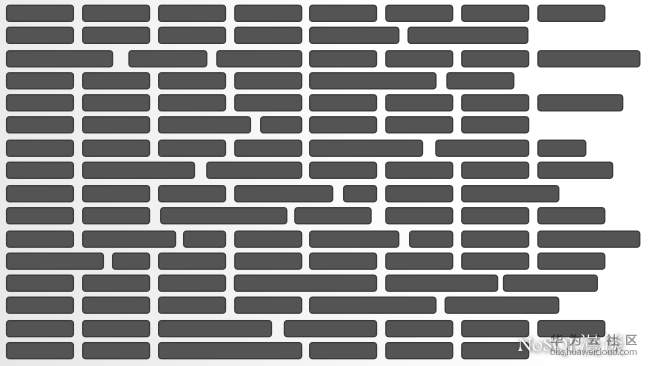
# 四、HBase

### 特点

用惯了Oracle/MySQL，心目中的数据表，应该是长成这样的：



这种表结构规整，每一行都有固定的列构成，因此，非常适合结构化数据的存储。但在NoSQL领域，数据表的模样却往往换成了另外一种"画风"：



行由看似"杂乱无章"的列组成，行与行之间也无须遵循一致的定义，而这种定义恰好符合半结构化数据或非结构化数据的特点。本文所要讲述的HBase，就属于该派系的一个典型代表。这些"杂乱无章"的列所构成的多行数据，被称之为一个"**稀疏矩阵**"，而上图中的每一个"黑块块"，在HBase中称之为一个KeyValue。

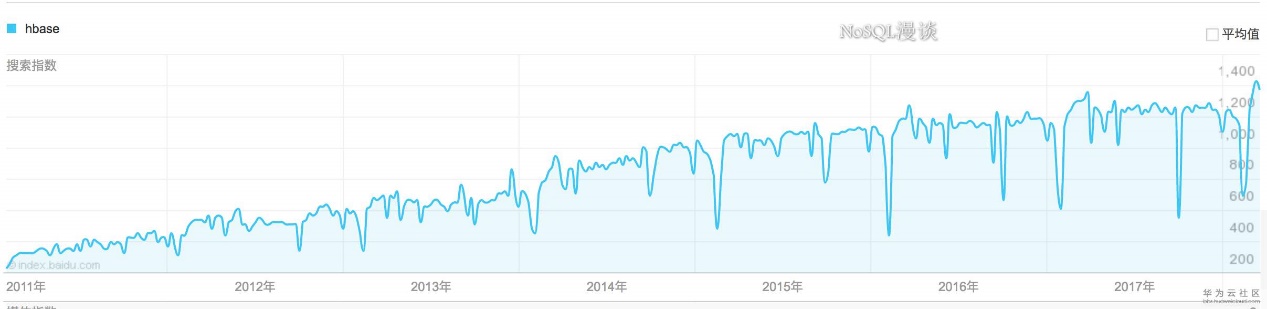
Apache HBase官方给出了这样的定义：

*Apache HBase™ is the Hadoop database, a distributed, scalable, big data store.*

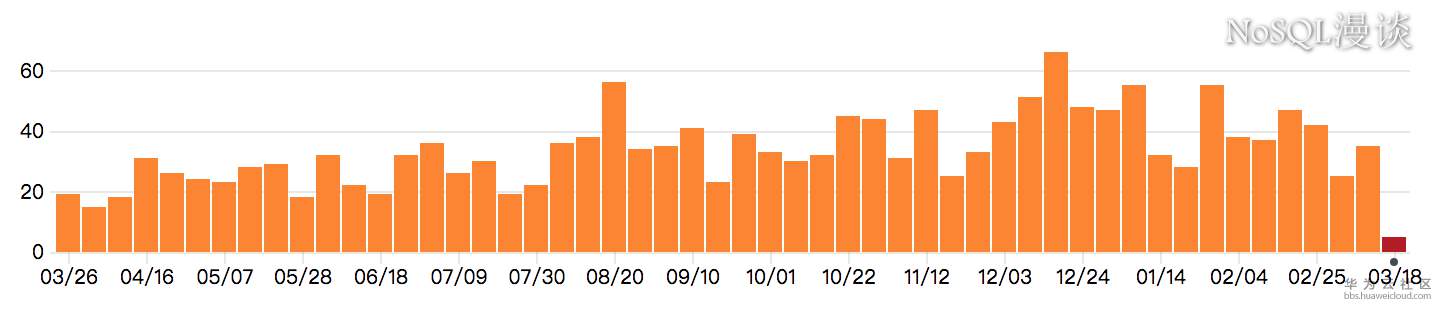
即：Apache HBase是基于Hadoop构建的一个**分布式**的、**可伸缩**的海量数据存储系统。

HBase常被用来存放一些结构简单，但数据量非常大的数据(通常在TB级别以上)，如历史订单记录，日志数据，监控Metris数据等等，HBase提供了简单的基于Key值的快速查询能力。

HBase在国内市场已经取得了非常广泛的应用，在搜索引擎中，也可以看出来，HBase在国内呈现出了逐年上升的势态：



从Apache HBase所关联的github项目的commits统计信息来看，也可以看出来该项目非常活跃：



HBase擅长于存储结构简单的海量数据但索引能力有限，而Oracle等传统关系型数据库(RDBMS)能够提供丰富的查询能力，但却疲于应对TB级别的海量数据存储，**HBase对传统的RDBMS并不是取代关系，而是一种补充**。

### 数据模型

**RowKey**

用来表示唯一一行记录的**主键**，HBase的数据是按照RowKey的**字典顺序**进行全局排序的，所有的查询都只能依赖于这一个排序维度。

通过下面一个例子来说明一下"**字典排序**"的原理：

RowKey列表{"abc", "a", "bdf", "cdf", "defg"}按字典排序后的结果为{"a", "abc", "bdf", "cdf", "defg"}

也就是说，当两个RowKey进行排序时，先对比两个RowKey的第一个字节，如果相同，则对比第二个字节，依此类推...如果在对比到第M个字节时，已经超出了其中一个RowKey的字节长度，那么，短的RowKey要被排在另外一个RowKey的前面。

**稀疏矩阵**

参考了Bigtable，HBase中一个表的数据是按照稀疏矩阵的方式组织的，"开篇"部分给出了一张关于HBase数据表的抽象图，我们再结合下表来加深大家关于"稀疏矩阵"的印象：



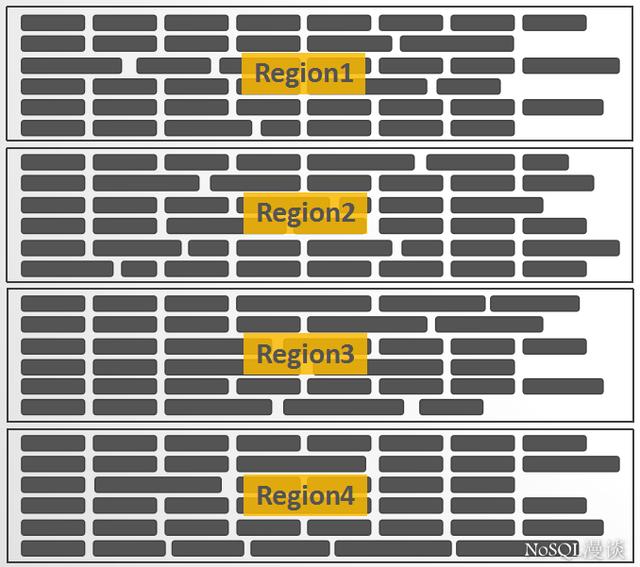
稀疏矩阵

看的出来：**每一行中，列的组成都是灵活的，行与行之间并不需要遵循相同的列定义**， 也就是HBase数据表"**schema-less**"的特点。

**Region**

区别于Cassandra/DynamoDB的"Hash分区"设计，HBase中采用了"Range分区"，将Key的完整区间切割成一个个的"Key Range" ，每一个"Key Range"称之为一个Region。

也可以这么理解：将HBase中拥有数亿行的一个大表，**横向切割**成一个个"**子表**"，这一个个"**子表**"就是**Region**：

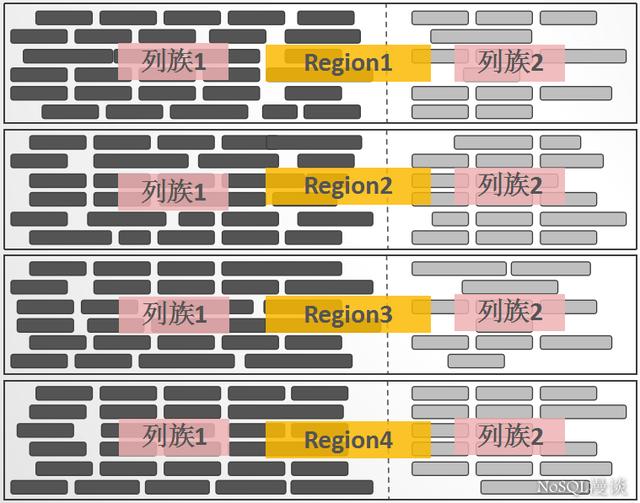


Region

**Region是HBase中负载均衡的基本单元**，当一个Region增长到一定大小以后，会自动分裂成两个。

**Column Family**

如果将Region看成是一个表的**横向切割**，那么，一个Region中的数据列的**纵向切割**，称之为一个**Column Family**。每一个列，都必须归属于一个Column Family，这个归属关系是在写数据时指定的，而不是建表时预先定义。



Column Family

**KeyValue**

KeyValue的设计不是源自Bigtable，而是要追溯至论文"The log-structured merge-tree(**LSM-Tree**)"。每一行中的每一列数据，都被包装成独立的拥有**特定结构**的KeyValue，KeyValue中包含了丰富的自我描述信息:

一条数据的HBase之旅，简明HBase入门教程-开篇

KeyValue

看的出来，KeyValue是支撑"稀疏矩阵"设计的一个关键点：一些Key相同的任意数量的独立KeyValue就可以构成一行数据。但这种设计带来的一个显而易见的缺点：**每一个KeyValue所携带的自我描述信息，会带来显著的数据膨胀**。

# 五、[Elasticsearch](https://www.elastic.co/" \t "_blank)

### 简介

[全文搜索](https://baike.baidu.com/item/%E5%85%A8%E6%96%87%E6%90%9C%E7%B4%A2%E5%BC%95%E6%93%8E)属于最常见的需求，开源的 [Elasticsearch](https://www.elastic.co/) （以下简称 Elastic）是目前全文搜索引擎的首选。

它可以快速地储存、搜索和分析海量数据。维基百科、Stack Overflow、Github 都采用它。



Elastic 的底层是开源库 [Lucene](https://lucene.apache.org/)。但是，你没法直接用 Lucene，必须自己写代码去调用它的接口。Elastic 是 Lucene 的封装，提供了 REST API 的操作接口，开箱即用。

### 深入简介

Elasticsearch是一个实时的分布式搜索和分析引擎。它可以帮助你用前所未有的速度去处理大规模数据。

它可以用于全文搜索，结构化搜索以及分析，当然你也可以将这三者进行组合。

Elasticsearch是一个建立在全文搜索引擎 Apache Lucene™ 基础上的搜索引擎，可以说Lucene是当今最先进，最高效的全功能开源搜索引擎框架。

但是Lucene只是一个框架，要充分利用它的功能，需要使用JAVA，并且在程序中集成Lucene。需要很多的学习了解，才能明白它是如何运行的，Lucene确实非常复杂。

Elasticsearch使用Lucene作为内部引擎，但是在使用它做全文搜索时，只需要使用统一开发好的API即可，而不需要了解其背后复杂的Lucene的运行原理。

当然Elasticsearch并不仅仅是Lucene这么简单，它不但包括了全文搜索功能，还可以进行以下工作:

1）分布式实时文件存储，并将每一个字段都编入索引，使其可以被搜索。

2）实时分析的分布式搜索引擎。

3）可以扩展到上百台服务器，处理PB级别的结构化或非结构化数据。

4）这么多的功能被集成到一台服务器上，你可以轻松地通过客户端或者任何你喜欢的程序语言与ES的RESTful API进行交流。

5）Elasticsearch的上手是非常简单的。它附带了很多非常合理的默认值，这让初学者很好地避免一上手就要面对复杂的理论，

6）它安装好了就可以使用了，用很小的学习成本就可以变得很有生产力。

7）随着越学越深入，还可以利用Elasticsearch更多高级的功能，整个引擎可以很灵活地进行配置。可以根据自身需求来定制属于自己的Elasticsearch。

**使用案例：**

维基百科使用Elasticsearch来进行全文搜做并高亮显示关键词，以及提供search-as-you-type、did-you-mean等搜索建议功能。

英国卫报使用Elasticsearch来处理访客日志，以便能将公众对不同文章的反应实时地反馈给各位编辑。

StackOverflow将全文搜索与地理位置和相关信息进行结合，以提供more-like-this相关问题的展现。

GitHub使用Elasticsearch来检索超过1300亿行代码。

每天，Goldman Sachs使用它来处理5TB数据的索引，还有很多投行使用它来分析股票市场的变动。

但是Elasticsearch并不只是面向大型企业的，它还帮助了很多类似DataDog以及Klout的创业公司进行了功能的扩展

### 优缺点：

优点：

Elasticsearch是分布式的。不需要其他组件，分发是实时的，被叫做”Push replication”。

Elasticsearch 完全支持 Apache Lucene 的接近实时的搜索。

处理多租户（multitenancy）不需要特殊配置，而Solr则需要更多的高级设置。

Elasticsearch 采用 Gateway 的概念，使得完备份更加简单。

各节点组成对等的网络结构，某些节点出现故障时会自动分配其他节点代替其进行工作。

缺点：

只有一名开发者（当前Elasticsearch GitHub组织已经不只如此，已经有了相当活跃的维护者）；

还不够自动（不适合当前新的Index Warmup API）

# 六、solr

### 简介

Solr（读作“solar”）是Apache Lucene项目的开源企业搜索平台。其主要功能包括全文检索、命中标示、分面搜索、动态聚类、数据库集成，以及富文本（如Word、PDF）的处理。Solr是高度可扩展的，并提供了分布式搜索和索引复制。Solr是最流行的企业级搜索引擎，Solr4 还增加了NoSQL支持。

Solr是用Java编写、运行在Servlet容器（如 Apache Tomcat 或Jetty）的一个独立的全文搜索服务器。 Solr采用了 Lucene Java 搜索库为核心的全文索引和搜索，并具有类似REST的HTTP/XML和JSON的API。Solr强大的外部配置功能使得无需进行Java编码，便可对 其进行调整以适应多种类型的应用程序。Solr有一个插件架构，以支持更多的高级定制。

因为2010年 Apache Lucene 和 Apache Solr 项目合并，两个项目是由同一个Apache软件基金会开发团队制作实现的。提到技术或产品时，Lucene/Solr或Solr/Lucene是一样的。

### Solr的优缺点

优点：

Solr有一个更大、更成熟的用户、开发和贡献者社区。

支持添加多种格式的索引，如：HTML、PDF、微软 Office 系列软件格式以及 JSON、XML、CSV 等纯文本格式。

Solr比较成熟、稳定。

不考虑建索引的同时进行搜索，速度更快。

缺点：

建立索引时，搜索效率下降，实时索引搜索效率不高。

### Elasticsearch与Solr的比较\*

当单纯的对已有数据进行搜索时，Solr更快。

当实时建立索引时, Solr会产生io阻塞，查询性能较差, Elasticsearch具有明显的优势。

随着数据量的增加，Solr的搜索效率会变得更低，而Elasticsearch却没有明显的变化。

综上所述，Solr的架构不适合实时搜索的应用。

### Elasticsearch 与 Solr 的比较总结

1）二者安装都比较简单；

2）Solr 利用 Zookeeper 进行分布式管理，而 Elasticsearch 自身带有分布式协调管理功能;

3）Solr 支持更多格式的数据，而 Elasticsearch 仅支持json文件格式；

4）Solr 官方提供的功能更多，而 Elasticsearch 本身更注重于核心功能，高级功能多有第三方插件提供；

5）Solr 在传统的搜索应用中表现好于 Elasticsearch，但在处理实时搜索应用时效率明显低于 Elasticsearch。

**Solr 是传统搜索应用的有力解决方案，但 Elasticsearch 更适用于新兴的实时搜索应用。**

# 七、neo4j

# 八、hive

# 九、pig