

# 不周山

一个算法攻城师的记忆: 关于生活、信息自由、数据挖掘、高性能计算

ABOUT 关于

登录

Categories: [Algorithm](#) [Database](#) [Life & Thinking](#) [Linux & Service](#) [Programming](#) [Semantic & NLP](#) [Tips](#) [未分类](#)

## mongodb小结

用了一阵子mongodb，作一些小结，作为将来的参考。按照以往的习惯，先作一个总览，然后再挑出一些自己比较关注的几个点，作为珠玑，加以串联阐述。

mongodb由C++写就，其名字来自humongous这个单词的中间部分，从名字可见其野心所在就是海量数据的处理。关于它的一个最简洁描述为：**scalable, high-performance, open source, schema-free, document-oriented database**。我对于文档型数据库有一些个人的偏好，这种偏好是从半年前研究couchdb而来的，因为我觉得用它来描述一个具有个性化特征的实体对象正合适，比如网站上的用户或商品书籍之类的条目。

一些概念：

跟mysqld一样，一个mongod服务可以有建立多个数据库，每个数据库可以有多张表，这里的表名叫collection，每个collection可以存放多个文档（document），每个文档都以BSON（binary json）的形式存放于硬盘中。跟关系型数据库不一样的地方是，它是的以单文档为单位存储的，你可以任意给一个或一批文档新增或删除字段，而不会对其它文档造成影响，这就是所谓的**schema-free**，这也是文档型数据库最主要的优点。跟一般的**key-value**数据库不一样的是，它的**value**中存储了结构信息，所以你又又可以像关系型数据库那样对某些域进行读写、统计等操作。可以说是兼备了**key-value**数据库的方便高效与关系型数据库的强大功能。

索引

跟关系型数据库类似，mongodb可以对某个字段建立索引，可以建立组合索引、唯一索引，也可以删除索引。当然建立索引就意味着增加空间开销，我的建议是，如果你能把一个文档作为一个对象的来考虑，在线上应用中，你通常只要对对象ID建立一个索引即可，根据ID取出对象某些数据放在memcache即可。如果是后台的分析需要，响应要求不高，查询非索引的字段即便直接扫表也费不了太多时间。如果还受不了，就再建一个索引得了。

默认情况下每个表都会有一个唯一索引：\_id，如果插入数据时没有指定\_id，服务会自动生成一个\_id，为了充分利用已有索引，减少空间开销，最好是自己指定一个unique的key为\_id，通常用对象的ID比较合适，比如商品的ID。

capped collection

capped collection是一种特殊的表，它的建表命令为：

```
db.createCollection("mycoll", {capped:true, size:100000})
```

允许在建表之初就指定一定的空间大小，接下来的插入操作会不断地按顺序APPEND数据在这个预分配好空间的文件中，如果已经超出空间大小，则回到文件头覆盖原来的数据继续插入。这种结构保证了插入和查询的高效性，它不允许删除单个记录，更新的也有限制：不能超过原有记录的大小。这种表效率很高，它适用于一些暂时保存数据的场合，比如网站上登录用户的session信息，又比如一些程序的监控日志，都是属于过了一定的时间就可以被覆盖的数据。

复制与分片

mongodb的复制架构跟mysql也很类似，除了包括master-slave构型和master-master构型之外，还有一个Replica pairs构型，这种构型在平常可以像master-slave那样工作，一但master出现问题，应用会自动了连接slave。要做复制也很简单，我自己使用过master-slave构型，只要在某一个服务启动时加上--master参数，而另一个服务加上--slave与--source参数，即可实现同步。

分片是个很头疼的问题，数据量大了肯定要分片，mysql下的分片正是成为无数DBA的噩梦。在mongodb下，文档数据库类似key-value数据库那样的易分布特性就显现出来了，无论构造分片服务，新增节点还是删除节点都非常容易实现。但mongodb在这方面做还不足够成熟，现在分片的工作还只做到alpha2版本（mongodb v1.1），估计还有很多问题要解决，所以只能期待，就

SUBSCRIBE



标签云

BI C couchdb Database  
google infobright  
interface lighttpd mongodb  
mysql package  
performance  
Python R semantic-  
web service tagging  
translation ubuntu vim  
webpy 互联网 交流 劝世 商  
务智能 并行 推荐系  
统 数据仓库 数据挖掘 机  
器学习 架构 模式识别 游记 源  
码 电影 神话 文摘 计算平台 阅读

文章索引

- > 2011 年八月 (1)
- > 2011 年七月 (2)
- > 2011 年六月 (4)
- > 2011 年五月 (1)
- > 2011 年四月 (2)
- > 2011 年三月 (2)
- > 2011 年二月 (4)
- > 2011 年一月 (4)
- > 2010 年十月 (1)
- > 2010 年九月 (1)
- > 2010 年八月 (2)
- > 2010 年七月 (3)
- > 2010 年六月 (2)
- > 2010 年五月 (2)
- > 2010 年四月 (5)
- > 2010 年三月 (4)
- > 2010 年二月 (2)
- > 2010 年一月 (3)
- > 2009 年十二月 (3)
- > 2009 年十一月 (6)
- > 2009 年十月 (2)
- > 2009 年九月 (5)
- > 2009 年八月 (15)
- > 2009 年七月 (6)
- > 2009 年六月 (1)
- > 2009 年四月 (1)
- > 2009 年三月 (4)
- > 2009 年二月 (1)
- > 2008 年十二月 (1)
- > 2008 年十一月 (7)
- > 2008 年十月 (3)
- > 2008 年八月 (1)
- > 2008 年三月 (1)
- > 2008 年二月 (2)
- > 2008 年一月 (3)
- > 2007 年十二月 (1)
- > 2007 年十一月 (2)
- > 2007 年八月 (5)
- > 2007 年七月 (7)
- > 2007 年六月 (8)
- > 2007 年五月 (15)
- > 2007 年四月 (1)

最近文章

- > 关于个性化的产品
- > 你喜欢什么编程语言？
- > 重返OpenParty
- > 自动化时代的机械工，记KDD2009的获胜者报告
- > 分类器评价、混淆矩阵与ROC曲线
- > 端午送礼，Package Rank计算源代码放出
- > Package Rank v1.1发布，及兵器谱分析
- > Package Rank，如果cran也可以有英雄榜
- > [无所不包的R]就地编译调试C/C++源代码的inline包
- > [无所不包的R]多核计算multicore package

近期评论

- > Nikang3148 在 R与C的接口：从R调用C的共享库 上的评论
- > 关于个性化的产品 | Planet Analyst 在 关于个性化的产品 上的评论
- > 关于个性化的产品 | Planet Analyst 在 开源推荐框架DUINE概览 上的评论
- > 重返OpenParty | Planet Analyst 在 重返OpenParty 上的评论
- > 自动化时代的机械工，

个人链接

- > 个性化推荐
- > 我的flickr
- > 我的twitter

不多说了。

性能

在我的使用场合下，千万级别的文档对象，近10G的数据，对有索引的ID的查询不会比mysql慢，而对非索引字段的查询，则是全面胜出。mysql实际无法胜任大数据量下任意字段的查询，而mongodb的查询性能实在让我惊讶。写入性能同样很令人满意，同样写入百万级别的数据，mongodb比我以前试用过的couchdb要快得多，基本10分钟以下可以解决。补上一句，观察过程中mongodb都远算不上是CPU杀手。

GridFS

gridfs是mongodb一个很有趣的类似文件系统的东西，它可以用一大块文件空间来存放大量的小文件，这个对于存储web2.0网站中常见的大量小文件（如大量的用户头像）特别有效。使用起来也很方便，基本上跟一般的文件系统类似。

用合适的数据库做适合的事情

mongodb的文档里提到的user case包括实时分析、logging、全文搜索，国内也有人使用mongodb来存储分析网站日志，但我认为mongodb用来处理有一定规模的网站日志其实并不合适，最主要的就是它占空间过于虚高，原来1G的日志数据它可以存成几个G，如此下去，一个硬盘也存不了几天的日志。另一方面，数据量大了肯定要考虑sharding，而mongodb的sharding到现在为止仍不太成熟。由于日志的不可更新性的，往往只需APPEND即可，又因为对日志的操作往往只集中于一两列，所以最合适作为日志分析的还是列存储型的数据库，特别是像infobright那样的为数据仓库而设计的列存储数据库。

由于mongodb不支持事务操作，所以事务要求严格的系统（如果银行系统）肯定不能用它。

mongodb占用空间过大的原因，在官方的FAQ中，提到有如下几个方面：

- 1、空间的预分配：为避免形成过多的硬盘碎片，mongodb每次空间不足时都会申请生成一大块的硬盘空间，而且申请的量从64M、128M、256M那样的指数递增，直到2G为单个文件的最大体积。随着数据量的增加，你可以在其数据目录里看到这些整块生成容量不断递增的文件。
- 2、字段名所占用的空间：为了保持每个记录内的结构信息用于查询，mongodb需要把每个字段的key-value都以BSON的形式存储，如果value域相对于key域并不大，比如存放数值型的数据，则数据的overhead是最大的。一种减少空间占用的方法是把字段名尽量取短一些，这样占用空间就小了，但这就要求在易读性与空间占用上作为权衡了。我曾建议作者把字段名作个index，每个字段名用一个字节表示，这样就不用担心字段名取多长了。但作者的担忧也不无道理，这种索引方式需要每次查询得到结果后把索引值跟原值作一个替换，再发送到客户端，这个替换也是挺耗费时间的。现在的实现算是拿空间来换取时间吧。
- 3、删除记录不释放空间：这很容易理解，为避免记录删除后的数据的大规模挪动，原记录空间不删除，只标记“已删除”即可，以后还可以重复利用。
- 4、可以定期运行db.repairDatabase()来整理记录，但这个过程会比较缓慢。

因为官方文档中对各方面的内容已经有很详细的叙述，所以我并没有再过多的引用原文与代码，只是结合自己的使用归纳一些心得，有兴趣的朋友不妨直接去翻文档中自己感兴趣的问题，超群的博客上有一个很好的入门介绍。

最后总结一句，文档型数据库有点像波粒二象性，总能在适当的时候表现出它作为关系型数据库或key-value数据库的优势来。

实战案例：

昨天我访问mongodb的python程序开始出错，经常抛出AssertionError异常，经查证只是master查询异常，slave正常，可判断为master的数据出了问题。

修复过程：

- 1、在master做db.repairDatabase()，不起作用；
- 2、停止slave的同步；
- 3、对slave作mongodump，备份数据；

记KDD2009的获胜者报告 | Planet Analyst 在 自动化时代的机械工，记KDD2009的获胜者报告 上的评论

› 我的豆瓣

朋友们

› Web设计师Tony

最近阅读最多

- › mongodb小结 - 30,662 views
- › R，不仅仅是一种语言 - 27,093 views
- › 2011推荐系统峰会及全民娱乐 - 23,336 views
- › 少数人的智慧(The Wisdom of the Few) - 11,153 views
- › 你喜欢什么编程语言？ - 7,134 views
- › 超越AB-Test，算法参数化与Google实验架构 - 6,661 views
- › 物理学背景的推荐算法与协同过滤 - 4,136 views
- › Youtube视频推荐算法：从10页论文到4页论文的变迁 - 4,029 views
- › follow人，还是follow内容 - 3,989 views
- › 开发大型机器学习系统的经验教训（来自googlesearch blog） - 3,972 views

业界同行

- › Beyond Search
- › COS发起人谢益辉
- › 向量 Vector 的空间

- 4、对master作mongostore，把备份数据恢复，使用-drop参数可以先把原表删除。
- 5、恢复slave的同步。

关于作者

阿稳, 豆瓣, 算法工程师

推荐系统; 数据挖掘; 算法架构及实现的可扩展性; R环境编程 如果你的问题已经能从我的博客中得到解答, 就最好不过了: <http://www.wentruen.net/blog/>



豆瓣

Twitter

blog

LinkedIn

SlideShare

新浪微博

向文章付费

请作者吃饭

14

喜欢

推荐

+

您可能也喜欢:



[阅读] MySQL性能调优与架构设计

BejingOpenParty 溪窗听雨见闻2009-08-29



[无所不包的R]就地编译调试C/C++源代码的inline包

[无所不包的R]多核计算multicore package

收藏或分享到:



Posted in Database.


Tagged with mongodb.

By wentruen – 2010年03月16号


 View Comments

 顶

 1 person liked this.

 DISQUS

添加新的评论 [Login](#)



Please wait...

Showing 4 comments 

排序 受欢迎的



sanpei

它这玩意换汤不换药, 从技术角度看没什么意思, 有能力还不如自己实现 (不过多运行实例的同步是个费劲的事情。)

关系型数据库的问题是不暴露细节, 需要不需要的都整体给你, 所以效率自然低。虽然我们确实需要这种东西, 不过这些NoSQL却吹得自己如何如何, 这是最让人反感的。



Suchao\_2005

外在非关系型，内在关系型

✎ 月前

👍 回复



柏林

谢谢分享。mongodb的外在非关系型和内在的关系型很棒。mongodb的外在非关系型和内在的关系型很棒。

✎ 月前

👍 回复



Susanmail09

我可以分享这篇文章<http://www.energy-pv.com>

📅 月前

👍 回复

✉ [Subscribe by email](#)   [RSS](#)

Trackbacks

引用通告网址

✎ [mongodb概述 | mongodb介绍](#)  
04/26/2010 11:07 上午

[...] 转自<http://www.wenttrue.net/blog/?p=772> Posted in Post ...

✎ [分布式文件存储的数据库开源项目MongoDB - 自由、创新、研究、探索 - 博客园](#)  
05/20/2010 01:46 下午

[...] [mongodb小结](#) [...]

✎ [自明 » MongoDB 系统介绍](#)  
12/22/2010 12:22 下午

[...] 在 我的使用场合下，千万级别的文档对象，近10G的数据，对有索引的ID的查询不会比mysql慢，而对非索引字段的查询，则是全面胜出。mysql实际无法胜任大数据量下任意字段的查询，而mongodb的查询性能实在让我惊讶。写入性能同样很令人满意，同样写入百万级别的数据，mongodb比我以前试用过的couchdb要快得多，基本10分钟以下可以解决。补上一句，观察过程中mongodb都远算不上是CPU杀手。 Via:<http://www.wenttrue.net/blog/?p=772> [...]

blog comments powered by DISQUS

« [开源推荐框架DUINE概览](#)

[物理学背景的推荐算法与协同过滤](#) »