mongoDB

MongoDB简介与实践

夏远峰(拉斯)

新浪微博：http://www.weibo.com/xiayuanfeng
个人博客：http://xiayuanfeng.iteye.com
邮箱：xiayuanfeng@gmail.com

SACC2011

个人简介

- 迷路的程序员一枚
- 时间仓促，水平有限，望海涵

今日主题

- 关于NoSQL
- 正确认识NoSQL
- 介绍MongoDB
- Replication
- Auto-Sharding
- MongoDB技巧篇

NoSQL背景

- 数据高并发性的实时读取与写入
- 海量数据存储
- 海量数据的智能计算与挖掘
- 高性能，可扩展，可用性

正确认识NoSQL

- NoSQL是用来区别关系型数据库的数据存储
- NoSQL有灵活的Schema。
- 避免使用JOIN(连接)查询。
- 支持数据的水平扩展。
- 最终一致性
- 简单的访问接口(API)

NoSQL歧义

- 歧义一:NoSQL就是不使用SQL语句。
- 歧义二:所有非关系型数据库都是NoSQL
- 歧义三:完全放弃使用关系型数据库

NoSQL的意义

NoSQL=Not Only SQL.
这告诉我们数据存储的选型不要局限于关系型数据库。
而是一个混合关系型数据库和NoSQL数据库的复杂架构

主流NoSQL分类

- **面向文档的存储**
- **Key-value的存储**
- **列存储/列族(Column Store/Column Families)**

介绍MongoDB

- 面向文档(Document)存储
- C++实现
- 支持多平台 Windows Linux, Mac OS-X, FreeBSD, Solaris
- 多语言驱动:
  - Ruby / Ruby-on-Rails
  - Java
  - C#
  - JavaScript
  - C / C++
  - Erlang Python, Perl

MongoDB优势

- 不需要额外的缓存
- 丰富的查询
- 丰富的类型
- 内置replication和sharding，可扩展。
- 商业支持 10gen

MongoDB设计理念

- 不求搞定一切
- 正确的工具做正确的事儿
- 性能与功能的取舍
- 放弃事务

MongoDB特性

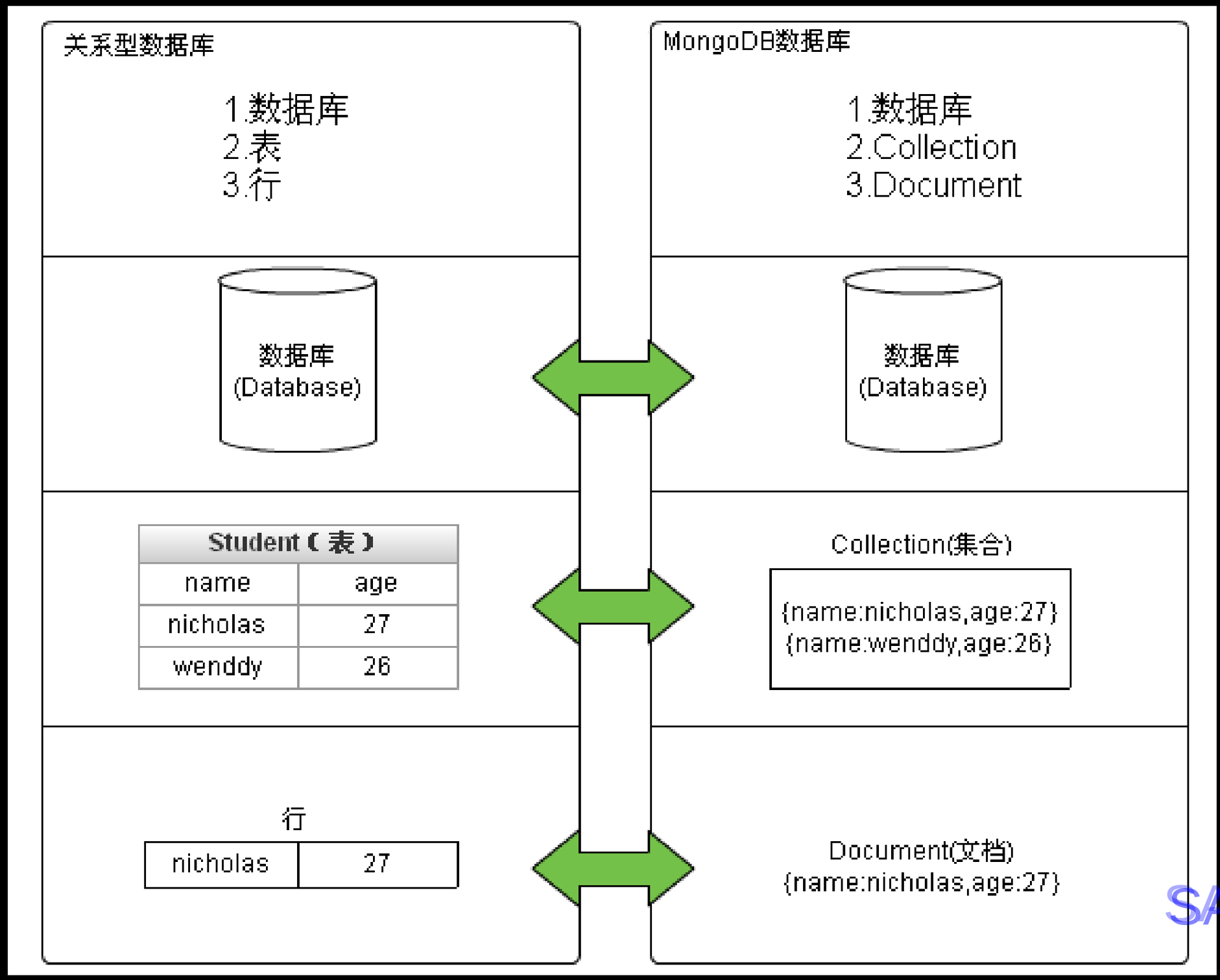
- 面向文档的存储(BSON)
- 支持动态查询
- 索引
- 调优工具
- 区别于关系型数据库的更新(in-place)
- Replication
- Auto-sharding
- MapReduce

MongoDB求助

- Google it
- Mail List
- Google Groups（你懂得）
- JIRA
- http://stackoverflow.com/
- 最后一步，问我可能也没用 交流为主
  xiayuanfeng@gmail.com

SACC2011

MongoDB数据模型



MongoDB术语

RDBMS	Mongo
Table, View	Collection
Row(s)	JSON Document
Index	Index
Join	Embedded Document
Partition	Shard
Partition Key	Shard Key

MongoDB Document

BSON（Like JSON 二进制）

```
{
   _id : ObjectId("4c4ba5c0672c685e5e8aabf3"),
   author : “nicholas",
   text : ”面向文档存储的数据库MongoDB",
   tags : [ ”数据库", ”MongoDB" ] }
}
```

MongoDB Collections

{“name”:”nicholas”,age:27}
{“name”:”lily”,age:29}
{“aaaa”:”ddddddddddddddddddadfadf”} ???!!! what
保存一个collection。

Schema-Free

- 不明确的需求。
- 对于开发人员和数据库管理员是个噩梦。
- 难以设定合理的索引。

MongoDB Database

- admin 全局数据库
- local 本地数据库

MongoDB 数据类型(1)

- JSON：null，布尔型(boolean)，数字类型(numeric)，字符串(string)，数组(array)，对象(object)

- BSON=JSON+
- **二进制数据(binary data)**
- **数组(array)**

{“comments”:[“哈哈”,”顶一下”]}

- **内嵌文档(embedded document)**

{“stuff”:{“name”:nicholas,age:27}}

MongoDB 数据类型(2)

- **Object Id**

当插入document的时候，MongoDB会自动设置一个_id做为key。

- **字符串(String)**

string类型是UTF-8编码的。

- **日期类型(date)**

MongoDB是以毫秒的形式来存储日期类型的。

{“createDate”:new Date()}

MongoDB 安装

1. 下载 http://www.mongodb.org/downloads
2. 解压缩
3. Linux /data/db or windows c:\data\db
4. 运行 mongod.exe

非常简单！

MongoDB 版本号

- MongoDB主要有三个A,B,C版本
- *A* 是主要版本. 功能上要么不变要么就做很大的改动。
- *B* 是发行版本号. 这种版本经常更新功能，包括一些新的特性并且常常不向后兼容。偶数是稳定版本，奇数是开发版本。
- *C* 版本号是用来修改BUG和安全性。

- 1.8.3

MongoDB Shell(1)

- Like mysql
- 操作语言Javascript
- 调用bin/mongo
- 标准的Javascript+，v8引擎需手动编译，性能OK。
- use 数据库名称
- show dbs;show collections
- *.help()

MongoDB Shell(2)

```
C:\Documents and Settings\Administrator>mongo
MongoDB shell version: 1.8.2
connecting to: test
>
> show dbs;
admin   (empty)
guupoo  0.453125GB
local   (empty)
test    0.203125GB
> use guupoo;
switched to db guupoo
> show collections;
gupiao
gupiao_day
jiangu
jiangu_notice
notice
people_jiangu_day
people_summary
system.indexes
>
```

增

- > var account={
- ... "email":"xiaxdx@163.com",
- ... "age":27,
- ... "blog_url":"http://blog.taobaokong.net",
- ... "date":new Date()
- ... }

- > db.account.insert(account);

查询

- db.account.find({“email”:”xiaxdx@163.com”})

- {
- "_id" : ObjectId("4cc2f17506eff28f57851c26"),
- "email" : "xiaxdx@163.com",
- "age" : 30,
- "blog_url" : "http://blog.taobaokong.net",
- "date" : "Sat Oct 23 2010 22:24:21 GMT+0800 (CST)"
- }

查询-$符号

```
db.collection.find({ "field" : { $gt: value } } );   // greater than  : field > value
db.collection.find({ "field" : { $lt: value } } );   // less than  :  field < value
db.collection.find({ "field" : { $gte: value } } );  // greater than or equal to : field >=
value
db.collection.find({ "field" : { $lte: value } } );  // less than or equal to : field <=
value
```

查询-$符号

- <, <=, >, >=
- $all
- $exists
- $mod
- $ne
- $in
- $nin
- $nor
- $or
- $and
- $size
- $type
- Regular Expressions
- Value in an Array
  - $elemMatch

更新

```
db.account.update({"email":"xiaxdx@163.com"},
{
  ‘$set’:{‘age’:31}
}
)
```

Modifier Operations

- $inc
- $set
- $unset
- $push
- $pushAll
- $addToSet
- $pop
- $pull
- $pullAll
- $rename
- $bit

删除

```
db.account.remove({"email":"xiaxdx@163.com"});
```

ObjectId

4c4ba5c0672c685e5e8aabf3 ??!!! What hell!!

4c4ba5c0 timestamp

672c68 machine id

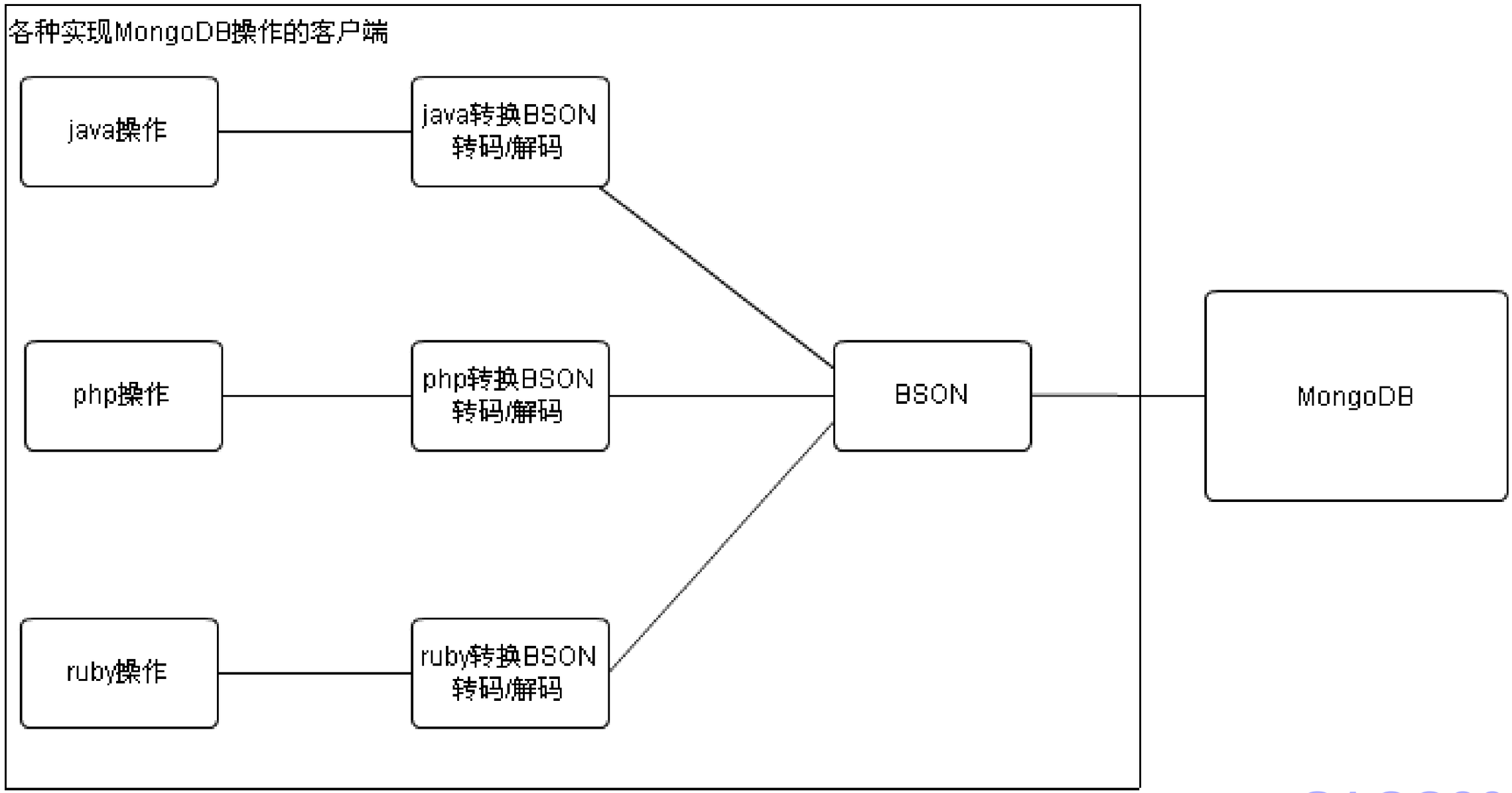
5e5e process id

8aabf3 counter

增量
为什么不用自增序列？？

SACC2011

BSON



索引(1)

- 可以参照MySQL的索引设置
- **db.things.ensureIndex({j:1});**
- **db.things.ensureIndex({"address.city": 1})**
- **Embedded document**

db.factories.insert({ name: "xyz", metro: { city: "New York", state: "NY" } });db.factories.ensureIndex({ metro : 1 });

- 数组 !索引了每个字段

索引(2)

- 组合索引

db.things.ensureIndex({j:1, name:-1});

- Covered Index
  - db.foo.ensureIndex({"x" : 1, "y" : 1, "z" : 1})
  - > db.foo.find({"x" : criteria, "y" : criteria},
    ... {"x" : 1, "y" : 1, "z" : 1, "_id" : 0})

不要总是使用索引

索引创建需要额外的空间

返回数据过多会得不偿失

MapReduce-入门

- { username: "jones", likes: 20, text: "Hello world!" }

- function()
- {
- emit(this.username, {count: 1, likes: this.likes});
- }

- function(key, values)
- {
- var result = {count: 0, likes: 0}; values.forEach(function(value) { result.count += value.count; result.likes += value.likes; }); return result; }

MapReduce-执行过程

- Map: key ->value 输出 的value为最终执行结果的聚合值

- Reduce:function reduce(key_obj, [value_obj, value_obj, ...]), 处理[value1,value2,value3], 可以循环处理, 返回的是map的第二个参数。递归调用

- function finalize(key, value) -> final_value
  –执行一次, 一般用来处理平均数
- 注意。Map的输出value和Reduce的返回value一致

MapReduce-案例


```
/* 77 */
{
  "_id" : ObjectId("4e437988e4b04b1e362b714c"),
  "className" : "com.guupoo.domain.JianGu",
  "jigou_name" : "招商证券",
  "jigou_people" : ["刘荣", "蔡宇滨"],
  "level" : "持有",
  "level_date" : new Date("Thu, 11 Aug 2011 00:00:00 GMT +08:00"),
  "level_shoupan" : 34.28,
  "now_shoupan" : 36.45,
  "now_shouyilv" : 0.06330221703617273 9,
  "stock_code" : "300159",
  "stock_hang" : "机械行业",
  "stock_name" : "新研股份",
  "url" : "http://biz.finance.sina.com.cn/qmx/stockreports.php?symbol=sz300159&report_id=1035304"
}
```


分析师	总收益率	总荐股概率	推荐股票上涨数	推荐股票下跌数
黄珺	99.27%	57.38%	35	25
胡乔	92.50%	56.67%	34	25
施亮	69.70%	63.16%	12	7
毛长青	69.70%	63.16%	12	7
包敦文	50.91%	37.50%	3	5
闫亚磊	49.88%	56.76%	21	16
许娟娟	48.23%	66.67%	22	11
曾光	42.20%	45.12%	37	33
郭敏	39.26%	57.14%	12	7
谈煊	29.04%	78.95%	15	3

MapReduce-Map

```
function Map() {
    for(var i=0;i<this.jigou_people.length;i++){
        var results={
            total_shouyilv:this.now_shouyilv,
            shouyilv:[this.now_shouyilv],
            jiangushu:1
        };
        emit(this.jigou_people[i],results);
    }
}
```

MapReduce-Reduce

```
1  function Reduce(key, arr_values) {
2      var results={total_shouyilv:0,shouyilv:[],jiangushu:0};
3      arr_values.forEach(function(value) {
4        results.total_shouyilv += value.total_shouyilv;
5        results.jiangushu+=value.jiangushu;
6        for(var i=0;i<value.shouyilv.length;i++){
7          results.shouyilv.push(value.shouyilv[i]);
8        }
9      });
10     return results;
11 }
```

MapReduce-Finalize

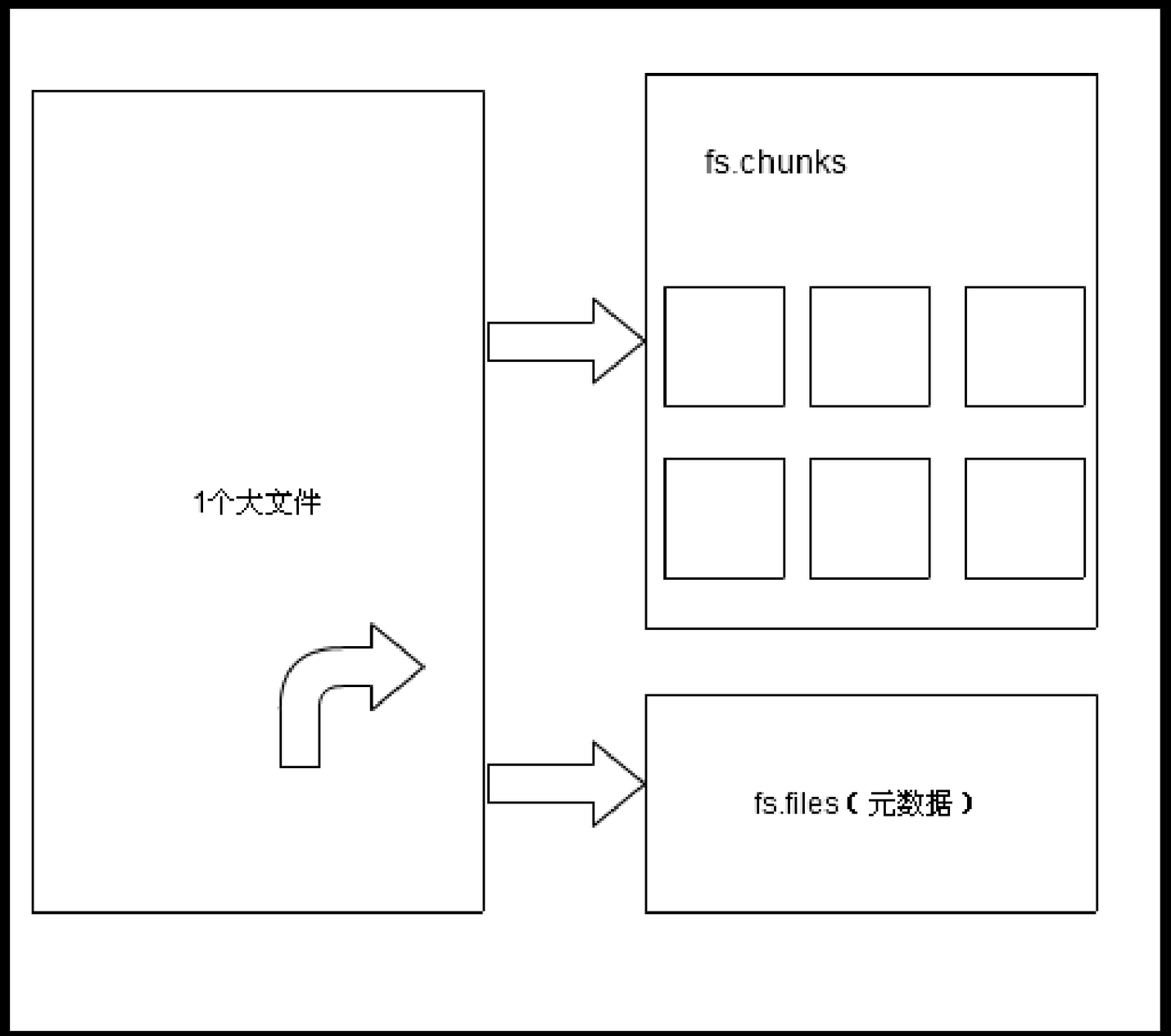
```
1  function Finalize(key, reduced) {
2      reduced.up=0;
3      reduced.down=0;
4      reduced.up_lv=0;
5      for(var i=0;i<reduced.shouyilv.length;i++){
6          if(reduced.shouyilv[i]>0){
7              reduced.up++;
8          }
9          if(reduced.shouyilv[i]<0){
10             reduced.down++;
11         }
12     }
13     reduced.up_lv=reduced.up/reduced.shouyilv.length;
14     delete reduced.shouyilv;
15     return reduced;
16 }
```

MapReduce-Result


```
/* 7 */
{
  "_id" : "Ning Ma",
  "value" : {
    "total_shouyilv" : -0.3807306639282840 3,
    "jiangushu" : 20.0,
    "up" : 5.0,
    "down" : 15.0,
    "up_lv" : 0.25
  }
}

/* 8 */
{
  "_id" : "Randy Zhou",
  "value" : {
    "total_shouyilv" : -1.116459731454275 7,
    "jiangushu" : 13.0,
    "up" : 1.0,
    "down" : 12.0,
    "up_lv" : 0.07692307692307692 7
  }
}
```

GridFS(1)



GridFS(2)

```
C:\Documents and Settings\Administrator>mongofiles put c:\version.conf
connected to: 127.0.0.1
added file: { _id: ObjectId('4e68a62f98c9c8f3183e4f4a'), filename: "c:\version.
onf", chunkSize: 262144, uploadDate: new Date(1315481135062), md5: "728db4613c3
aad773f564da53d1fd73", length: 423 }
done!

C:\Documents and Settings\Administrator>mongofiles list
connected to: 127.0.0.1
c:\version.conf 423

C:\Documents and Settings\Administrator>mongofiles search version
connected to: 127.0.0.1
c:\version.conf 423

C:\Documents and Settings\Administrator>mongofiles search o
connected to: 127.0.0.1
c:\version.conf 423

C:\Documents and Settings\Administrator>_
```

GridFS(3)

- **不要使用GridFS来保存小的二进制数据**
- GridFS需要两次查询:一次取出文件的元数据，另一次取出内容
- 允许相同的文件名存在。
- MD5校验文件完整性

Replication目标

- 可伸缩性
  - –性能
  - –冗余
- 高可用
- 隔离性
  - –运行备份或者独立的后台复杂任务
  - –开发时，进行在线业务测试

MongoDB Replication

- Master-Slave (不推荐)
  - Master mongod -- master
  - Slave mongod -- source <master url>:<master port>
  - -- slavedelay <seconds> 备份

- Replica Sets(就叫副本集 WOW！)
  - 故障自动切换
  - 强烈推荐

  副本集是M/S超集

Oplog

- 主服务器记录变化的日志。
- 从发现变化进行复制
- Oplog timestamp 从服务器通过最近读取，来同步。
- Oplog大小可设置。

Replica Sets

- Master->Primary（就一个）
- Slave->Secondary（多个）
- 主人/奴隶-》主要/次要
- 自动故障切换

Replica Sets

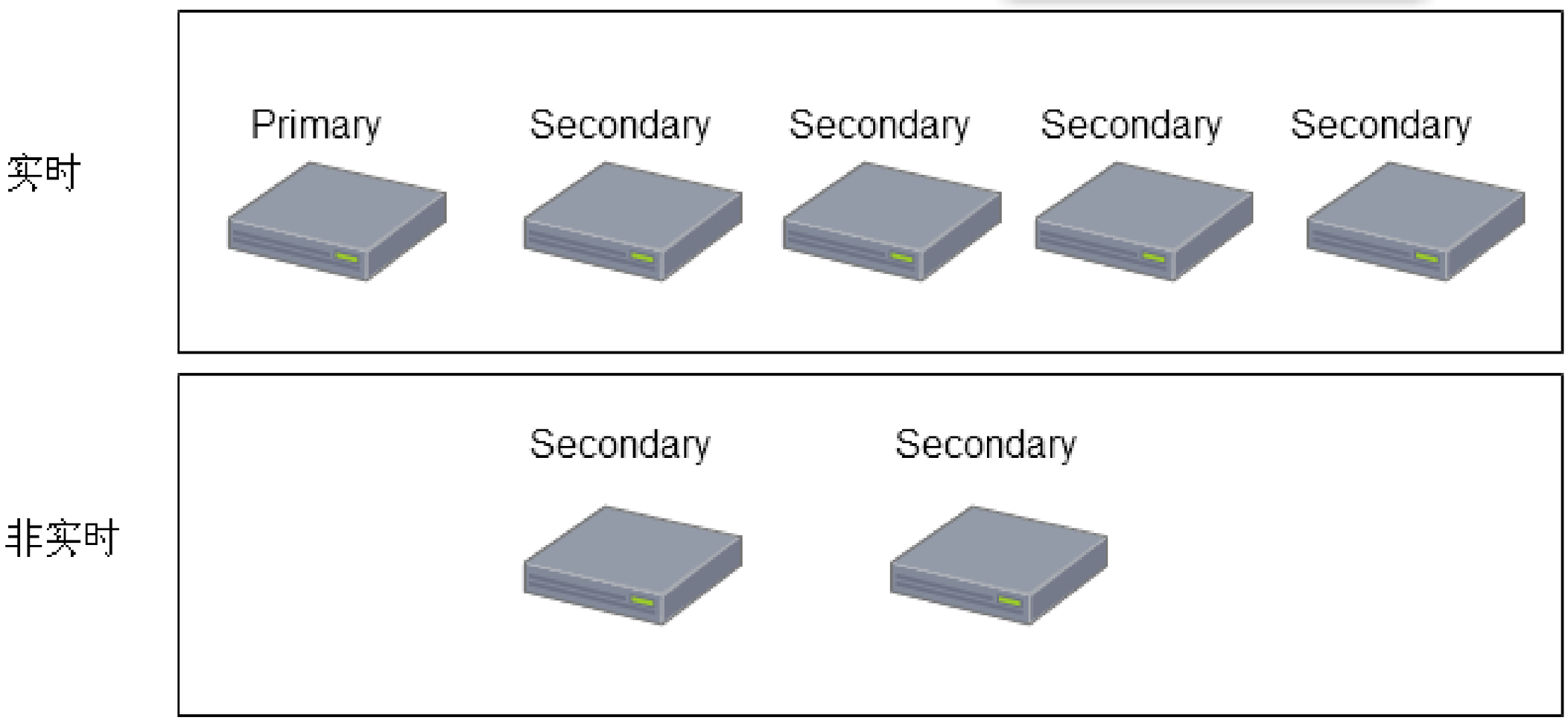
- mongod --replSet foo --port 27017 --dbpath /data/db1
  mongod --replSet foo --port 27018 --dbpath /data/db2
  mongod --replSet foo --port 27019 --dbpath /data/db3

- mongo localhost:27017
  - > rs.initiate()
  - > rs.add("localhost:27018")
  - > rs.add("localhost:27019")

Replica Sets

- rs.status()
- rs.stepDown()
- db.isMaster()

Replica Sets



大规模数据处理

- 内存与硬盘
  内存比硬盘要快10W-100W
  总线速度大概100倍
  内存中完成任务，好的数据结构预算法。

- 两个指标:CPU 和I/O

```
top - 05:38:12 up 16 days, 11:49,  1 user,  load average: 0.67, 0.55, 0.45
%iowait

    0.01

    0.01

    0.01

    0.01

    0.01
```

关于Linux页面缓存那点事儿

1. 避免swap交换

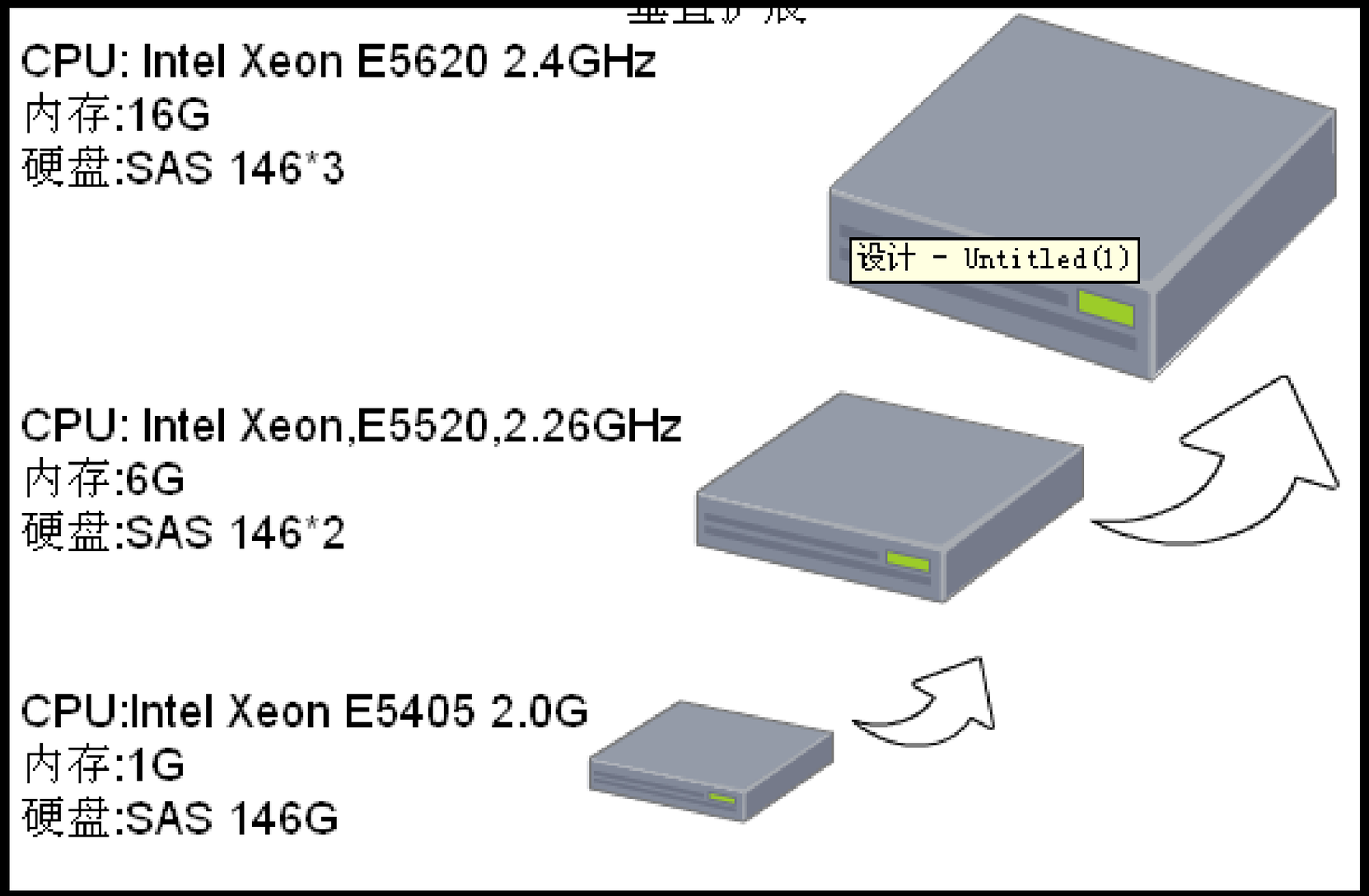
2. 物理内存不足，使用硬盘交换

3. LRU

```
03:50:01 AM   pswpin/s pswpout/s
04:00:01 AM      0.00         0.00
04:10:02 AM      0.00         0.00
04:20:01 AM      0.00         0.00
04:30:01 AM      0.00         0.00
04:40:01 AM      0.00         0.00
04:50:01 AM      0.00         0.00
05:00:01 AM      0.00         0.00
05:10:01 AM      0.00         0.00
```

降低I/O

- 垂直扩展-此方法很暴力:增加内存
- 横向扩展



可扩展性

- 计算密集型

难度低:没有离散数据，应用服务器。

- I/O密集型

当前主要瓶颈，数据关联严重，很难分散。

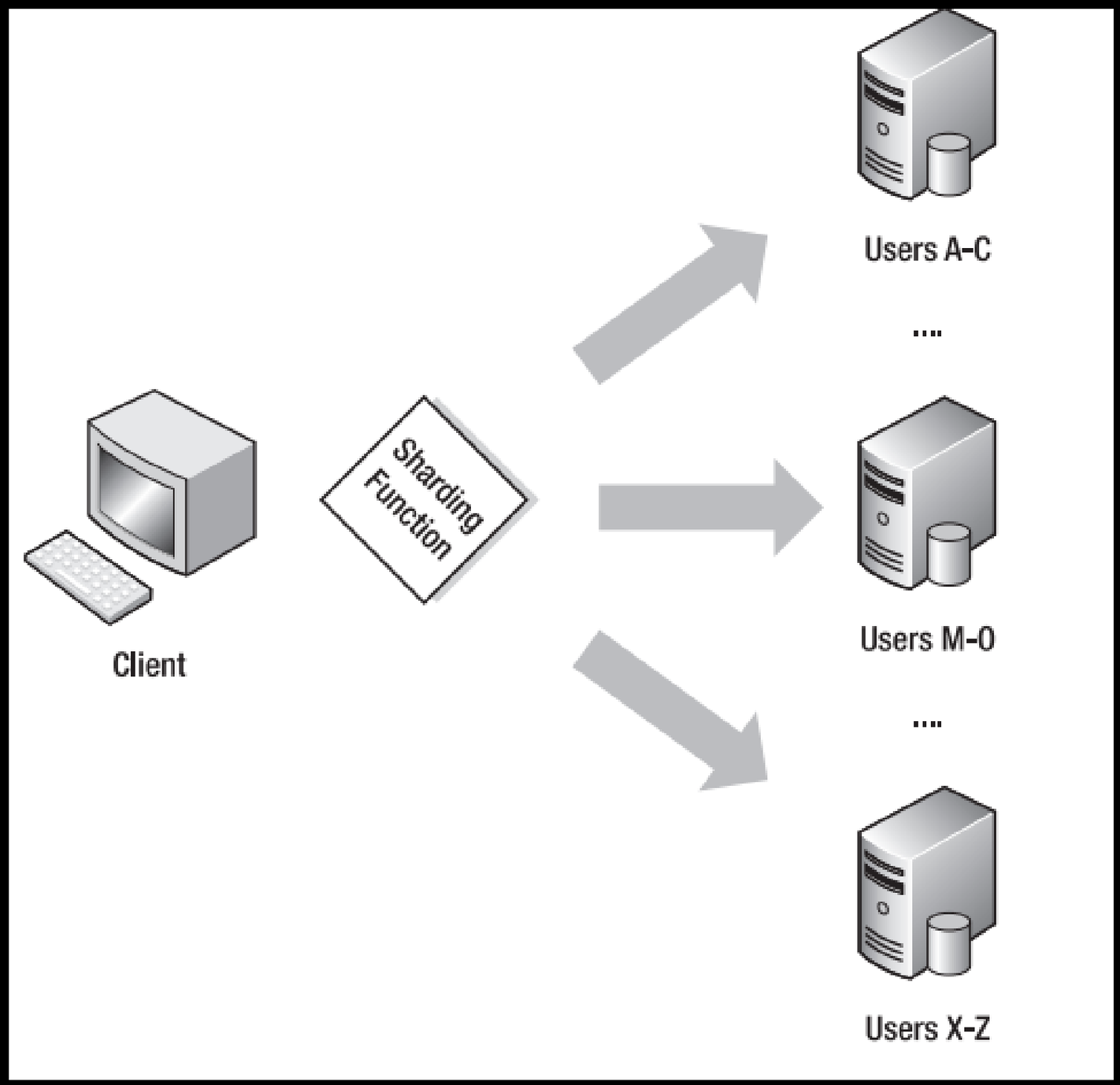
Replication??

- Replication无法解决内存中处理的问题

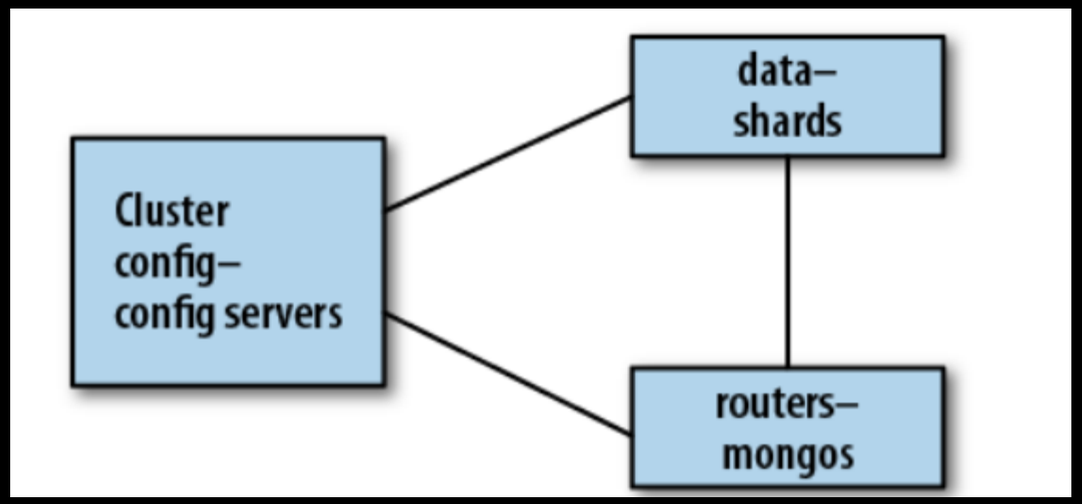
Sharding目标

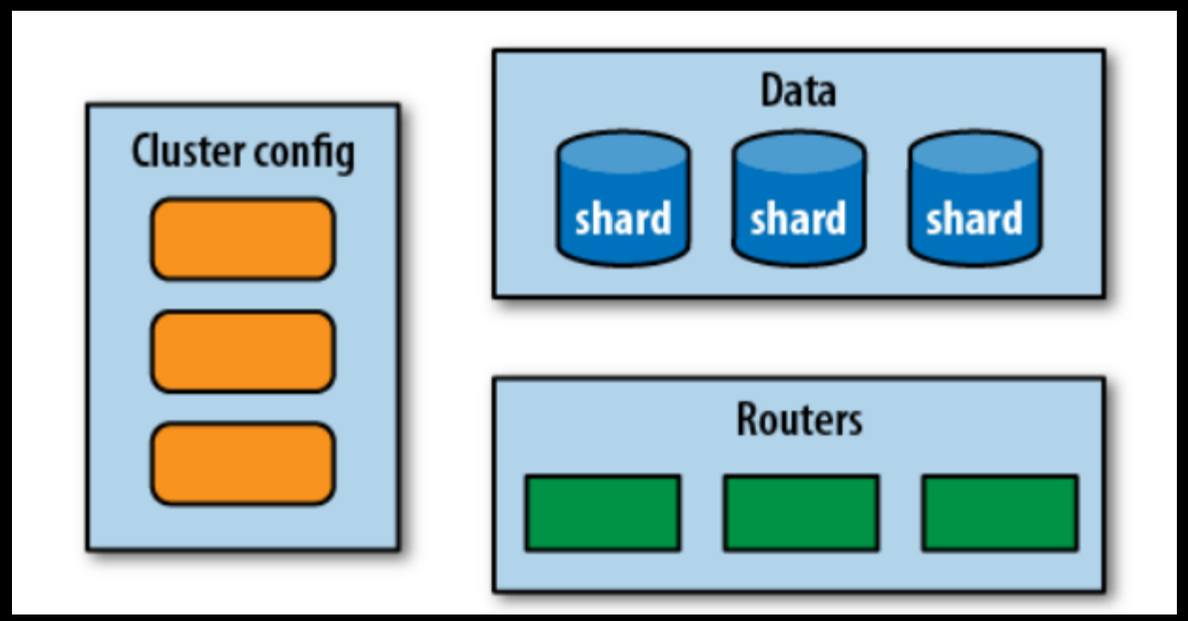
- MongoDB把一个庞大数据量的collection拆分到很多服务器上
- 让集群不可见
  - 让应用访问集群就如访问单独的Mongod那样简单。为了完成这个目标，MongoDB就有一个特殊的路由过程叫做mongos
- 让集群一直保持读写正常
  - Replica set
- 很简单的进行集群的扩展
  - 当系统需要更多的硬盘空间与资源的时候，就可以添加服务器了。MongoDB可以再你需要的时候便捷的来进行集群的扩展

Sharding实现



Sharding架构





关于Shard Key

- 固定值不适合做shard key，chunk数固定
- 增式的shard key，数据始终在一个新增的chunk，造成热点
- 一个优秀的shard key
  - 升序的字段+可搜索的字段

Sharding-Config

- Config Server
  - –$ ssh ny-01
    ny-01$ mongod
    $ ssh sf-01
    sf-01$ mongod
    $ ssh moon-01
    moon-01$ mongod

Sharding- mongos

- mongos
  - $ ssh ny-02
  - ny-02$ mongos --configdb ny-01,sf-01,moon-01

Sharding- shards

- Shards
  –$ mongo ny-02:27017/admin
  MongoDB shell version: 1.7.5
  connecting to: admin
  >

> db.runCommand({"addShard" : "sf-02:27017"})
{ "shardAdded" : "shard0000", "ok" : 1 }

Sharding- 开始

假设我们在做一个blog应用，因此所有的collection都存储在blog database。我们使用下列命令开启shard：

```
> db.adminCommand({"enableSharding" : "blog"})
{ "ok" : 1 }
```

shard key 为{"date" : 1, "author" : 1}.

```
> db.adminCommand({"shardCollection" : "blog.posts",
  key : {"date" : 1, "author" : 1}})
{ "collectionSharded" : "blog.posts", "ok" : 1 }
```

Sharding- 注意事项

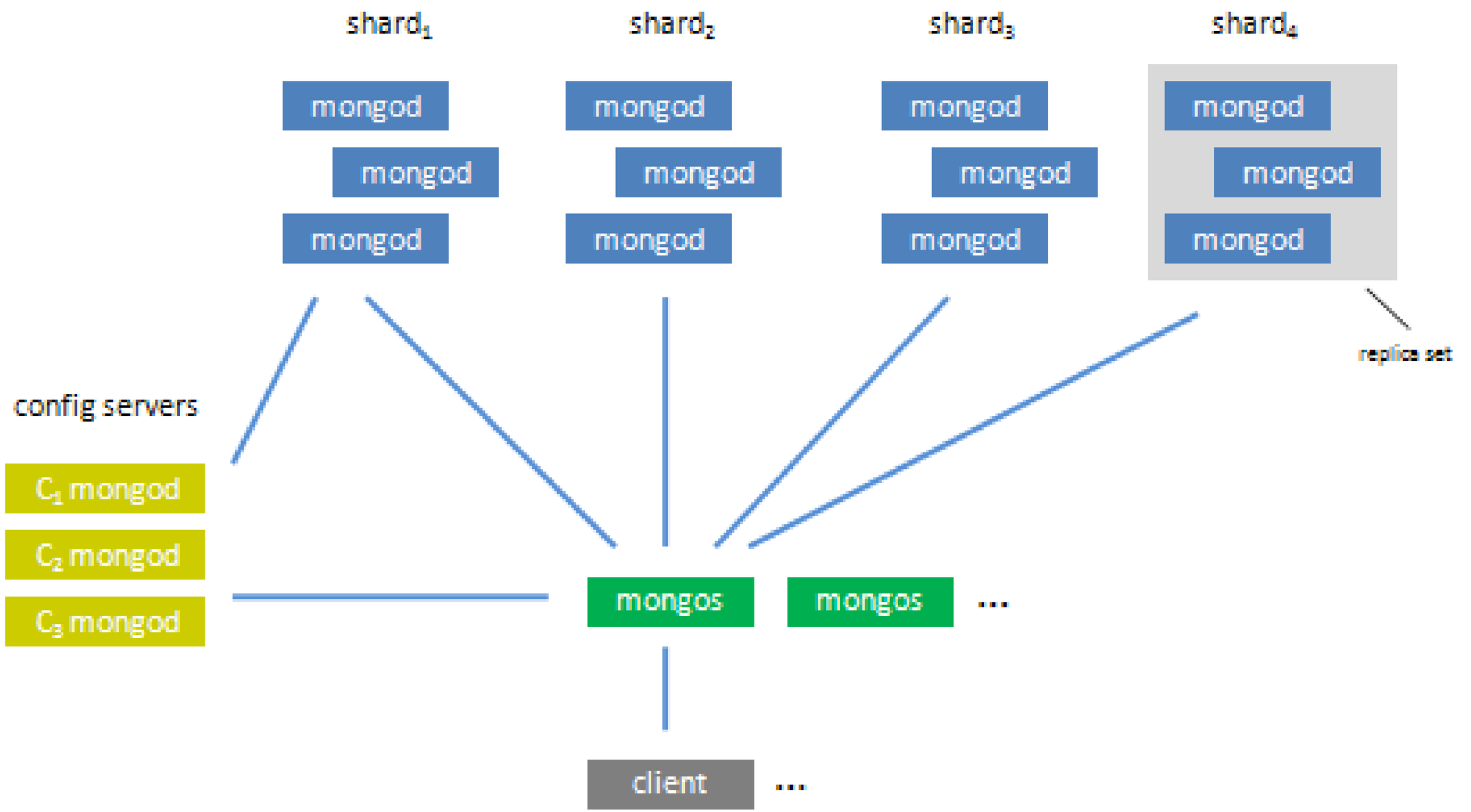
读写分离
db.getMongo().setSlaveOk()

Counting计数 误差

除了shard key 都不能保证唯一

单条记录的更新，加上shard-key作为条件

Sharding+Replica Set



MongoDB建议

DB Profiler (MySQL Slow logs)
>use blog
>db.setProfilingLevel(1,500)
> db.system.profile.find({millis:{$gt:10}}).sort({millis:-1})

后台创建索引
>db.posts.ensureIndex({author:1},{background:true})

冗余数据来提升性能，引用数据来保证数据的完整性

尽可能的用一条查询语句来获取数据

无限增长的数据字段不要内嵌

MongoDB建议

无论什么时候都要预分配空间

```
>collection.insert({"_id" : 123, /* other fields */, "garbage"
   : someLongString})
>collection.update({"_id" : 123}, {"$unset" : {"garbage" :
   1}})
```

数据完整性的代码
一致性修复
预分配
聚合器
Schema 校验器
备份任务

MongoDB建议

处理replica set的错误与故障转移
客户端处理，如何处理无法写入？队列？

最小化磁盘访问
添加内存
使用SSD

创建层级的document便于更快的检索

AND-查询顺序由小到大

在开发中，要使用安全写入

MongoDB监控

- munin
  - Server stats: this will retrieve server stats (requires python; uses http interface)
  - Collection stats, this will display collection sizes, index sizes, and each (configured) collection count for one DB (requires python; uses driver to connect)
- Ganglia:
  - ganglia-gmond
  - mongodb-ganglia
- cacti
- Mikoomi provides a MongoDB plugin for Zabbix
- Nagios
- mtop - A top like utility for Mongo
- Mongo Live - A Chrome extension that provides a real-time server status view (uses the rest interface).

MongoDB管理

db.addUser (username, password)
db.removeUser(username)

db.currentOp()
db.killOp()

db.foo.stats()
db.foo.dataSize()
db.foo.storageSize()
db.foo.totalIndexSize()
db.foo.totalSize()

实际案例

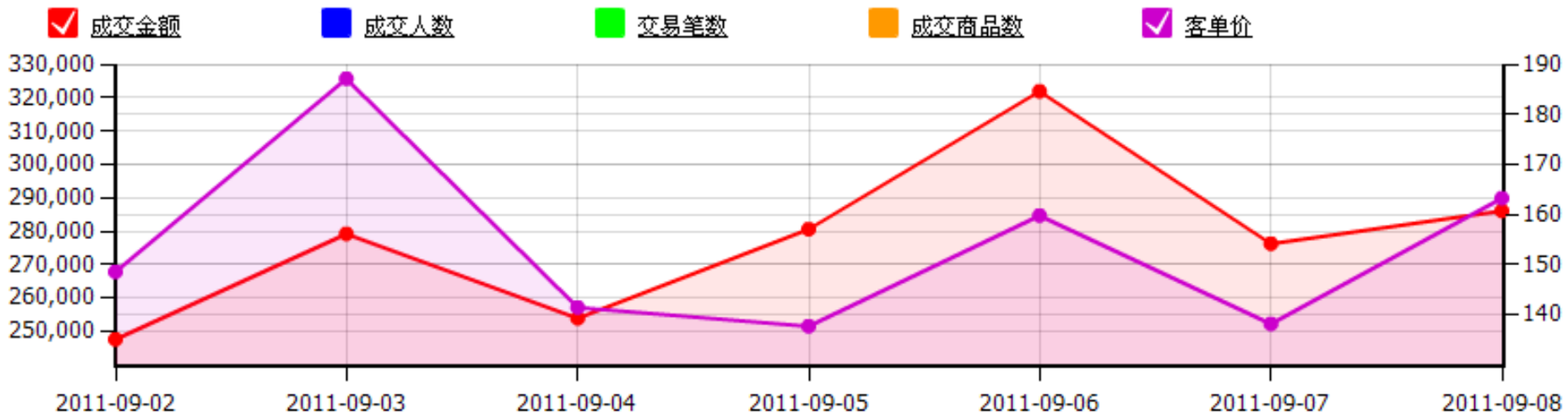
淘宝趋势分析 | 买家概要分析 | 买家精确分析 | 商品精确分析 | 返回监控店铺列表

成交趋势分析

昨天 | 最近7天 | 最近1个月 | 最近3个月 | 自定义

2011-09-02 - 2011-09-08

成交趋势图



成交概况

成交金额（￥）	成交人数	交易笔数	成交商品数	日平均客单价（￥）
1,944,860	12,760	13,621	59,823	154
日平均成交金额 277,837	日平均成交人数 1,822	日平均交易笔数 1,945	日平均成交商品数 8,546	

SACC2011

实际案例

机构荐股收益排行

分析师	总收益率	总荐股概率	推荐股票上涨数	推荐股票下跌数
黄珺	99.27%	57.38%	35	25
胡乔	92.50%	56.67%	34	25
施亮	69.70%	63.16%	12	7
毛长青	69.70%	63.16%	12	7
包敷文	50.91%	37.50%	3	5
闫亚磊	49.88%	56.76%	21	16
许娟娟	48.23%	66.67%	22	11
曾光	42.20%	45.12%	37	33
郭敏	39.26%	57.14%	12	7
谈煊	29.04%	78.95%	15	3

跳转: 1 显示行数: 10 第1页 共 111页 前一页后一页

所有互联网数据源

代理服务器爬取，防止搜索引擎，个别网站的屏蔽

垂准爬虫集群

Saitseo-news
228 11001

Saitseo-blog
228 11003

Saitseo-video
229 9005

Saitseo-weibo
229 9009

Saitseo-paper
229 9013

Saitseo-bbs
229 9003

通用爬虫集群

Mongodb 索引

news

news comment

video

blog

weibo

bbs

Lucene索引

API 模块

news api

bbs api

video api

blog api

weibo api

comment api

general api

Saitdiscover
商业版

Saitdiscover
政府版

前端APP应用(提供各种语言版本的包方便二次开发)

移动应用

口碑监测

猛烈点击如下地址

http://www.mongodb.org/display/DOCS/Production+Deployments

关于基础学习的问题

- 操作系统缓存
- 多线程，多进程
- 虚拟内存机制
- 文件系统

与君共勉

SACC2011

国内首部MongoDB著作,资深MongoDB亲自执笔
内容丰富全面，注重实战，数位数据库专家联袂推荐

数据库技术丛书

MongoDB in Action

MongoDB实战

夏远峰◎著

机械工业出版社
China Machine Press

互动交流

SACC2011

谢谢

SACC2011