jimmy609的专栏

个人资料

jimmy609

加关注 发私信

访问： 395455次
积分： 3885
等级： BLOG 5
排名： 第5380名

原创： 44篇 转载： 40篇
译文： 1篇 评论： 47条

文章搜索

文章存档

2015年05月 (1)
2015年02月 (3)
2015年01月 (1)
2014年11月 (2)
2014年10月 (3)

展开

阅读排行

oracle中导出导入表以及 (31031)
java、js中实现无限层级 (29603)
配置Tomcat使用https协 (23667)
java系统高并发解决方案 (20330)
java系统高并发解决方案 (18395)
springMVC乱码问题 (17451)
SqlServer导入bak文件以

转 java系统高并发解决方案（转载）

2014-07-15 10:32 20343人阅读 评论(1) 收藏 举报

转载博客地址：http://blog.csdn.net/zxl333/article/details/8454319

转载博客地址：http://blog.csdn.net/zxl333/article/details/8685157

一个小型的网站，比如个人网站，可以使用最简单的html静态页面就实现了，配合一些图片达到美化效果，所有的页面均存放在一个目录下，这样的网站对系统架构、性能的要求都很简单，随着互联网业务的不断丰富，网站相关的技术经过这些年的发展，已经细分到很细的方方面面，尤其对于大型网站来说，所采用的技术更是涉及面非常广，从硬件到软件、编程语言、数据库、WebServer、防火墙等各个领域都有了很高的要求，已经不是原来简单的html静态网站所能比拟的。

大型网站，比如门户网站。在面对大量用户访问、高并发请求方面，基本的解决方案集中在这样几个环节：使用高性能的服务器、高性能的数据库、高效率的编程语言、还有高性能的Web容器。但是除了这几个方面，还没法根本解决大型网站面临的高负载和高并发问题。

上面提供的几个解决思路在一定程度上也意味着更大的投入，并且这样的解决思路具备瓶颈，没有很好的扩展性，下面我从低成本、高性能和高扩张性的角度来说说我的一些经验。

1、HTML静态化

其实大家都知道，效率最高、消耗最小的就是纯静态化的html页面，所以我们尽可能使我们的网站上的页面采用静态页面来实现，这个最简单的方法其实也是最有效的方法。但是对于大量内容并且频繁更新的网站，我们无法全部手动去挨个实现，于是出现了我们常见的信息发布系统CMS，像我们常访问的各个门户站点的新闻频道，甚至他们的其他频道，都是通过信息发布系统来管理和实现的，信息发布系统可以实现最简单的信息录入自动生成静态页面，还能具备频道管理、权限管理、自动抓取等功能，对于一个大型网站来说，拥有一套高效、可管理的CMS是必不可少的。

除了门户和信息发布类型的网站，对于交互性要求很高的社区类型网站来说，尽可能的静态化也是提高性能的必要手段，将社区内的帖子、文章进行实时的静态化，有更新的

(15782)
Eclipse搭建android环境
(12687)
CSS中div覆盖另一个div
(12297)
iText和flying saucer结合
(11743)

评论排行

javaweb中使用百度、谷 (7)
mybatis、oracle批量插入 (5)
SpringMVC中servletFile (5)
java系统高并发解决方案 (4)
java中生成30万的excel (4)
Jquery控制My97DatePic (3)
java、js中实现无限层级 (2)
jdbc连接sqlserver报错ja (2)
iText和flying saucer结合 (2)
记录下mybatis中#{}和${ (2)

推荐文章

*Android RocooFix热修复框架
*笑谈Android图表------MPAndroidChart
*Nginx正反向代理、负载均衡等功能实现配置
*浅析ZeroMQ工作原理及其特点
*Android开源框架Universal-Image-Loader基本介绍及使用
*Spring Boot实践折腾记（三）：三板斧，Spring Boot下使用Mybatis

最新评论

javaweb中使用百度、谷歌地图
jimmy609: @s407272507:根据手机号码定位需要号码供应商开通白名单的，要么就用gps方式安装到手机去定...
javaweb中使用百度、谷歌地图
jimmy609: @zhoujingzhoujing1:需要一个key，自己申请
eclipse搭建android环境以及an
forgetue1: 棒棒棒！11111111
javaweb中使用百度、谷歌地图
u013038643: 挺厉害呀
IE下js里面new Date("2011-11-
qq_30661101: 楼主真乃神人也！
java系统高并发解决方案（转载）
helloworldsss: 负载均衡
java系统高并发解决方案之图片
helloworldsss: @u014034854:http://www.open-open.com/doc/view/6674...
java系统高并发解决方案之图片
helloworldsss: @xgujh:http://www.open-open.com/doc/view/66743eb4...
SpringMVC中servletFileUpload
u014771464: 如果是ssh2框架中呢？我试过网上流传的改web.xml中struts2相关配置得的/*为*.act...
mybatis、oracle批量插入配置
jimmy609: @li45874361:验证过得，

时候再重新静态化也是大量使用的策略，像Mop的大杂烩就是使用了这样的策略，网易社区等也是如此。

同时，html静态化也是某些缓存策略使用的手段，对于系统中频繁使用数据库查询但是内容更新很小的应用，可以考虑使用html静态化来实现，比如论坛中论坛的公用设置信息，这些信息目前的主流论坛都可以进行后台管理并且存储再数据库中，这些信息其实大量被前台程序调用，但是更新频率很小，可以考虑将这部分内容进行后台更新的时候进行静态化，这样避免了大量的数据库访问请求。

2、图片服务器分离

大家知道，对于Web服务器来说，不管是Apache、IIS还是其他容器，图片是最消耗资源的，于是我们有必要将图片与页面进行分离，这是基本上大型网站都会采用的策略，他们都有独立的图片服务器，甚至很多台图片服务器。这样的架构可以降低提供页面访问请求的服务器系统压力，并且可以保证系统不会因为图片问题而崩溃，在应用服务器和图片服务器上，可以进行不同的配置优化，比如apache在配置ContentType的时候可以尽量少支持，尽可能少的LoadModule，保证更高的系统消耗和执行效率。

3、数据库集群和库表散列

大型网站都有复杂的应用，这些应用必须使用数据库，那么在面对大量访问的时候，数据库的瓶颈很快就能显现出来，这时一台数据库将很快无法满足应用，于是我们需要使用数据库集群或者库表散列。

在数据库集群方面，很多数据库都有自己的解决方案，Oracle、Sybase等都有很好的方案，常用的MySQL提供的Master/Slave也是类似的方案，您使用了什么样的DB，就参考相应的解决方案来实施即可。

上面提到的数据库集群由于在架构、成本、扩张性方面都会受到所采用DB类型的限制，于是我们需要从应用程序的角度来考虑改善系统架构，库表散列是常用并且最有效的解决方案。我们在应用程序中安装业务和应用或者功能模块将数据库进行分离，不同的模块对应不同的数据库或者表，再按照一定的策略对某个页面或者功能进行更小的数据库散列，比如用户表，按照用户ID进行表散列，这样就能够低成本的提升系统的性能并且有很好的扩展性。sohu的论坛就是采用了这样的架构，将论坛的用户、设置、帖子等信息进行数据库分离，然后对帖子、用户按照板块和ID进行散列数据库和表，最终可以在配置文件中进行简单的配置便能让系统随时增加一台低成本的数据库进来补充系统性能。

4、缓存

缓存一词搞技术的都接触过，很多地方用到缓存。网站架构和网站开发中的缓存也是非常重要。这里先讲述最基本的两种缓存。高级和分布式的缓存在后面讲述。
架构方面的缓存，对Apache比较熟悉的人都能知道Apache提供了自己的缓存模块，也可以使用外加的Squid模块进行缓存，这两种方式均可以有效的提高Apache的访问响应能力。网站程序开发方面的缓存，Linux上提供的Memory Cache是常用的缓存接口，可以在web开发中使用，比如用Java开发的时候就可以调用MemoryCache对一些数据进行缓存和通讯共享，一些大型社区使用了这样的架构。另外，在使用web语言开发的时候，各种语言基本都有自己的缓存模块和方法，PHP有Pear的Cache模块，Java就更多了，.net不是很熟悉，相信也肯定有。

5、镜像

镜像是大型网站常采用的提高性能和数据安全性的方式，镜像的技术可以解决不同网络接入商和地域带来的用户访问速度差异，比如ChinaNet和EduNet之间的差异就促使了很多网站在教育网内搭建镜像站点，数据进行定时更新或者实时更新。在镜像的细节技术方面，这里不阐述太深，有很多专业的现成的解决架构和产品可选。也有廉价的通过软件实现的思路，比如Linux上的rsync等工具。

6、负载均衡

负载均衡将是大型网站解决高负荷访问和大量并发请求采用的终极解决办法。

负载均衡技术发展了多年，有很多专业的服务提供商和产品可以选择，我个人接触过一些解决方法，其中有两个架构可以给大家做参考。

1）硬件四层交换

第四层交换使用第三层和第四层信息包的报头信息，根据应用区间识别业务流，将整个区间段的业务流分配到合适的应用服务器进行处理。 第四层交换功能就象是虚IP，指向物理服务器。它传输的业务服从的协议多种多样，有HTTP、FTP、NFS、Telnet或其他协议。这些业务在物理服务器基础上，需要复杂的载量平衡算法。在IP世界，业务类型由终端TCP或UDP端口地址来决定，在第四层交换中的应用区间则由源端和终端IP地址、TCP

和UDP端口共同决定。

在硬件四层交换产品领域，有一些知名的产品可以选择，比如Alteon、F5等，这些产品很昂贵，但是物有所值，能够提供非常优秀的性能和很灵活的管理能力。Yahoo中国当初接近2000台服务器使用了三四台Alteon就搞定了。

2）软件四层交换

大家知道了硬件四层交换机的原理后，基于OSI模型来实现的软件四层交换也就应运而生，这样的解决方案实现的原理一致，不过性能稍差。但是满足一定量的压力还是游刃有余的，有人说软件实现方式其实更灵活，处理能力完全看你配置的熟悉能力。

软件四层交换我们可以使用Linux上常用的LVS来解决，LVS就是Linux Virtual Server，他提供了基于心跳线heartbeat的实时灾难应对解决方案，提高系统的鲁棒性，同时可供了灵活的虚拟VIP配置和管理功能，可以同时满足多种应用需求，这对于分布式的系统来说必不可少。

一个典型的使用负载均衡的策略就是，在软件或者硬件四层交换的基础上搭建squid集群，这种思路在很多大型网站包括搜索引擎上被采用，这样的架构低成本、高性能还有很强的扩张性，随时往架构里面增减节点都非常容易。这样的架构我准备空了专门详细整理一下和大家探讨。

一：高并发高负载类网站关注点之数据库

没错,首先是数据库,这是大多数应用所面临的首个SPOF。尤其是Web2.0的应用，数据库的响应是首先要解决的。
一般来说MySQL是最常用的，可能最初是一个mysql主机，当数据增加到100万以上，那么，MySQL的效能急剧下降。常用的优化措施是M-S（主-从）方式进行同步复制，将查询和操作分别在不同的服务器上进行操作。我推荐的是M-M-Slaves方式，2个主Mysql，多个Slaves，需要注意的是，虽然有2个Master，但是同时只有1个是Active，我们可以在一定时候切换。之所以用2个M，是保证M不会又成为系统的SPOF。
Slaves可以进一步负载均衡，可以结合LVS,从而将select操作适当的平衡到不同的slaves上。
以上架构可以抗衡到一定量的负载，但是随着用户进一步增加，你的用户表数据超过1千万，这时那个M变成了SPOF。你不能任意扩充Slaves，否则复制同步的开销将直线上升，怎么办？我的方法是表分区，从业务层面上进行分区。最简单的，以用户数据为例。根据一定的切分方式，比如id，切分到不同的数据库集群去。

全局数据库用于meta数据的查询。缺点是每次查询，会增加一次，比如你要查一个用户nightsailer,你首先要到全局数据库群找到nightsailer对应的cluster id，然后再到指定的cluster找到nightsailer的实际数据。
每个cluster可以用m-m方式，或者m-m-slaves方式。这是一个可以扩展的结构，随着负载的增加，你可以简单的增加新的mysql cluster进去。

需要注意的是：
1、禁用全部auto_increment的字段
2、id需要采用通用的算法集中分配
3、要具有比较好的方法来监控mysql主机的负载和服务的运行状态。如果你有30台以上的mysql数据库在跑就明白我的意思了。
4、不要使用持久性链接（不要用pconnect）,相反，使用sqlrelay这种第三方的数据库链接池，或者干脆自己做，因为php4中mysql的链接池经常出问题。

二：高并发高负载网站的系统架构之HTML静态化

其实大家都知道，效率最高、消耗最小的就是纯静态化 http://www.ablanxue.com/shtml/201207/776.shtml的html页面，所以我们尽可能使我们的网站上的页面采用静态页面来实现，这个最简单的方法其实也是 最有效的方法。但是对于大量内容并且频繁更新的网站，我们无法全部手动去挨个实现，于是出现了我们常见的信息发布系统CMS，像我们常访问的各个门户站点 的新闻频道，甚至他们的其他频道，都是通过信息发布系统来管理和实现的，信息发布系统可以实现最简单的信息录入自动生成静态页面，还能具备频道管理、权限 管理、自动抓取等功能，对于一个大型网站来说，拥有一套高效、可管理的CMS是必不可少的。

除了门户和信息发布类型的网站，对于交互性要求很高的社区类型网站来说，尽可能的静态化也是提高性能的必要手段，将社区内的帖子、文章进行实时的静态化，有更新的时候再重新静态化也是大量使用的策略，像Mop的大杂烩就是使用了这样的策略，网易社区等也是如此。

同时，html静态化也是某些缓存策略使用的手段，对于系统中频繁使用数据库查询但是内容更新很小的应用，可以考虑使用html静态化来实现，比如论坛 中论坛的公用设置信息，这些信息目前的主流论坛都可以进行后台管理并且存储再数据库中，这些信息其实大量被前台程序调用，但是更新频率很小，可以考虑将这 部分内容进行后台更新的时候进行静态化，这样避免了大量的数据库访问请求高并发。

网站HTML静态化解决方案
当一个Servlet资源请求到达WEB服务器之后我们会填充指定的JSP页面来响应请求:

HTTP请求---Web服务器---Servlet--业务逻辑处理--访问数据--填充JSP--响应请求

HTML静态化之后:

HTTP请求---Web服务器---Servlet--HTML--响应请求

静态访求如下

Servlet:

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    if(request.getParameter("chapterId") != null){
        String chapterFileName =
"bookChapterRead_"+request.getParameter("chapterId")+".html";
        String chapterFilePath = getServletContext().getRealPath("/") + chapterFileName;
        File chapterFile = new File(chapterFilePath);
        if(chapterFile.exists()){response.sendRedirect(chapterFileName);return;}//如果有这
个文件就告诉浏览器转向
        INovelChapterBiz novelChapterBiz = new NovelChapterBizImpl();
        NovelChapter novelChapter =
novelChapterBiz.searchNovelChapterById(Integer.parseInt(request.getParameter("chapterId
章节信息
        int lastPageId =
novelChapterBiz.searchLastCHapterId(novelChapter.getNovelId().getId(),
novelChapter.getId());
        int nextPageId =
novelChapterBiz.searchNextChapterId(novelChapter.getNovelId().getId(),
novelChapter.getId());
        request.setAttribute("novelChapter", novelChapter);
        request.setAttribute("lastPageId", lastPageId);
        request.setAttribute("nextPageId", nextPageId);
        new CreateStaticHTMLPage().createStaticHTMLPage(request, response,
getServletContext(),
                chapterFileName, chapterFilePath, "/bookRead.jsp");
    }
}
```

生成HTML静态页面的类:

```
package com.jb.y2t034.thefifth.web.servlet;
import java.io.ByteArrayOutputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpServletResponseWrapper;
/**
* 创建HTML静态页面
* 功能：创建HTML静态页面
* 时间：2009年1011日
* 地点：home
* @author mavk
*
```

```
*/
public class CreateStaticHTMLPage {
    /**
     * 生成静态HTML页面的方法
     * @param request 请求对象
     * @param response 响应对象
     * @param servletContext Servlet上下文
     * @param fileName 文件名称
     * @param fileFullPath 文件完整路径
     * @param jspPath 需要生成静态文件的JSP路径(相对即可)
     * @throws IOException
     * @throws ServletException
     */
    public void createStaticHTMLPage(HttpServletRequest request, HttpServletResponse
response,ServletContext servletContext,String fileName,String fileFullPath,String
jspPath) throws ServletException, IOException{
        response.setContentType("text/html;charset=gb2312");//设置HTML结果流编码(
即HTML文件编码)
        RequestDispatcher rd = servletContext.getRequestDispatcher(jspPath);//得到JSP
资源
        final ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream();//用于从ServletOutputStream中接收资源
        final ServletOutputStream servletOuputStream = new ServletOutputStream(){//用于
从HttpServletResponse中接收资源
            public void write(byte[] b, int off,int len){
                byteArrayOutputStream.write(b, off, len);
            }
            public void write(int b){
                byteArrayOutputStream.write(b);
            }
        };
        final PrintWriter printWriter = new PrintWriter(new
OutputStreamWriter(byteArrayOutputStream));//把转换字节流转换成字符流
        HttpServletResponse httpServletResponse = new
HttpServletResponseWrapper(response){//用于从response获取结果流资源(重写了两个
方法)
            public ServletOutputStream getOutputStream(){
                return servletOuputStream;
            }
            public PrintWriter getWriter(){
                return printWriter;
            }
        };
        rd.include(request, httpServletResponse);//发送结果流
        printWriter.flush();//刷新缓冲区，把缓冲区的数据输出
        FileOutputStream fileOutputStream = new FileOutputStream(fileFullPath);
        byteArrayOutputStream.writeTo(fileOutputStream);//把byteArrayOuputStream中的资
源全部写入到fileOuputStream中
        fileOutputStream.close();//关闭输出流，并释放相关资源
        response.sendRedirect(fileName);//发送指定文件流到客户端
    }
}
```

三：高并发高负载类网站关注点之缓存、负载均衡、存储

缓存是另一个大问题，我一般用memcached来做缓存集群，一般来说部署10台左右就差不多（10g内存池）。需要注意一点，千万不能用使用
swap，最好关闭linux的swap。

负载均衡/加速

可能上面说缓存的时候，有人第一想的是页面静态化，所谓的静态html，我认为这是常识，不属于要点了。页面的静态化随之带来的是静态服务的
负载均衡和加速。我认为Lighttped+Squid是最好的方式了。
LVS <------->lighttped====>squid(s) ====lighttpd

上面是我经常用的。注意，我没有用apache，除非特定的需求，否则我不部署apache，因为我一般用php-fastcgi配合lighttpd,
性能比apache+mod_php要强很多。

squid的使用可以解决文件的同步等等问题，但是需要注意，你要很好的监控缓存的命中率，尽可能的提高的90%以上。
squid和lighttped也有很多的话题要讨论，这里不赘述。

存储
存储也是一个大问题，一种是小文件的存储，比如图片这类。另一种是大文件的存储，比如搜索引擎的索引，一般单文件都超过2g以上。
小文件的存储最简单的方法是结合lighttpd来进行分布。或者干脆使用Redhat的GFS，优点是应用透明，缺点是费用较高。我是指
你购买盘阵的问题。我的项目中，存储量是2-10Tb，我采用了分布式存储。这里要解决文件的复制和冗余。
这样每个文件有不同的冗余，这方面可以参考google的gfs的论文。
大文件的存储，可以参考nutch的方案，现在已经独立为hadoop子项目。(你可以google it)

其他：
此外，passport等也是考虑的，不过都属于比较简单的了。
四：高并发高负载网站的系统架构之图片服务器分离
大家知道，对于Web 服务器来说，不管是Apache、IIS还是其他容器，图片是最消耗资源的，于是我们有必要将图片与页面进行分离，这是基本上大型网站都会采用的策略，他们都有独立的图片服务器，甚至很多台图片服务器。这样的架构可以降低提供页面访问请求的服务器系统压力，并且可以保证系统不会因为图片问题而崩溃，在应用 服务器和图片服务器上，可以进行不同的配置优化，比如apache在配置ContentType的时候可以尽量少支持，尽可能少的LoadModule， 保证更高的系统消耗和执行效率。

利用Apache实现图片服务器的分离
缘由：
起步阶段的应用，都可能部署在一台服务器上（费用上的原因）
第一个优先分离的，肯定是数据库和应用服务器。
第二个分离的，会是什么呢？各有各的考虑，我所在的项目组重点考虑的节约带宽，服务器性能再好，带宽再高，并发来了，也容易撑不住。因此，我这篇文章的重点在这里。这里重点是介绍实践，不一定符合所有情况，供看者参考吧，
环境介绍：
WEB应用服务器：4CPU双核2G, 内存4G
 部署：Win2003/Apache Http Server 2.1/Tomcat6
数据库服务器：4CPU双核2G, 内存4G
 部署：Win2003/MSSQL2000
步骤：
步骤一：增加2台配置为：2CPU双核2G，内存2G普通服务器，做资源服务器
 部署：Tomcat6，跑了一个图片上传的简单应用，（记得指定web.xml的<distributable/>），并指定域名为res1.***.com,res2.***.com，采用ajp协议
步骤二：修改Apache httpd.conf配置
 原来应用的文件上传功能网址为：
  1、/fileupload.html
  2、/otherupload.html
 在httpd.conf中增加如下配置

```
<VirtualHost *:80>
 ServerAdmin webmaster@***.com
 ProxyPass /fileupload.html balancer://rescluster/fileupload lbmethod=byrequests stickysession=JSESSIONID nofailover=Off timeout=5 maxattempts=3
 ProxyPass /otherupload.html balancer://rescluster/otherupload.html lbmethod=byrequests stickysession=JSESSIONID nofailover=Off timeout=5 maxattempts=3
 #<!--负载均衡-->
 <Proxy balancer://rescluster/>
   BalancerMember ajp://res1.***.com:8009 smax=5 max=500 ttl=120 retry=300 loadfactor=100 route=tomcat1
   BalancerMember ajp://res2.***.com:8009 smax=5 max=500 ttl=120 retry=300 loadfactor=100 route=tomcat2
 </Proxy>

< /VirtualHost>
```

步骤三，修改业务逻辑：
所有上传文件在数据库中均采用全url的方式保存，例如产品图片路径存成：http://res1.***.com/upload/20090101/product120302005.jpg

现在，你可以高枕无忧了，带宽不够时，增加个几十台图片服务器，只需要稍微修改一下apache的配置文件，即可。

五：高并发高负载网站的系统架构之数据库集群和库表散列

大型网站都有复杂的应用，这些应用必须使用数据库，那么在面对大量访问的时候，数据库的瓶颈很快就能显现出来，这时一台数据库将很快无法满足应用，于是我们需要使用数据库集群或者库表散列。

在数据库集群方面，很多数据库都有自己的解决方案，Oracle、Sybase等都有很好的方案，常用的MySQL提供的Master/Slave也是类似的方案，您使用了什么样的DB，就参考相应的解决方案来实施即可。

上面提到的数据库集群由于在架构、成本、扩张性方面都会受到所采用DB类型的限制，于是我们需要从应用程序的角度来考虑改善系统架构，库表散列是常用并 且最有效的解决方案。我们在应用程序中安装业务和应用或者功能模块将数据库进行分离，不同的模块对应不同的数据库或者表，再按照一定的策略对某个页面或者 功能进行更小的数据库散列，比如用户表，按照用户ID进行表散列，这样就能够低成本的提升系统的性能并且有很好的扩展性。sohu的论坛就是采用了这样的 架构，将论坛的用户、设置、帖子等信息进行数据库分离，然后对帖子、用户按照板块和ID进行散列数据库和表，最终可以在配置文件中进行简单的配置便能让系 统随时增加一台低成本的数据库进来补充系统性能。

集群软件的分类：
一般来讲，集群软件根据侧重的方向和试图解决的问题，分为三大类：高性能集群（High performance cluster，HPC）、负载均衡集群（Load balance cluster， LBC），高可用性集群（High availability cluster，HAC）。
高性能集群（High performance cluster，HPC），它是利用一个集群中的多台机器共同完成同一件任务，使得完成任务的速度和可靠性都远远高于单机运行的效果。弥补了单机性能上的不足。该集群在天气预报、环境监控等数据量大，计算复杂的环境中应用比较多；
负载均衡集群（Load balance cluster， LBC），它是利用一个集群中的多台单机，完成许多并行的小的工作。一般情况下，如果一个应用使用的人多了，那么用户请求的响应时间就会增大，机器的性能也会受到影响，如果使用负载均衡集群，那么集群中任意一台机器都能响应用户的请求，这样集群就会在用户发出服务请求之后，选择当时负载最小，能够提供最好的服务的这台机器来接受请求并相应，这样就可用用集群来增加系统的可用性和稳定性。这类集群在网站中使用较多；
高可用性集群（High availability cluster，HAC），它是利用集群中系统 的冗余，当系统中某台机器发生损坏的时候，其他后备的机器可以迅速的接替它来启动服务，等待故障机的维修和返回。最大限度的保证集群中服务的可用性。这类系统一般在银行，电信服务这类对系统可靠性有高的要求的领域有着广泛的应用。
2 数据库集群的现状
数据库集群是将计算机集群技术引入到数据库中来实现的，尽管各厂商宣称自己的架构如何的完美，但是始终不能改变Oracle当先，大家追逐的事实，在集群的解决方案上Oracle RAC还是领先于包括微软在内的其它数据库厂商，它能满足客户高可用性、高性能、数据库负载均衡和方便扩展的需求。
Oracle’s Real Application Cluster (RAC)
Microsoft SQL Cluster Server (MSCS)
IBM’s DB2 UDB High Availability Cluster(UDB)
Sybase ASE High Availability Cluster (ASE)
MySQL High Availability Cluster (MySQL CS)
基于IO的第三方HA(高可用性)集群
当前主要的数据库集群技术有以上六大类，有数据库厂商自己开发的；也有第三方的集群公司开发的；还有数据库厂商与第三方集群公司合作开发的，各类集群实现的功能及架构也不尽相同。
RAC（Real Application Cluster，真正应用集群）是Oracle9i数据库中采用的一项新技术，也是Oracle数据库支持网格计算环境的核心技术。它的出现解决了传统数据库应用中面临的一个重要问题：高性能、高可伸缩性与低价格之间的矛盾。在很长一段时间里，甲骨文都以其实时应用集群技术(Real Application Cluster，RAC)统治着集群数据库市场

六：高并发高负载网站的系统架构之缓存

缓存一词搞技术的都接触过，很多地方用到缓存。网站架构和网站开发中的缓存也是非常重要。这里先讲述最基本的两种缓存。高级和分布式的缓存在后面讲述。
架构方面的缓存，对Apache比较熟悉的人都能知道Apache提供了自己的缓存模块，

也可以使用外加的Squid模块进行缓存，这两种方式均可以有效的提高Apache的访问响应能力。

网站程序开发方面的缓存，Linux上提供的Memory Cache是常用的缓存接口，可以在web开发中使用，比如用Java开发的时候就可以调用MemoryCache对一些数据进行缓存和通讯共享，一些大 型社区使用了这样的架构。另外，在使用web语言开发的时候，各种语言基本都有自己的缓存模块和方法，PHP有Pear的Cache模块，Java就更多 了，.net不是很熟悉，相信也肯定有。

Java开源缓存框架

JBossCache/TreeCache JBossCache是一个复制的事务处理缓存，它允许你缓存企业级应用数据来更好的改善性能。缓存数据被自动复制，让你轻松进行Jboss服务器之间的集群工作。JBossCache能够通过Jboss应用服务或其他J2EE容器来运行一个Mbean服务，当然，它也能独立运行。 JBossCache包括两个模块：TreeCache和TreeCacheAOP。TreeCache --是一个树形结构复制的事务处理缓存。 TreeCacheAOP --是一个“面向对象”缓存，它使用AOP来动态管理POJO

OSCache OSCache标记库由OpenSymphony设计，它是一种开创性的JSP定制标记应用，提供了在现有JSP页面之内实现快速内存缓冲的功能。OSCache是个一个广泛采用的高性能的J2EE缓存框架，OSCache能用于任何Java应用程序的普通的缓存解决方案。OSCache有以下特点：缓存任何对象，你可以不受限制的缓存部分jsp页面或HTTP请求，任何java对象都可以缓存。 拥有全面的API--OSCache API给你全面的程序来控制所有的OSCache特性。 永久缓存--缓存能随意的写入硬盘，因此允许昂贵的创建（expensive-to-create）数据来保持缓存，甚至能让应用重启。 支持集群--集群缓存数据能被单个的进行参数配置，不需要修改代码。 缓存记录的过期--你可以有最大限度的控制缓存对象的过期，包括可插入式的刷新策略（如果默认性能不需要时）。

JCACHE JCACHE是一种即将公布的标准规范（JSR 107），说明了一种对Java对象临时在内存中进行缓存的方法，包括对象的创建、共享访问、假脱机（spooling）、失效、各JVM的一致性等。它可被用于缓存JSP内最经常读取的数据，如产品目录和价格列表。利用JCACHE，多数查询的反应时间会因为有缓存的数据而加快（内部测试表明反应时间大约快15倍）。

Ehcache Ehcache出自Hibernate，在Hibernate中使用它作为数据缓存的解决方案。

Java Caching System JCS是Jakarta的项目Turbine的子项目。它是一个复合式的缓冲工具。可以将对象缓冲到内存、硬盘。具有缓冲对象时间过期设定。还可以通过JCS构建具有缓冲的分布式构架，以实现高性能的应用。 对于一些需要频繁访问而每访问一次都非常消耗资源的对象，可以临时存放在缓冲区中，这样可以提高服务的性能。而JCS正是一个很好的缓冲工具。缓冲工具对于读操作远远多于写操作的应用性能提高非常显著。

SwarmCache SwarmCache是一个简单而功能强大的分布式缓存机制。它使用IP组播来有效地在缓存的实例之间进行通信。它是快速提高集群式Web应用程序的性能的理想选择。

ShiftOne ShiftOne Object Cache这个Java库提供了基本的对象缓存能力。实现的策略有先进先出（FIFO），最近使用（LRU），最不常使用（LFU）。所有的策略可以最大化元素的大小，最大化其生存时间。

WhirlyCache Whirlycache是一个快速的、可配置的、存在于内存中的对象的缓存。它能够通过缓存对象来加快网站或应用程序的速度，否则就必须通过查询数据库或其他代价较高的处理程序来建立。

Jofti Jofti可对在缓存层中(支持EHCache，JBossCache和OSCache)的对象或在支持Map接口的存储结构中的对象进行索引与搜索。这个框架还为对象在索引中的增删改提供透明的功能同样也为搜索提供易于使用的查询功能。

cache4j cache4j是一个有简单API与实现快速的Java对象缓存。它的特性包括：在内存中进行缓存，设计用于多线程环境，两种实现：同步与阻塞，多种缓存清除策略：LFU, LRU, FIFO，可使用强引用(strong reference)与软引用(soft reference)存储对象。

Open Terracotta 一个JVM级的开源群集框架，提供：HTTP Session复制，分布式缓存，POJO群集，跨越群集的JVM来实现分布式应用程序协调(采用代码注入的方式，所以你不需要修改任何)。

sccache SHOP.COM使用的对象缓存系统。sccache是一个in-process cache和二级、共享缓存。它将缓存对象存储到磁盘上。支持关联Key，任意大小的Key和任意大小的数据。能够自动进行垃圾收集。

Shoal Shoal是一个基于Java可扩展的动态集群框架，能够为构建容错、可靠和可用的Java应用程序提供了基础架构支持。这个框架还可以集成到不希望绑定到特定通信协议，但需要集群和分布式系统支持的任何Java产品中。Shoal是GlassFish和JonAS应用服务器的集群引擎。

Simple-Spring-Memcached Simple-Spring-Memcached，它封装了对MemCached的调用，使MemCached的客户端开发变得超乎寻常的简单。

顶 8 踩 0

上一篇 TOMCAT服务器配置域名
下一篇 java中生成静态html（转载）

猜你在找

- MySQL数据库管理
- Qt网络编程实战之HTTP服务器
- 高并发集群架构超细精讲
- Linux运维高薪入门及进阶全新经典视频-老男孩Linux第
- 韦东山嵌入式Linux第一期视频
- java系统高并发解决方案之图片服务器分离
- java系统高并发解决方案之图片服务器分离
- java系统高并发解决方案
- java系统高并发解决方案之图片服务器分离
- java系统高并发解决方案

查看评论

1楼 helloworldsss 2016-05-07 21:28发表

负载均衡

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery BI
HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity Splashtop
UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack FTC
OPhone CouchBase 云计算 iOS6 Rackspace Web
大数据 aptech Perl Tornado Ruby Hibernate Thi
Cloud Foundry Redis Scala Django Bootstrap