MEMCACHED 缓存数据库应用实践

惨绿少年 Linux运维, NoSQL, 数据库 0评论 来源：本站原创 38℃ 字体： 小 中 大

1.1 数据库对比

缓存： 将数据存储到内存中，只有当磁盘胜任不了的时候，才会启用缓存

缺点：断电数据丢失(双电)，用缓存存储数据的目的只是为了应付大并发的业务。

数据库： mysql(关系型数据库，能够保证数据一致性，保证数据不丢失，当因为功能太多，导致性能不高) === 数据参考

缓存数据库： memcache redis(非关系型数据库，性能极高，但不保证数据完整性) === 业务的数据提供者

memcachedb 会将内存的数据写入到磁盘中

redis 主要工作场所是内存中，但是定期备份内存数据到硬盘

1.1.1 数据库的选择

数据存储，数据仓库选择mysql这种磁盘的数据库

高并发，业务大的应用选择memcache这种内存数据库

1.1.2 数据库分类

关系型数据库 mysql

非关系型数据库（NOSQL） memcached redis MongoDB

1.2 memcached介绍

Memcached是一款开源的、高性能的纯内存缓存服务软件。Mem是内存的意思，cache是缓存的意思，d是daemon的意思。

memcache 是项目名称，也是一款软件，其架构是**C/S架构。**

memcached官网：http://memcached.org/

1.2.1 memcache优点

① 对于用户来讲，用户访问网站更快了，体验更好了。

②对网站来说，数据库压力降低了。只有当内存没有数据时才会去请求数据库。第一次写入的数据也会请求数据库。一般公司没有预热，只有当用户读取过数据库才会放到Memcached中。

② 提升了网站的并发访问，减少服务器数量。

1.3 Memcached在企业中使用场景

1.3.1 作为数据库的前端缓存应用

当数据库（mysql）承受不了大并发的请求时，可以将数据缓存到内存中（缓存数据库），然后就可以解决

作为数据库的前端缓存最大目的：减少数据库被大量访问的压力

1.3.2 作为集群后端的session会话保持

session存储在文件，数据库，memcache，或内存等的服务端上，

cookie 存放在客户端浏览器上。

session是一个存在服务器上的类似于一个散列表格的文件。里面存有我们需要的信息，在我们需要用的时候可以从里面取出来。

session依赖cookie存在，请求客户端到达服务端后，服务端会随机生成一个字符串，作为该用户的标识，该字符串通过cookie返回给客户端，客户端浏览器会以该字符串为key放到session id里面，随机字符串的key里面可以先没有值。如果用户再次提交，请求信息中的用户名密码等用户信息保存在随机字符串的value中，请求到达服务端，用户名密码正确，随机字符串会被授权，提一个标记给到sessionid中的随机字符串的value中，证明该用户已经是登录状态，客户端再次带着该随机字符串访问服务端，服务端会知道该用户已经登录不需验证，直接返回请求的信息。

session和cookie区别

1、cookie数据存放在用户的浏览器上，session数据存储在服务器上

2、cookie在本地的浏览器中，可以被提取分析，安全性差。为了安全，登录账户等信息可以缓存在session中。

3、session会在一定时间内保存在服务器上，访问量增大会给服务器带来压力，可以使用缓存工具，如memcache等

1.3.3 网站开发如何判断用户信息

最开始的技术方法：服务器在你的浏览器中写一个cookies，这个cookies就包含了你的用户名及登录信息。因为cookies是存储在本地浏览器中，所以第三方工具很容易盗取cookies信息。

最开始：

cookies cookies名字：内容（用户名，登录信息）

改进后：

本地浏览器存放：

cookies cookies名字：内容（session id 编号）

服务器存放：

session session id：内容（用户名，登录信息）

主流使用场景：cookies + session

1.3.4 session共享的不同解决方案

1、session文件提供NFS共享

2、session文件提供rsync scp共享

3、将session的内容存放在数据库（mysql）中，所有的机器都可以通过ip：port读取

4、将session的内容存放在缓存数据库中，所有的机器都可以通过ip：port读取

好处：利用断电、重启丢失数据的特性。定时清理数据；提高并发

1.3.5 memcache原理优点

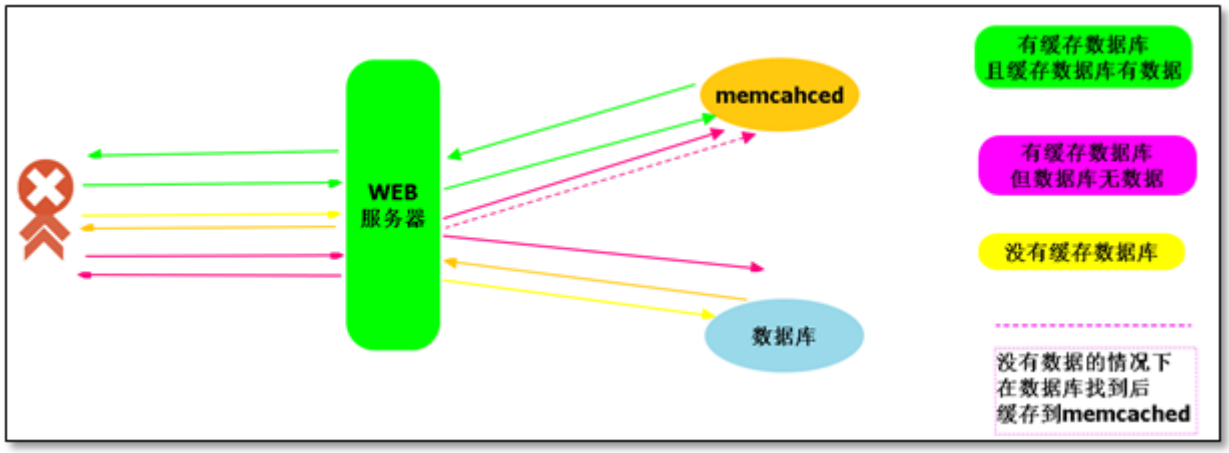
启动Memcached时，根据指定的内存大小参数，会被分配一个内存个间。当我们读取数据库的各类业务数据后，数据会同时放入Memcached缓存中，，下一次用户请求同样的数据，程序直接去Memcached取数据返回给用户。

优点：

① 对于用户来讲，用户访问网站更快了，体验更好了。#

② 对网站来说，数据库压力降低了。只有当内存没有数据时才会去请求数据库。第一次写入的数据 也会请求数据库。一般公司没有预热，只有，用户读取过数据库才会放到Memcached中。

③ 提升了网站的并发访问，减少服务器数最。



原理图

1.4 Memcached分布式缓存集群

memcached天生不支持分布式集群,需要通过程序支持分布式存储

1.4.1 Memcached分布式缓存集群的特点

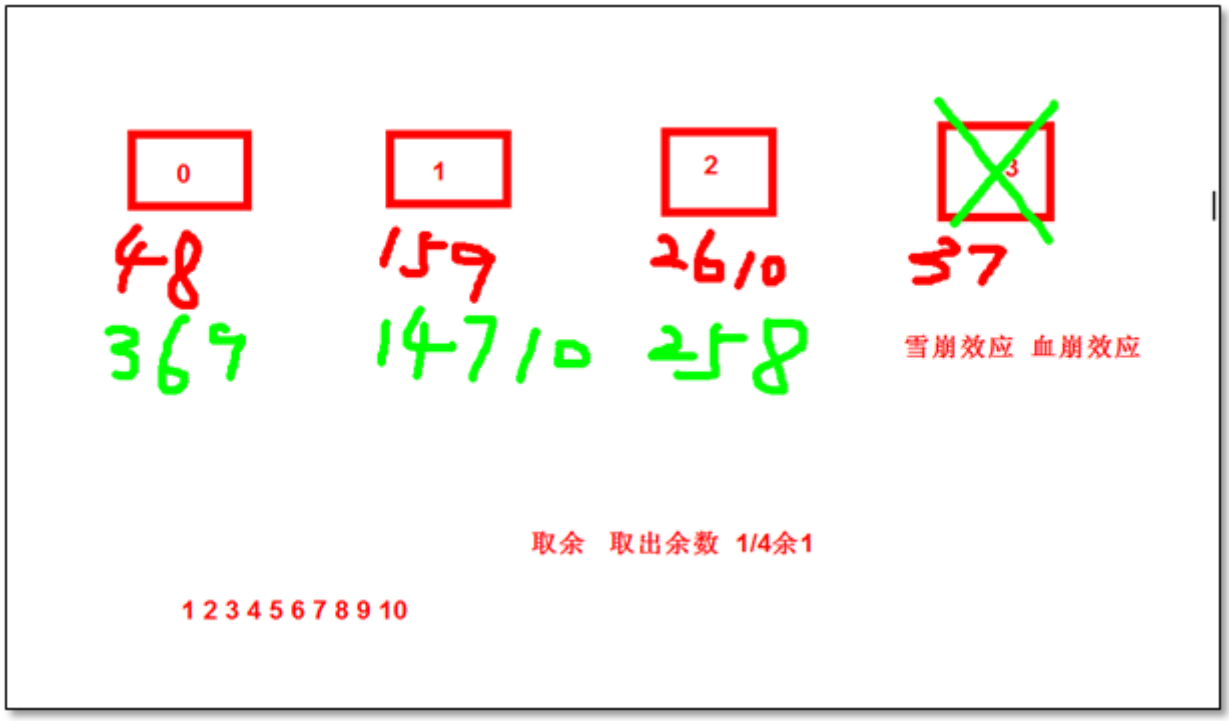
1. 所有MC服务器内存的内容都是不一样的。这些服务器内容加起来接近数据库的容量。比如1T的数据库，一台缓存数据库的内存没有那么大，因此分成10台缓存服务器。

2. 通过在客户端(Web)程序或者MC的负载均衡器上用HASH算法，让同一内容都分配到一个MC服务器。

3. 普通的HASH算法对于节点宕机会带来大量的数据流动(失效)，可能会引起雪崩效应。

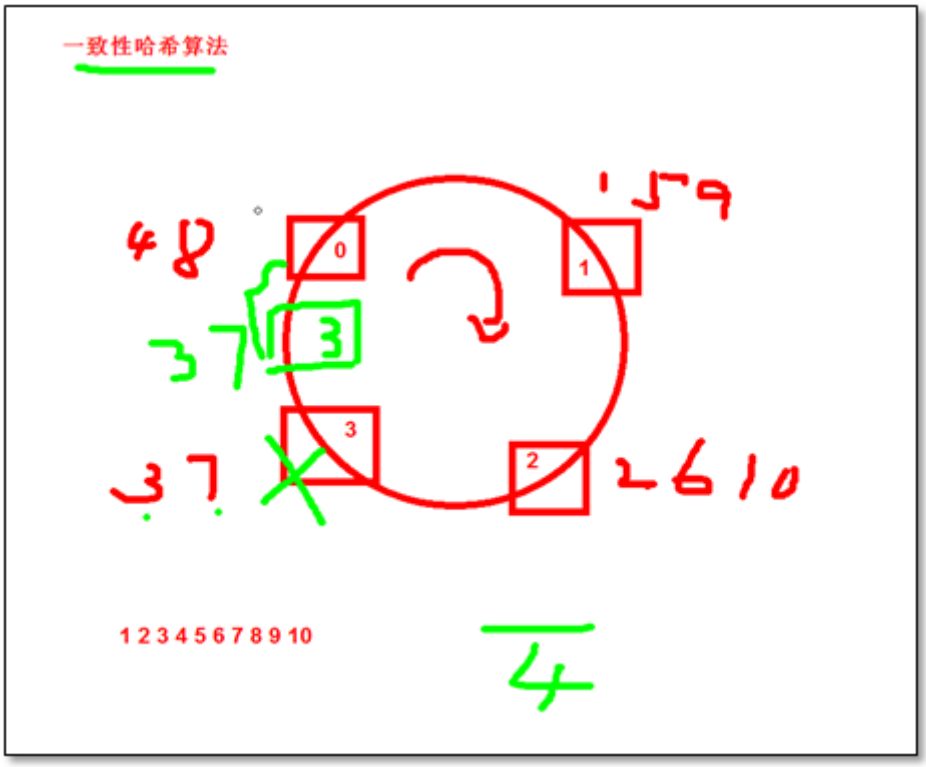
4. 一致性HASH可以让节点宕机对节点的数据流动(失效)降到最低。

普通的*hash*算法



首先将key处理为一个32位字符串，取前8位，在经过hash计算处理成整数并返回，然后映射到其中一台服务器这样得到其中一台服务器的配置，利用这个配置完成分布式部署。在服务器数量不发生变化的情况下，普通hash分布可以很好的运作，当服务器的数量发生变化，问题就来了。试想，增加一台服务器，同一个key经过hash之后，与服务器取模的结果和没增加之前的结果肯定不一样，这就导致了，之前保存的数据丢失。

一致性*hash*算法



一致性哈希算法

优点：在分布式的cache缓存中，其中一台宕机，迁移key效率最高

将服务器列表进行排序，根据mHash($key) 匹配相邻服务器

一致性hash算法 将数据流动降到最低

参考资料

```
http://blog.csdn.net/cywosp/article/details/23397179
http://blog.csdn.net/zhangskd/article/details/50256111
```

第2章 memcached使用

2.1 安装memcached

2.1.1 环境说明

```
[root@cache01 ~]# cat /etc/redhat-release
CentOS Linux release 7.4.1708 (Core)
[root@cache01 ~]# uname -r
3.10.0-693.el7.x86_64
[root@cache01 ~]# getenforce
Disabled
[root@cache01 ~]# systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@cache01 ~]# hostname -I
10.0.0.21 172.16.1.21
```

2.1.2 安装memcached

```
[root@cache01 ~]# yum -y install memcached
```

2.1.3 查看配置

```
[root@cache01 ~]# cat /etc/sysconfig/memcached
PORT="11211"
USER="memcached"
MAXCONN="1024"
CACHESIZE="64"
OPTIONS=""
```

2.1.4 查看启动脚本

```
[root@cache01 ~]# cat /usr/lib/systemd/system/memcached.service
[Unit]
Description=Memcached
Before=httpd.service
After=network.target

[Service]
Type=simple
```

```
EnvironmentFile=-/etc/sysconfig/memcached
ExecStart=/usr/bin/memcached -u $USER -p $PORT -m $CACHESIZE -c $MAXCONN $OPTIONS

[Install]
WantedBy=multi-user.target
```

2.1.5 启动服务

```
[root@cache01 ~]# systemctl start memcached.service
```

2.2 管理memcached

2.2.1 memcached数据库语法格式

```
set             key   0       0          10

     \r\n
```

\n 换行且光标移至行首

\r 光标移至行首，但不换行

参数	说明
	是在取回内容时，与数据和发送块一同保存服务器上的任意16位无符号整形（用十进制来书写）。客户端可以用它作为“位域”来存储一些特定的信息；它对服务器是不透明的。
	是终止时间。如果为0，该项永不过期(虽然它可能被删除，以便为其他缓存项目腾出位置)。如果非0（Unix时间戳或当前时刻的秒偏移），到达终止时间后，客户端无法再获得这项内容
	是随后的数据区块的字节长度，不包括用于分页的“\r\n”。它可以是0（这时后面跟随一个空的数据区块）。
\r\n	是大段的8位数据，其长度由前面的命令行中的指定。

2.2.2 数据库使用

写入读取数据

```
[root@cache01 ~]# printf "set key008 0 0 10\r\noldboy0987\r\n"|nc 10.0.0.21 11211
STORED
[root@cache01 ~]# printf "get key008\r\n"|nc 10.0.0.21 11211
VALUE key008 0 10
oldboy0987
END
```

写入数据长度不符合，定义过大

```
[root@cache01 ~]# printf "set key009 0 0 11\r\noldboy0987\r\n"|nc 10.0.0.21 11211
[root@cache01 ~]# printf "get key009\r\n"|nc 10.0.0.21 11211
END
```

写入数据长度不符合，定义过小

```
[root@cache01 ~]# printf "set key010 0 0 9\r\noldboy0987\r\n"|nc 10.0.0.21 11211
CLIENT_ERROR bad data chunk
ERROR
[root@cache01 ~]# printf "get key010\r\n"|nc 10.0.0.21 11211
END
```

时效性

```
[root@cache01 ~]# printf "set key011 0 10 10\r\noldboy0987\r\n"|nc 10.0.0.21 11211
STORED
[root@cache01 ~]# printf "get key011\r\n"|nc 10.0.0.21 11211
VALUE key011 0 10
oldboy0987
END
[root@cache01 ~]# printf "get key011\r\n"|nc 10.0.0.21 11211
END
```

删除数据

```
[root@cache01 ~]# printf "delete key008\r\n"|nc 10.0.0.21 11211
DELETED
[root@cache01 ~]# printf "get key008\r\n"|nc 10.0.0.21 11211
END
```

2.3 memcache php版本客户端安装使用

命令集

```
#编译进去php_mem
tar zxvf memcache-2.2.5.tgz
cd memcache-2.2.5
/application/php/bin/phpize
./configure --enable-memcache --with-php-config=/application/php/bin/php-config --with-zlib-dir
make
make install
# 激活php_memcached
sed -i '$a extension=memcache.so' /application/php/lib/php.ini
pkill php
/application/php/sbin/php-fpm -t
/application/php/sbin/php-fpm
/application/php/bin/php -m|grep memcache
```

检查当前环境

查看php的模块

⊞⊟

```
查看php的模块
[root@web06 ~]# /application/php/bin/php -m
[PHP Modules]
bcmath
Core
ctype
curl
date
dom
ereg
fileinfo
filter
ftp
gd
hash
iconv
json
libxml
mbstring
mcrypt
mhash
mysql
mysqlnd
openssl
pcntl
pcre
PDO
pdo_mysql
pdo_sqlite
Phar
posix
Reflection
session
shmop
SimpleXML
soap
sockets
SPL
sqlite3
standard
sysvsem
tokenizer
xml
xmlreader
xmlrpc
xmlwriter
xsl
zlib

[Zend Modules]
```

View Code 查看php的模块

执行过程

编译安装

```
[root@web06 memcache-2.2.5]# make install
Installing shared extensions:     /application/php-5.5.32/lib/php/extensions/no-debug-non-zts-20
[root@web06 memcache-2.2.5]# ls /application/php/lib/php/extensions/no-debug-non-zts-20121212/
memcache.so
[root@web06 memcache-2.2.5]# sed -i '$a extension=memcache.so' /application/php/lib/php.ini
[root@web06 memcache-2.2.5]# pkill php
[root@web06 memcache-2.2.5]# /application/php/sbin/php-fpm -t
[17-Nov-2017 11:39:13] NOTICE: configuration file /application/php-5.5.32/etc/php-fpm.conf test

[root@web06 memcache-2.2.5]# /application/php/sbin/php-fpm
[root@web06 memcache-2.2.5]# /application/php/bin/php -m|grep memcache
memcache
```

2.3.1 编写测试文件

```
[root@web01 blog]# cat /application/nginx/html/blog/mc.php
php
    $memcache = new Memcache;
    $memcache->connect('10.0.0.21', 11211) or die ("Could not connect");
    $memcache->set('key20171117', 'hello,world');
    $get_value = $memcache->get('key20171117');
    echo $get_value;
?>
```

浏览器访问

数据库读取测试

```
[root@cache01 ~]# printf "get key20171117 \r\n"|nc 10.0.0.21 11211
VALUE key20171117 0 11
hello,world
END
```

2.4 web管理memcached

使用的软件memadmin

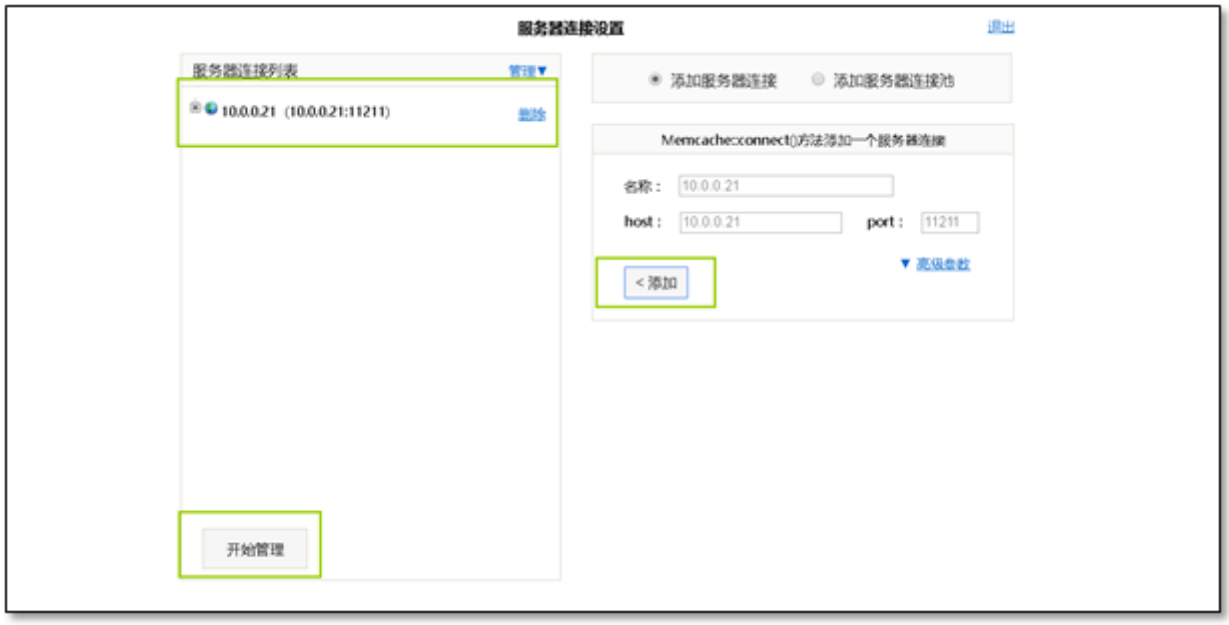
官网：http://www.junopen.com/memadmin/

将程序包放如站点目录，浏览器进行访问即可

```
[root@web06 tools]# tar xf memadmin-1.0.12.tar.gz -C /application/nginx/html/blog/
```

默认用户名密码为admin

添加一个新的memcached服务器



web界面管理全中文，较为简单

2.5 memcached数据缓存

通过程序实现

2.5.1 blog站点实现memcached存储

```
[root@web06 ~]# cat /application/nginx/html/blog/wp-content/object-cache.php
```

```
function WP_Object_Cache() {
        global $memcached_servers;

        if ( isset($memcached_servers) )
                $buckets = $memcached_servers;
        else
                $buckets = array('172.16.1.21','');

        reset($buckets);
        if ( is_int( key($buckets) ) )
                $buckets = array('default' => $buckets);

        foreach ( $buckets as $bucket => $servers) {
                $this->mc[$bucket] = new Memcache();
                foreach ( $servers as $server  ) {
                        list ( $node, $port ) = explode(':', $server);
                        if ( !$port )
                                $port = ini_get('memcache.default_port');
```

2.6 memcached session共享

方法1：

通过程序实现，web01只需要往memcahce写session，web02从memcahce读session**（更具有通用性）**

方法2：

通过php的配置文件，让php默认将session存储在文件中，修改为存储在memcached中

```
sed -i 's#session.save_handler = files#session.save_handler = memcache#;$a session.save_path = "
```

使用这个功能，需要使用php的session函数

本文出自“惨绿少年”，欢迎转载，转载请注明出处！http://blog.znix.top

赞0