Mongodb

入门PPT

mongoDB

简介

- MongoDB是一个跨平台，面向文档的数据库，提供高性能，高可用性和易于扩展。MongoDB是工作在集合和文档上一种概念
- 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。他支持的数据结构非常松散，是类似json的bson格式，因此可以存储比较复杂的数据类型。Mongo最大的特点是他支持的查询语言非常强大，其语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且还支持对数据建立索引
- 主要特性有：
- 面向集合存储，易于存储对象类型的数据
- 模式自由
- 支持动态查询
- 支持完全索引，包含内部对象
- 支持复制和故障恢复
- 使用高效的二进制数据存储，包括大型对象
- 文件存储格式为BSON(一种JSON的扩展)

Mongodb基本概念介绍

- 文档(document)是MongoDB中数据的基本单元，非常类似于关系型数据库系统中的行(但是比行要复杂的多)
- 集合(collection)就是一组文档，如果说MongoDB中的文档类似于关系型数据库中的行，那么集合就如同表
- MongoDB的单个计算机可以容纳多个独立的数据库，每一个数据库都有自己的集合和权限
- MongoDB自带简洁但功能强大的JavaScript shell，这个工具对于管理MongoDB实例和操作数据作用非常大
- 每一个文档都有一个特殊的键"_id",它在文档所处的集合中是唯一的，相当于关系数据库中的表的主键

NoSql简介

- NoSQL(Not Only SQL)，意即“不仅仅是SQL”,指的是非关系型的数据库。是一项全新的数据库革命性运动，早期就有人提出，发展至2009年趋势越发高涨。NoSQL的拥护者们提倡运用非关系型的数据存储，相对于铺天盖地的关系型数据库运用，这一概念无疑是一种全新的思维的注入。
- 关系型数据库中的表都是存储一些结构化的数据，每条记录的字段的组成都一样，即使不是每条记录都需要所有的字段，但数据库会为每条数据分配所有的字段。而非关系型数据库以键值对(key-value)存储，它的结构不固定，每一条记录可以有不一样的键，每条记录可以根据需要增加一些自己的键值对，这样就不会局限于固定的结构，可以减少一些时间和空间的开销。

常见的NoSql(非关系型数据库)数据库

- CouchDB
  Redis
  MongoDB
  Neo4j
  HBase
  BigTable

目前国内外正在应用NoSQL的网站有：	
新浪微博	Redis
Google	Bigtable
Amazon	SimpleDB
淘宝数据平台	Tair
视觉中国网站	MongoDB
优酷运营数据分析	MongoDB
飞信空间	HandlerSocket
豆瓣社区	BeansDB

NoSql数据库优缺点

- 在优势方面主要体现在下面几点：
  - 简单的扩展
  - 快速的读写
  - 低廉的成本
  - 灵活的数据模型
- 在不足方面主要有下面几点：
  - 不提供对SQL的支持
  - 支持的特性不够丰富
  - 现有的产品不够成熟

mongoDB和关系型数据库的对比图

对比项	mongoDB	mysql oracle
表	集合	二维表table
表的一行数据	文档document	一条记录record
表字段	键key	字段field
字段值	值value	值value
主外键	无	PK,FK
灵活度扩展性	极高	差

1.关系数据的表的record必须保证拥有每一个field
2.mongoDB的每一个document的key可以不一样
3.关系型数据查询使用SQL
4.mongoDB查询使用内置find函数→基于BSON的特出查询工具

1、逻辑结构关系对比

关系型数据库：

MySQL数据库(database)、表(table)、记录(rows)三个层次概念组成。

非关系型数据库：

MongoDB 数据库(database)、集合(collection)、文档对象(document)三个层次概念组成。

MongoDB 里的集合对应于关系型数据库里的表，但是集合中没有列、行和关系的概念，集合中只有文档，一个文档就相当与一条记录，这体现了模式自由的特点。

Linux下安装Mongodb

- 先在Linux下安装JDK，并配置Java环境变量
- 解压 mongodb-linux-x86_64-2.6.4.tgz 到所存放的目录如：/usr/local/mongodb
- 启动mongodb（启动前，先指定mongodb的data、log目录，如果没有就创建一个）

执行如下命令即可启动： ./mongod --dbpath=/usr/local/mongodb/data/ --logpath=/usr/local/mongodb/log --fork

--dbpath==>数据文件存放路径 --logpath==>数据库日志输出文件 --fork 后台起动

启动配置

cat /etc/profile.d/mongodb.sh

vim mongodb.sh(进入)

#set mongo env

MONGODB_HOME=/usr/local/mongodb/

PATH=$PATH:$MONGODB_HOME/bin（保存退出）

查看是否启动成功了，默认端口号是27017，当然在启动时也可以指定未使用的其它端口。

最后，将客户端mogo文件在/bin下软链接，方便随处执行：

Mongodb基本操作

- show dbs:显示数据库列表
- show collections：显示当前数据库中的集合（类似关系数据库中的表）
- show users：显示用户
- use <db name>：切换当前数据库，这和MS-SQL里面的意思一样
- db.help()：显示数据库操作命令，里面有很多的命令
- db.foo.help()：显示集合操作命令，同样有很多的命令，foo指的是当前数据库下，一个叫foo的集合，并非真正意义上的命令
- db.foo.find()：对于当前数据库中的foo集合进行数据查找（由于没有条件，会列出所有数据）
- db.foo.find({ a : 1 })：对于当前数据库中的foo集合进行查找，条件是数据中有一个属性叫a，且a的值为1
- MongoDB没有创建数据库的命令，但有类似的命令。
- 修复当前数据库 db.repairDatabase();
- 查看当前使用的数据库

  db.getName();

  db; db和getName方法是一样的效果，都可以查询当前使用的数据库
- 显示当前db状态 db.stats();
- 当前db版本 db.version();
- 查看当前db的链接机器地址 db.getMongo();

Mongodb基本操作

- //查询　　db.foo.find()
- //新增　　创建集合有两种方式，显示创建和隐式创建显示创建可以使用命令 db.createCollection("集合名称")

  隐式创建可以使用命令 db.集合名称.insert({}),指创建集合并同时向集合中插入数据,例如：db.customer.insert({name:"jack"})

  db.collection.insert()

  db.collection.save()==>可以手动指定 _id
- //删除

  db.collection.drop()==》删除集合

  db.collection.remove({"_id":1})==>删除记录
- //修改

  db.collection.update({age:10},{$set:{name:"updatename"}})
- //排序　1升序 -1降序

  db.foo.find({$or:[{age:{$lt:10}},{age:{$gt:999998}}]}).sort({age:1})
- //分页

  skip==>从几开始

  limit==>取几条

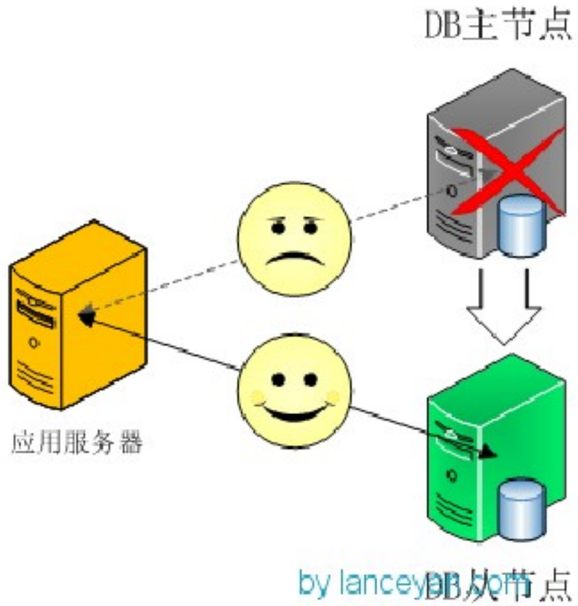
  db.foo.find().skip().limt()

  findOne查询第一条

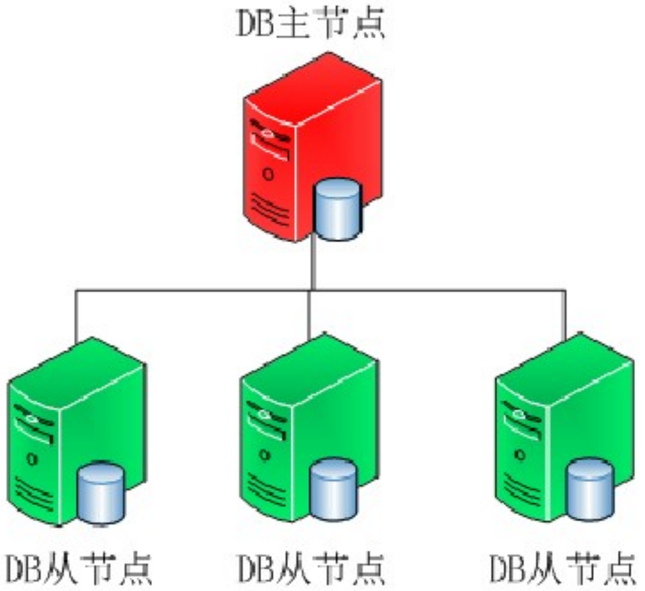
  db.foo.findOne()

Mongodb集群搭建

主从模式。使用mysql数据库时大家广泛用到，采用双机备份后主节点挂掉了后从节点可以接替主机继续服务。所以这种模式比单节点的高可用性要好很多。



多个从节点。现在只是一个数据库服务器又提供写又提供读，机器承载会出现瓶颈。大家还记得mysql里的读写分离吗？ 把20%的写放到主节点，80%的读放到从节点分摊了减少了服务器的负载。但是大部分应用都是读操作带来的压力，一个从节点压力负载不了，可以把一个从节 点变成多个节点。那mongodb的一主多从可以支持吗？答案是肯定的。



Mongodb集群搭建（1）

- Mongodb一共有三种集群搭建的方式：Replica Set（副本集）、Sharding（切片）和Master-Slaver（主从）。下面要搭建的是Sharding，Sharding集群也是三种集群中最复杂的。
- 下面看一下怎么一步步搭建一个mongodb的主从复制节点：
- 1、准备两台机器 192.168.0.1 和 192.168.0.2。 192.168.0.1 当作**主节点**，192.168.0.2作为**从节点**。
- 2、分别下载mongodb安装程序包。在192.168.0.1上建立文件夹 /data/mongodbtest/master，192.168.0.2建立文件夹 /data/mongodbtest/slave。
- 3、在192.168.0.1启动mongodb主节点程序。注意后面的这个 “ **-master** ”参数，标示主节点。mongod -dbpath /data/mongodbtest/master -master 输出日志如下，成功！

```
[initandlisten] MongoDB starting : pid=18285 port=27017
dbpath=/data/mongodbtest/master master=1
#日志显示主节点参数
[initandlisten] options: { dbpath: “/data/mongodbtest/master”, master:
true }
……..
[initandlisten] waiting for connections on port 27017
```

Mongodb集群搭建（2）

- 4、在192.168.0.2启动mongodb从节点程序。关键配置，指定主节点ip地址和端口 -source 192.168.0.1:27017 和 标示从节点 -source 参数。

  mongod -dbpath /data/mongodbtest/slave **-slave -source** 192.168.0.1:27017

  输出日志如下，成功！

  [initandlisten] MongoDB starting : pid=17888 port=27017 dbpath=/data/mongodbtest/slave slave=1
  ……..
  #日志显示从节点参数
  [initandlisten] options: { dbpath: “/data/mongodbtest/slave”, **slave: true, source**: “192.168.0.1:27017” }
  ……..
  [initandlisten] waiting for connections on port 27017
  #日志显示从节点 从主节点同步复制数据
  [replslave] repl: from host:192.168.0.1:27017

- 5、测试主从复制

  在主节点上连接到终端：

  查询后数据已经同步过来了。再看看日志，发现从主机确实从主机同步了数据。

  查看服务状态

Mongodb安全验证（1）

- 登录

```
mkdir /usr/local/mongodb/authData
touch /usr/local/mongodb/authLogs
pkill mongo
mongod --dbpath=/usr/local/mongodb/authData/ -- logpath=/usr/local/mongodb/authLogs --fork
mongo -u root -p root --authenticationDatabase admin
```

- 进入数据库中切换用户

```
use admin
db.auth("username","password")
```

- 创建一般用户

```
db.createUser(
 {
  user:"oneUser",
  pwd:"12345",
  roles:[
   {role:"read",db:"db01"},
   {role:"read",db:"db02"},
   {role:"read",db:"db03"}
  ]
 }
)
```

Mongodb安全验证（2）

- 创建超级用户(不受限制)

```
db.createUser(
 {
   user:"oneUser",
   pwd:"12345",
   roles:["root"]
 }
)
```

- 创建一个有读写权限的用户 超级

```
db.createUser(
 {
   user: "tom",
   pwd: "tom",
   roles:
   [
     {
       role: "userAdminAnyDatabase",
       db: "admin"
     }
   ]
 }
)
```

查看当前用户的信息

```
db.runCommand({usersInfo:"userName"})
```

查看当前用户的权限

```
db.runCommand(
 {
   usersInfo:"userName",
   showPrivileges:true
 }
)
```

修改密码

```
use admin
db.changeUserPassword("username", "xxx")
```

Mongodb副本集搭建

- **副本集中数据同步过程：**Primary节点写入数据，Secondary通过读取Primary的oplog得到复制信息，开始复制数据并且将复制信息写入到自己的oplog。如果某个操作失败，则备份节点停止从当前数据源复制数据。如果某个备份节点由于某些原因挂掉了，当重新启动后，就会自动从oplog的最后一个操作开始同 步，同步完成后，将信息写入自己的oplog，由于复制操作是先复制数据，复制完成后再写入oplog，有可能相同的操作会同步两份，不过MongoDB 在设计之初就考虑到这个问题，将oplog的同一个操作执行多次，与执行一次的效果是一样的。**简单的说就是：**
- 当Primary节点完成数据操作后，Secondary会做出一系列的动作保证数据的同步：
  1：检查自己local库的oplog.rs集合找出最近的时间戳。
  2：检查Primary节点local库oplog.rs集合，找出大于此时间戳的记录。
  3：将找到的记录插入到自己的oplog.rs集合中，并执行这些操作。

Mongodb副本集搭建（1）

- 规划：

  1个: Master-->主人 192.168.1.2

  2 slave replication

  192.168.1.3

  192.168.1.4

  主从集群：master/slave 缺点：主节点挂了。就废了。

  推荐：副本集 副本集节点最好奇数台

  仲裁节点

SSH
secure shell
ssh免密登录
ssh-keygen -t rsa 三个空格
ssh-copy-id hostname
(copy公钥(~/.ssh/id_rsa.pub)到目标主机的家目录的 .ssh/authorized_keys)
就可以免密登录
ssh hostname
发送命令到远程机器
ssh user@hostname commond
e.g ssh root@192.168.1.3 mkdir folder
发送文件或文件夹到远程机器
scp -r folder root@192.168.1.3:/path

Mongodb副本集搭建（2）

- mongod --replSet application --dbpath /data/node1/ --port 9927 --oplogSize 1024

其中application是副本集的名称，节点必须相同，-dbpath指定数据库储存路径，-port指定侦听端口，-oplogSize指定数据同步之前的缓存的空间大小

- 初始化副本集需要配置表，申明配置表如下：

config = {_id: "application", members: []}

注意_id和副本集启动的共享名称一致。下面来逐步添加节点的数据信息：

config.members.push({_id: 0, host: "centos:27017"})

config.members.push({_id: 1, host: "centos2:27017"})

config.members.push({_id: 2, host: "centos3:27017", arbiterOnly: true})

也可以使用rs.add和rs.addArb函数来实现同样的操作。然后需要用这个表作为参数初始化副本集，在9927端口的shell执行：

- rs.initiate(config)

返回ok为1表示初始化成功，三个节点互相检测通信，需要1分钟左右(反应够慢的^-)，可以查看三个终端窗口的信息确认，完成通信后，在master的shell回车执行命令确认配置：

- rs.isMaster()
- rs.slaveOk()，只对当前连接有效。
- 查看副本集的状态 rs.status()