DOI:10.13612/j.cnki.cntp.2009.24.169

ASP.NET 中文件上传到数据库与上传到文件系统的比较

卢宇星
(中国神华铁路货车运输分公司)

摘 要:文件上传和下载浏览是动态网站的一项重要的功能,那么,文件上传到服务器的文件系统中还是保存到数据库中?本文从两者编程过程中的相同点和不同点进行了对照,并从事务支持、完整性、安全性和性能方面等做了分析和比较。
关键词:ASP.net 文件上传 数据库存储与文件系统存储比较

目前,公司中文件的上传到服务器和文件浏览是日常的工作,因而对于公司来说,文件上传系统是信息传达的重要部分。在实现此功能的基础上,上传的文件是保存在数据库中还是保存在文件系统中,是编程人员总会遇到的问题。下面就关于这两方面进行比较。

对于上传的文件保存到数据库中就是在服务器中先创建数据库文件,然后将上传的文件以二进制数据形式保存到数据库中对应的二进制的字段中;而将上传的文件保存到文件系统是以文件的形式将上传文件保存到服务器对应的文件夹中。下面就两种方法比较异同点。

1 两种方法中的相同点

在两种方法中,都要使用 fileupload 控件,该控件是表示一个 HTML 文件输入,负责接收上传的文件。其控件代码如下:

<asp:FileUpload ID = "fileuploader" runat="server" />。

两种方法都要使用数据库,建立数据库文件。用数据库系统保存上传的文件,其中的数据库是用来保存上传的文件名及文件内容;用文件系统保存上传的文件,其中的数据库是用来保存文件名和文件保存的路径。

2 两种方法的不同点

上传的方法不同。将文件上传到服务器的文件系统中,是直接使用 fileupload 控件的 SaveAs (filepath as string, filename as string)方法,把已上传的文件保存在 filename 所指定的系统文件夹中,其代码如下:

```
Protected void btnadd_Click (object sender, Eventargs e )
{ if (fileuploader.HasFile)
{ fileuploader.SaveAs(Path.Combine(Server.MapPath("./"),fileuploader.FileName)); }
}
```

将文件保存到数据库中,使用的是 fileupload 控件的 FileBytes 的属性,其类型是 byte(),它是字节数组,包含 fileupload 控件中上传文件的二进制内容,具体的操作是将文件的二进制数据保存到数据库记录对应的字段中。其代码如下 (编程代码 1):

```
<asp:SqlDataSource
        id="scrfiles"
        ConnectionString="<%$ ConnectionStrings:ccConnectionString1 %>"
        insertcommand = "insert filecontent (filename,filebytes) values (@filename,@filebytes)"
        runat="server">
        <InsertParameters>
            <asp:ControlParameter Name = "filename" ControlID = "fileuploader" PropertyName="filename" />
            <asp:ControlParameter Name = "filebytes" ControlID = " fileuploader" PropertyName = " FileBytes" />
        </InsertParameters>
</asp:SqlDataSource>
```

数据库中保存的内容不同。文件上传到服务器的文件系统中,后台数据库保存的内容为文件名称和文件保存的路径。

编写程序如下:

```
private void setintoku()
    {
        SqlConnection con = new SqlConnection ("server = .\\SQLEXPRESS;database = databasename;uid=username;pwd=password");
        SqlCommand cmd = new SqlCommand("insert into files (filename,filepath) values(@filename,@filepath)", con);
        cmd.Parameters.AddWithValue ("@filemame", fileuploader.FileName);
        cmd.Parameters.AddWithValue ("@filepath", Path.Combine(Server.MapPath("./"), fileuploader.FileName));
        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();
}
```

文件上传到数据库系统中,后台数据库保存的内容为文件名称和文件内容的二进制数据。其编程参照编程代码 1.

建立数据库的结构不同。文件上传到服务器的文件系统中,后台数据库表 files 定义如下:

字段名	数据类型
Id	Internet (indentity)
Filename	nvarchar(50)
Filepath	nvarchar(50)

文件上传到数据库中,后台数据库表 filecontent 的定义如下:

字段名	数据类型
Id	Internet (indentity)
Filename	nvarchar (50)
Filebyte	varbinary(MAX)

用户浏览 (获取) 文件的途径不同。文件存储在文件系统中,由于文件的路径是存放在数据库中,因而,将数据库中的路径字段值取出,做为浏览文件的超级链接地址,IIS 即

注意分析:(1) 如果目前机组的机械制动性能良好,风闸灵活,没有发卡现象,开停机次数不多,并且推力轴承已经使用或准备更改成塑料瓦,不必安装电气制动系统。(2)机械制动系统性能良好,推力轴承原来使用的巴氏合金瓦性能稳定,不计划更换成塑料瓦,若加闸转速比较高,粉尘污染较大,建议安装电气制动系统。(3)如果机械制动性能不好,风闸经常发卡,正准备更换风闸,这种情况,应优先考虑安装电气制动系统。安装一套电气制动系统投资为 20-60 万元(不同的容量有所不同),比更换一套风闸的价格还要低,因此不管是从技术先进性还是从经济性来说都应优先考虑安装电气制动系统。

系统的安全使用,不仅对系统自身可靠性进行检测,而且对运行人员要进行技术培训,必须要求熟悉操作流程,职业技术水平较高。

3 结束语

本文在提出发电机定子三相短路电气制动、发电机定子三相短路电气制动和反接制动停机各自的优缺点的基础上引出了水轮发电机的电气制动并介绍了电气制动的工作原理。同时通过实验着重分析了几种电气制动方法的特点及其应用场合,深入探讨了发一变组高压短路制动和中小机组反接制动的可行性、实用性,为水电厂选择合理的电气制动方式提供了依据,最后并对电气制动的应用提出几点建议。

参考文献

[1]王定一,水电站控制技术,武汉:华中理工大学出版社,1988.
[2]楼永仁,黄声先,李植鑫,水电站自动化,水利电力出版社,1995.
[3]刘忠源等,水电站自动化,北京:水利电力出版社,1986.
[4]张红,王辑祥,水轮发电机组电气制动方法的研究与探讨,广西水利水电,2004.3,13-16 页.
[5]梁勇,宋自灵,黄海琪等,电气制动停机技术在我区水轮发电机组上的应用,广西电力技术,2001.1,43-44 页.
[6]肖宾,肖洁,大中型水电机组电气制动选型与设计,贵州水力发电,2000.12,68-70 页.

可将指定路径的文件显示在用户端的浏览器中，其代码如下：

```
<asp:HyperLinkField DataNavigateUrlFields = "filepath" DataTextField = "filename" HeaderText=" 文件标题" SortExpression="filename" DataNavigateUrlFormatString="~/{0}" >
```

；其中 filepath 为数据库中指定记录的文件路径字段的值。

文件存储在数据库中，用户要浏览文件，必须将保存在数据库中的二进制文件读取出，显示在用户端的浏览器中，这就需要编写专门的文件下载程序，因为 IIS 不能自动从数据库中读取文件。其代码如下：

```
<asp:HyperLink id="lnkfile" text=´<%#Eval ("filename")%>´ navigateurl=´<%#Eval("id","~/FileHandler.ashx?id={0}")%>´
runat="server" />
```

，其中链接地址采用调用参数传递的方法，将数据库中对应记录的 id 号作为传递参数，并编写 HTTP 处理器程序 FileHandler.ashx，将超级链接中的参数 ID 指向的记录的文件内容字段值，以二进制数据的格式回写到浏览器中。其中调用 HTTP 处理器程序 FileHandler.ashx，其代码如下：

```
<%
@ WebHandler Language="C#" Class="FileHandler" %>
using System.Text;
using System;
using System.IO;
using System.Web;
using System.Data;
using System.Data.SqlClient;
public class FileHandler :IHttpHandler{
const string ConString = "server=.\\SQLEXPRESS;database =databasename;uid =username;pwd=password";

public void ProcessRequest (HttpContext context) {
context.Response.ContentType = "application/msword";
SqlConnection con = new SqlConnection(ConString);
SqlCommand cmd = new SqlCommand("select filebytes from filecontent where Id=@Id", con);
cmd.Parameters.AddWithValue ("@Id",context.Request["id"]);
using(con)
{
con.Open();
byte [] file =(byte [])cmd.ExecuteScalar();
if (file==null)
{ context.Response.Write ("wrong");
}
else{
context.Response.BinaryWrite (file);
}
}
}
public bool IsReusable {
get {
return false;
}
}
}
```

3 两种方法的性能方面的比较

3.1 事物支持。

ACID 原则，是数据库系统提出的原则，即为事务的原子性、一致性、独立性及持久性。

事务的原子性是指一个事务要么全部执行,要么不执行. 也就是说一个事务不可能只执行了一半就停止了；事务的一致性是指事务的运行并不改变数据库中数据的一致性；事务的独立性是指两个以上的事务不会出现交错执行的状态；事务的持久性是指事务运行成功以后,就系统的更新是永久的,不会无缘无故的回滚。

将文件上传到文件系统，由于这个事物是分为两个步骤执行，第一步是将文件上传到服务器的文件系统中，第二步是将文件名称和上传的路经写到数据库文件中。对于这两个步骤,要求必须同时完成,要么就都不完成。但是在实际中,文件保存在文件系统中，可能因为外界的因素，两个步骤只完成了其中的一个，因而导致文件保存的位置与数据库中的超链接字段内容不符的情况，对于事物支持来说,这种上传文件的方式不具备。

将文件上传到数据库中，这个事物是唯一的、完整的一个操作,上传的方法就是将文件直接写到数据库中,不出现分离、交错、更新等问题,因而数据库系统支持 ACID 原则。

即要维护原子性、一致性、独立性和持久性，那么需要将文件保存在服务器的数据库中更优。

3.2 参照完整性以避免破坏文件链接

参照完整性是两部分数据之间的关系状态。这里指的是文件和文件链接地址之间的参照完整性。实施参照完整性是使用数据库存储文件而不用文件系统存储的最为充足的理由。数据库存储文件,链接地址是将数据库中文件名对应记录的 id 号作为传递参数,并编写 HTTP 处理器程序 FileHandler.ashx,将超级链接中的参数 ID 指向的记录的二进制文件内容字段值以二进制数据的格式回写到浏览器中。其文件名和文件内容是保存在数据库中的同一个记录中，对于数据库进行查询将同时获取到两部分内容；而将文件存储到文件系统中,文件是保存在文件系统中,而地址保存在数据库中，如果其中一项单独发生变化,则直接会破坏文件的链接。

3.3 安全性比较

文件保存在文件系统中，从安全的方面考虑,要对文件系统的安全性进行设置,可以使用文件系统文件权限保护文件；文件保存在数据库中，安全性方面是要从数据库方面进行设置,可以使用数据库用户、角色和权限限制对数据库的访问。如果配置得当,两者旗鼓相当。

但是,人们对于文件系统已经较熟悉,文件的检索、移动、复制、删除已经成为习惯,如果获取了对文件系统的访问权，很容易获取到文件 ;相对而言,如果对数据库 SQL 不熟悉,对于移动文件和检索几乎是不可能的。同时，数据库中保存文件的内容是以二进制形式保存的,对于文件的读取、修改等操作,需要读取、编辑、保存文件内容到硬盘中,这必须用编程方法来完成。其操作相对复杂了很多。因而,数据库存储文件在安全性方面更好一些。

3.4 性能上比较

文件系统是用来优化文件和检索的。因而在文件的存储和检索方面，文件系统做的比数据库更好些。

数据库是为相对小的记录，而不是大文件来优化的。SQL 存储是把记录存储在数据库页面的结构中,它的大小约为 8KB,然而大部分文件都大于 8KB,SQL 把文件拆分成 8KB 文件块,这样文件就存储于多个页面中,这样,文件的检索会增加性能的开销 ;同时,SQL 对于文件的输出遵循 TDS 协议，它的文件传递能力不如文件系统优良；对于大文件（200MB 左右）,如果同时访问的用户太多,意味中同时打开数据库的连接增多，其性能会降低。

但是，在日常的工作中，上传的工作文件,其一般文件就为几十 KB,文件较小,且不太频繁的文件访问,其性能差异就会不大。

3.5 数据备份和复制

数据库的备份和复制是可以自动进行的。在这方面数据库系统存储文件会更好些。

3.6 数据库性能的比较

文件系统保存的仅仅是文件的路径,其占用资源较小；而数据库系统保存的是文件本身，其每条记录的大小将会随着上传文件的大小而定,占用资源较大。对于数据库的查询等各类操作，要求硬件资源和系统开销较高。

3.7 编程的复杂度

因为 IIS 不能直接读取数据库中的文件，因而在数据库中存储文件中，对于文件的读取,要单独编写代码,会比直接存储到文件系统要编些更多的代码。

以上就是对两个系统存储文件的比较。个人认为,对于日常工作中,上传的文件是基于文字性的文件，同时浏览连接的不是很频繁,数据库存储文件还是优于文件系统。具体采取哪一种方法，可根据实际情况具体分析优劣而定。

参考文献

[1]《深入 ASP.NET 2.0 开发》