**Morning：**

1、数据库简介（SQL&NoSQL）

1）传统Web架构问题

Web应用的数据都存在RDBMS（关系型数据库），当访问量加大，数据库的响应恶化。

2）数据存储位置对比（性能+价格）

CPU缓存 > 内存 > 磁盘 > 数据库(租用)

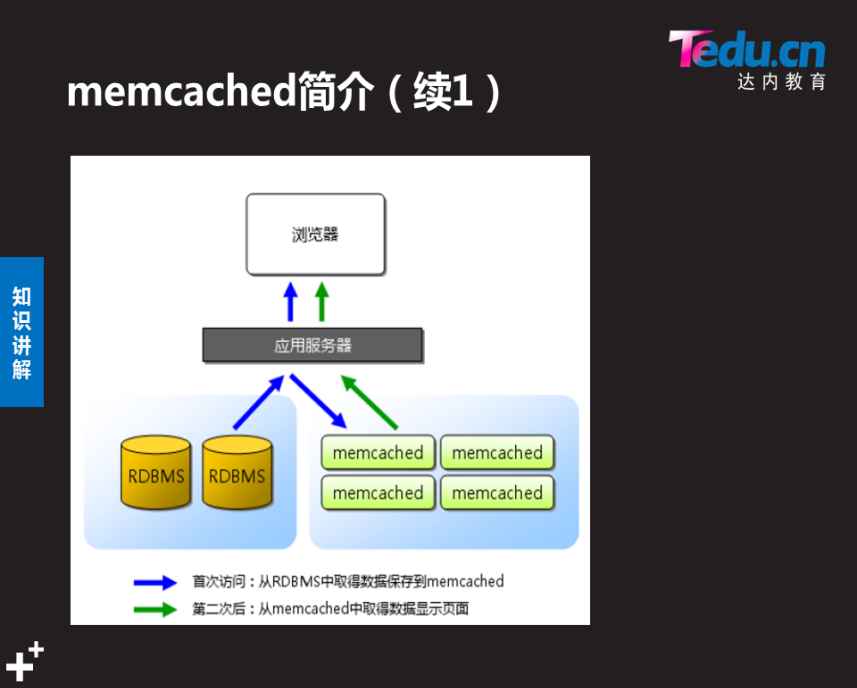
3）SQL（关系型数据库）与NoSQL

SQL：存硬盘，含有数据库-表等数据关系。

NoSQL：存内存，非关系（mogondb，redis，memcached）。速度快，但重启可能丢失（需要同步到硬盘），无法记录关系。

1. memcached原理

客户首次访问动态网站，通过脚本从硬盘读取数据，返回时服务器同时将数据存一份到内存，第二次访问时通过内存获取数据。



**Afternoon：**

1. 搭建memcached服务

**-构建memcached服务**

1. 安装memcached软件

[root@proxy ~]# yum -y install memcached

1. 查看memcached配置文件（不需修改）

vim /usr/lib/systemd/system/memcached.service

[Service]

EnvironmentFile=-/etc/sysconfig/memcached

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

vim /etc/sysconfig/memcached

PORT="11211" #监听端口号

USER="memcached" #程序用户名

MAXCONN="1024" #最大连接数，并发量

CACHESIZE="64" #最大占用内存64M

OPTIONS=""

1. 启动memcached服务

systemctl start memcached

netstat -anptu | grep 11211

**-使用telnet测试memcached服务器功能**

1. 安装telnet软件

[root@proxy ~]# yum -y install telnet

1. 使用telnet测试memcached服务器功能

[root@proxy ~]# telnet 192.168.4.5 11211

###########测试增、删、改、查#################

#0为不压缩，180s为缓存时间，10为字节数

>add myname 0 180 10 #新建变量，已存在则报错

>replace myname 0 180 10 #替换变量，不存在则报错

>set myname 0 180 10 #添加或替换变量

>get name #读取变量的值

>append myname 0 180 10 #向变量中追加数据

>delete myname #删除变量

>lush\_all #清空所有

>stats

#查看状态，cmd为总数hits为正常misses为失败

>quit #退出登录

#####################################

2、LNMP+memcached

**步骤1：部署LNMP环境**

参考阶段2-->OPERATION-->DAY02-->page02

**步骤2：启动服务**

nginx(80)、mariadb(3306)、php-fpm(9000)、memcached(11211)

**步骤3：创建PHP页面，使用PHP语言测试memcached服务**

1. 部署测试页面

cp /root/lnmp\_soft/php\_scripts/mem.php /usr/local/nginx/html

1. 客户端测试（结果会失败）

[root@client ~]# firefox <http://192.168.4.5/mem.php>

1. 为PHP添加memcache扩展

[root@proxy ~]# yum list | grep memcache

[root@proxy ~]# yum -y install php-pecl-memcache

[root@proxy ~]# systemctl restart php-fpm

4）客户端再次测试（结果成功）

[root@client ~]# firefox <http://192.168.4.5/mem.php>

1. PHP的本地Session信息（暂无法实现动态页面同步）

**-关于集群访问动态页面结果不同步**

1）hash加密

2）通过调用共享memcache数据库的信息，达到数据共享

**-关于session与cookie**

1）client发送请求 ---> server读（保存uuid文件session，随机乱码）

2）server回应网页的内容+头部信息cookie:uuid ---> client

3）client再次访问cookie:uuid ------> server查找uuid，无需重新登陆

**步骤1：部署后端LNMP服务器+动静分离**

**步骤2：部署前端Nginx调度服务器（地址池：web1+web2）**

**步骤3：部署带登陆界面的测试页面**

1）web1部署测试页面（带登陆界面）

[root@web1 ~]# cd lnmp\_soft/php\_scripts/

[root@web1 php\_scripts]# tar -xf php-memcached-demo.tar.gz

[root@web1 php\_scripts]# cd php-memcached-demo

[root@web1 php-memcached-demo]# cp -a \* /usr/local/nginx/html/

1. 真机访问web1测试页面

[root@room9pc01 ~]# firefox 192.168.2.100/index.php

#输入用户名+密码

[root@web1 ~]# cd /var/lib/php/session/ && ls

sess\_ahilcq9bguot0vqsjtd84k7244

#查看web1服务器本地的Session信息，文件记录了用户信息

1. web2部署测试页面，真机访问web2测试页面

注意：对index.php和home.php添加<body bgcolor=green>加以区分

4）浏览器访问前端调度器测试（推荐google浏览器）

[root@room9pc01 ~]# http://192.168.4.5/index.php

#填写信息后刷新，还需要再次注册，两台服务器用的是本地session

1. PHP实现session共享

**步骤1：在proxy构建memcached服务**

**步骤2：在后端LNMP服务器上部署Session共享**

1）为web1、web2添加php-memcache扩展包

[root@web1 ~]# yum -y install php-pecl-memcache

2）修改php-fpm配置文件，重启服务

[root@web1 ~]# vim /etc/php-fpm.d/www.conf

最后两行，修改前：

php\_value[session.save\_handler] = files

php\_value[session.save\_path] = /var/lib/php/session

#定义session会话信息存在本地，目录为/var/.../session

最后两行，修改后：

php\_value[session.save\_handler] = memcache

php\_value[session.save\_path] = "tcp://192.168.2.5:11211"

#定义session信息存在公共的memcached服务器上，对应的服务器IP及端口

1. 重启php-fpm服务

[root@web1 ~]# systemctl restart php-fpm

**步骤3：客户端测试**

1. 真机google浏览器测试

[root@room9pc01 ~]# http://192.168.4.5/index.php

1. proxy利用telnet验证memcached的session记录

[root@proxy ~]# telnet 192.168.4.5 11211

get aqr3jppb68lvmdhdotnnl5ipq3 #通过网页查看session id