数据库存取图片的优化算法

张祖莲[1],李景林[1],王命全[1],李少雪[2],任宏宇[3]
(1.新疆气象局 新疆兴农网信息中心,新疆 乌鲁木齐 830002;
2.新疆维吾尔自治区人大常委会办公厅,新疆 乌鲁木齐 830002;
3.河南省许昌市 94537 部队司令部,河南 许昌 461101)

摘 要:通过对数据库如何存取图片的研究,提出在.Net平台下基于Web的SQL Server数据库存取图片的优化算法,该算法能有效优化图片大小,节省数据库容量,不占用服务器空间,减少服务器的响应时间,提高网页中的图片加载速度。

关键词:数据库;优化图片;存取图片;网页打开速度;数据库容量
中图分类号:TP311 **文献标识码**:A **文章编号**:1674-7720(2011)21-0044-03

Optimized algorithm of database store and read the picture

Zhang Zulian[1], Li Jinglin[1], Wang Mingquan[1], Li Shaoxue[2], Ren Hongyu[3]
(1.Information Center of Xinjiang Develop Agriculture Net, Xinjiang Weather Bureau, Urumqi 830002, China;
2.Xinjiang Uygur Minority Autonomous Region Office of the NPC Standing Committee, Urumqi 830002, China;
3.Henan Province Xuchang City 94537 Command Headquarters, Xuchang 461101, China)

Abstract: Through the search of the database store and read the picture, the paper provides an optimized algorithm of SQL Server database store and read the picture based on Web in the .NET platform. The algorithm can optimize pictures size, reduce database capacity, not take up server space, reduce server run-time, raise the pictures apply to carry velocity of the Web pages, effectively speed up the velocity of opening Web pages.

Key words: database; optimize the picture; store and read the picture; velocity of openning the web pages; database capacity

互联网上几乎所有网页都由图片和文字组成,许多大型网站都涉及图片管理功能的实现。在网络环境中,由于图片文件相对较小,因此比音频和视频文件更加便于传输。在很多应用领域中出现了对专用的网络环境下图片数据库的需求,即利用数据库平台实现图片大量的集中存储,同时以网络为连接,通过基于HTTP协议的网络浏览器实现对数据库(包括图片数据)的远程访问[1]。

网页中图片的加载速度直接影响该网页打开速度。访问者认为,打开速度较快的网站质量更高、更可信、也更有趣;网页打开速度越慢,访问者的心理挫折感就越强,就会对网站的可信性和质量产生怀疑。

基于以上情况,如何有效提高网站页面打开速度,是每个网站管理者和开发人员非常关注和必须解决的问题。本文通过对数据库如何存取图片的研究,提出了一种优化算法。

1 图片存取的方法分析

目前网页上的图片存储一般有两种方法:

(1)将图片以独立文件的形式存放至服务器中,同时在数据库对应表中的文本类型字段中存放该图片的文件路径[2]。这种方法具有缩小数据表,提高数据库相关操作速度且容易编程实现的优点,但是却造成了图片及其描述信息的分离,如果数据过期或不用而删掉,对应图片没有删掉,仍占着服务器空间[1]。

(2)将图片直接放入数据库对应表的image数据类型字段中,使得图片和数据描述成为一个整体[3]。这种方法存放的图片格式很灵活,管理很方便,而且安全性也很好,最重要的是,如果过期或不用的数据删掉,对应的所有图片及相关信息也都删掉,不再占用服务空间。但是如果图片越大,占用数据库存储空间也越大,网页打

开加载图片也就越慢[1]。

在目前的研究中，采用第二种方法存取图片时，基于移动设置优化 PNG 图片，在 PNG 图片的数据结构中，部分数据块是可选的，剔除可选数据段，对 PNG 进行适当的处理，可以有效减少占用空间，优化移动设备上的图片应用。图片处理前后的大小相差很多，结果很不稳定[4]。

本文采用第二种存取图片的方法，针对图片占用数据库存储空间过大、网页加载图片过慢的问题，提出了优化图片算法。

2 建数据库建表

本实验采用转换为二进制的方式将图片文件的全部数据存储到 image[5-6]字段中，使用时再通过特定的过程将二进制信息转换成原来格式的文件内容，从而真正地实现数据的整体存储，避免了平台更换和数据移植时可能带来的记录失效问题。

(1) 在数据库 SQL 2005 中建立 testimgbig 和 testimgsmall 两个数据库；

(2) 在两个数据库中分别建表 imgbig 和 imgsmall，如表 1 和表 2 所示；

表 1 imgbig 表

列名	数据类型	允许空
id	int	
imgbig	image	允许

表 2 imgsmall 表

列名	数据类型	允许空
id	int	
imgsmal	image	允许

(3)分别在两个数据库中建立Addimgbig 和 Addimgsmall 两个存储过程。

3 实验过程

本实验采用 C# 编程语言，Visual Studio 2005 开发工具，Microsoft SQL Server 2005 数据库以及 IIS5.0，在服务器的网站根目录下建立“tmpimages”及两个子文件夹“img1”和“img2”。

3.1 数据库存取图片流程

(1)未优化图片存入数据库流程。将本地图片上传到服务器“tmpimages/ img1”文件夹下，在服务器端将上传的图片转化为字符流存入数据库；

(2)优化图片存入数据库流程。本地图片上传到服务器“tmpimages/ img1”文件夹下，在服务器端将上传的图片进行优化处理，然后将图片存到服务器“tmpimages/ img2”文件夹下，同时将“img2”中刚优化的图片转化为字符流存入数据库；

(3)读取图片流程。连接图片所在数据库，找到此图片表名及字段，然后将对应图片 ID 号的字符流转化为可视图片。

3.2 相关核心代码

(1)将图片存入数据库的相关核心代码

```
SqlConnection conn=new SqlConnection     //连接数据库;
SqlCommand cmd=conn.CreateCommand();
cmd.CommandType=CommandType.StoredProcedure;
                                          //调用存储过程
cmd.CommandText="Addimgbig";
string fullfileName=this.FileUpload1 .FileName;
                                          //获取路径名
if(fullfileName.Trim().Length ! =0)      //路径名不能为空
{
  string typebig=fullfileName.Substring(fullfileName.
                                  LastIndexOf(".")+1);
  Random rad=new Random();
  string RanNum=Convert.ToString(rad.Next());
                                          //获取随机数
  string dt=年月日时分秒+6 位随机数;
                                  //防止图片重名，给图片重命名
  string fileName = "";
  if (fullfileName ! = "")                //为了防止用户
                  不添加图片时，保存文件的语句出现问题
  {
fileName=dt+fullfileName.Substring(fullfileName.LastIndexOf
(".")); //获取图片名称
  }
  string url1=@"tmpimages\\img1"+"\\"+fileName;
                                  //定数据库中字段 url
  if(typebig ! ="")
  {

  this.FileUpload1 .SaveAs(Server.MapPath("tmpimages
                                  \\img1")+"\\"+fileName);
  pathNamebig1=Server.MapPath(url1);
  System.IO.FileStream fs=new System.IO.FileStream
                  (pathNamebig1, System.IO.FileMode.Open);
       byte[]imagebig=new byte[fs.Length];
       BinaryReader br=new BinaryReader(fs);
       imagebig=br.ReadBytes(Convert.ToInt32(fs.Length));
       cmd.Parameters.AddWithValue("@imgbig", imagebig);
     }
}
conn.Open();                              //打开数据库
cmd.ExecuteNonQuery();                    //执行数据库
conn.Close();                             //关闭数据库
```

(2)优化图片的相关核心代码

```
System.Drawing.Image originalImage=
                  System.Drawing.Image.FromFile(filePath);
int ow=originalImage.Width;
int oh=originalImage.Height;
int towidth=220;
int toheight=220;                         //定义优化图片宽高
                  也可以只定义宽或高，动态按比例缩小
//从文件取得图片对象
```

```
System.Drawing.Image image=System.Drawing.Image.
                                FromFile(filePath, true);
//取得图片大小
System.Drawing.Size size=new System.Drawing.
                        Size((int)image.Width,(int)image.Height);
//新建一个 bmp 图片
System.Drawing.Image bitmap=new System.Drawing.Bitmap
(towidth,toheight);System.Drawing.Graphics g=System.Drawing.
Graphics.FromImage(bitmap);                    //新建一个画板
//设置高质量插值法
g.InterpolationMode =System.Drawing.Drawing2D.
InterpolationMode.Default;
//设置高质量,低速度呈现平滑程度
g.SmoothingMode =System.Drawing.Drawing2D.SmoothingMode.
Default;
g.Clear(System.Drawing.Color.White);      //清空一下画布
//在指定位置画图
g.DrawImage (image,new System.Drawing.Rectangle(0,0,
bitmap.Width,bitmap.Height),new System.Drawing.Rectangle (0,
0,image.Width,image.Height),System.Drawing.GraphicsUnit.Pixel);
 if(File.Exists(saveImg))File.Delete(saveImg);//删除已有文件
//保存高清晰度的缩略图
bitmap.Save (saveImg.ToString (),System.Drawing.Imaging.
ImageFormat.Jpeg);
g.Dispose();
image.Dispose();
bitmap.Dispose();
```

(3)读取图片的相关核心代码

```
SqlConnection conn=new SqlConnection ();//连接数据库
string sql="select imgbig from imgbig where id='"+
imgid+"'";
SqlCommand command=new SqlCommand(sql,conn);
conn.Open();                                    //打开数据库
SqlDataReader dr=command.ExecuteReader();
while(dr.Read())
  {
   Response.BinaryWrite((byte[])dr[0]);      //读取数据流
  }
Response.Write("");
conn.Close();                                   //关闭数据库
```

4 实验数据

页面加载时间测试方法：用秒表在本地机上测试。测试未优化和优化后各 20 次，去掉差距较大的 4 个，然后取平均数。

图片选用使用数码相机拍摄的未处理的图片（主要是为了更容易测试算法有效性），上传的所有图片都是同一张图片，以便于比较。

进行数据库备份。当备份到 1、10、20 张图片时，分别备份一次。

实验结果如表 3 所示，部分数据如图 1~图 4 所示，测试页面如图 5 所示。

表 3 实验数据

类型	图片大小	页面加载时间/s	备份数据库大小/MB	图片数量
未优化	3.37 MB	0.071	4.90	1 张
优化后	5.80 KB	0.030	1.45	
未优化	3.37 MB×10	5.051	36.1	10 张
优化后	5.80 KB×10	0.051	1.51	
未优化	3.37 MB×20	10.012	71.1	20 张
优化后	5.80 KB×20	0.081	1.58	

图 1 未优化图片

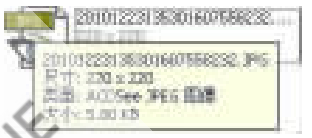

图 2 优化后图片

图 3 未优化 1 张图片数据库备份

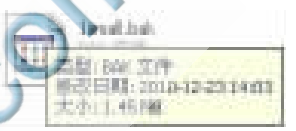

图 4 优化后 1 张数据库备份

图 5 测试页面

在测试页面中，未优化图片和优化图片的显示从视觉上效果一样，并没有出现因为压缩而丢失像素模糊现象。

由以上数据可以看出，本文提出的存取图片优化算法是一种有效的图片压缩、优化方法，同时能节省数据库容量，明显提高了网页加载图片速度。在服务器的“img1”和“img2”文件夹中的图片可以定期删除掉，不占用服务器空间。

本文测试是在本机上进行的，用了同一张原图片测试，明显看出页面加载时间的差距和数据库的容量。大型网站有更多图片，如果采用优化算法，将大大减少页面加载时间和数据库容量，同时不占用服务器空间，有效提高网页打开速度。

本算法在基于 Web 网站开发中对图片处理具有很高的实用价值，用 C#、Java 语言开发网站都可以借鉴此算法。本文只是在本机上用该算法对少量图片进行测

试，当有大数据量时，该算法效果将更加明显。虽然图片的像素降低从视觉效果上差别不大，但如果对图像精确度要求特别高还有待考虑，该算法还有待不断改进和优化。

参考文献

[1] 李伟民，何伟，李平.基于 Web 的 SQLServer 数据库存取图片的 Delphi 实现 [J]. 计算机工程与设计，2007，28(12)：2943-2945.

[2] 战仁军，张明书.图像文件在数据库中的存取[J].西安工程科技学院学报，2003，17(4)：369-372.

[3] 张永仁，黄科军，李德孝.基于数据库的文件管理[J].计算机工程与设计，2006，27(11)：2044-2045.

[4] 徐逸卿，刘林.基于移动设备的 PNG 图片优化实现[J].多媒体技术及应用，2008，3(9)：2070-2075.

[5] 郭东青，李佳，刘彬彬.数据库创建、数据仓库与优化[M].北京：清华大学出版社，2001.

[6] 古凌风.用 ADO 技术实现数据库图像字段的存取[J].计算机工程与设计，2004，25(8)：1388-1392.

(收稿日期：2011-08-19)

作者简介：

张祖莲，女，1984 年生，硕士，主要研究方向：网络信息检索。

李景林，男，1957 年生，高级工程师，主要研究方向：气象信息服务，农业信息服务。

王命全，男，1986 年生，硕士，主要研究方向：网格计算。