应用存储过程实现数据分页①

黄龙军

(绍兴文理学院 计算机系, 绍兴 312000)

摘 要: 讨论了在数据库服务器端应用存储过程的方法来实现海量数据的分页, 重点讨论分页存储过程的设计思想、实现方法以及在通过 ADO.NET 命令对象调用存储过程的方法. 通过该方法, 减少了海量数据查询时的网络流量, 提高了系统的效率.

关键词: 海量数据; 分页; 存储过程; ADO.NET; SQL

Implementation of Data Paging with Stored Procedure

HUANG Long-Jun

(Department of Computer Science, Shaoxing University, Shaoxing 312000, China)

Abstract: This paper describes the paging of massive data with stored procedures in the database server, focuses on the design ideas and implementation methods of stored procedures for data paging, and their calls by ADO.NET command object. The method greatly reduces the network traffic when the massive data queries, and improves the efficiency of the system.

Key words: massive data; paging; stored procedure; ADO.NET; SQL

1 引言

Web 系统经常会遇到大量数据的分页显示问题, 传统的分页技术在提取数据时用户等待时间较长[1]. 分页技术是指 Web 网站针对用户的大批量数据进行查询请求时, 将用户所需数据分批取出, 传送至客户端浏览器的技术[1]. 分页的目的可减少网络的流量和用户端的负荷, 让用户有更好的使用体会. 文献 1 从 Ajax 技术的角度, 讨论 Ajax 技术采用异步通信方式, 只更新所需数据的方法来提高系统性能, 但采用的编程方法是直接在功能代码中嵌入 SQL, 存在安全性不高、网络流量较大、查询效率不够高等方面的问题. Visual Studio 2010 中的 GridView 控件实现分页效果的方法很简单, 选择 “启用分页”并设置相应属性即可. 但这种实现方法在每次查询当前页的数据时都要到服务器端取得所有数据, 然后再显示那一页的数据, 存在网络流量大的问题, 特别是在海量数据的访问中, 查询速度慢, 系统效率低. 本文采用直接在数据库服务器端使用存储过程的方法来实现分页效果, 客户端只要传递参储过程名及少量参数即可在服务器端获取其所需的数据, 能有效的提高系统的查询性能.

2 相关知识

存储过程是可以一次执行的 SQL 代码块, 作为一个单元存储在服务器端[2], 并在服务器端执行. 存储过程具有模块化程序设计、提高执行效率、减少网络流量、提供安全机制等优点[3], 这正是我们采用存储过程来实现数据分页的原因. 下面我们对在存储过程中使用的 SQL 的相关知识[3]进行说明, 以便读者能更好地理解后面讨论的存储过程.

使用 SELECT…INTO <表名>可以自动创建一个新表, 并将查询结果行插入到该表中. 使用 INTO 子句生成(临时)结果表, 可以达到缓存数据的目的.

ROW_NUMBER( )函数返回结果集分区内行的序列号(bigint 类型), 每个分区的第一行从 1 开始.

① 收稿时间:2012-05-14;收到修改稿时间:2012-06-10

RANK( )、DENSE_RANK ( )、NTILE(<行数>)等函数也可以达到类似的效果.

SET NOCOUNT 设置为 ON, 可以阻止在结果集中返回受 Transact-SQL 语句或存储过程影响的行计数的消息. 当存储过程中包含一些并不返回许多实际数据的语句, 或者包含循环时, 网络通信流量便会大量减少. 因此, 将 SET NOCOUNT 设置为 ON 可显著提高性能.

SET ROWCOUNT <行数>使 SQL Server 在返回指定的行数之后停止处理查询. 其实, 设置该选项将使大多数 Transact-SQL 语句在受到指定数目的行的影响后停止处理. 把 SET ROWCOUNT 指定为 0 可以重新返回所有的行.

SELECT 语句中的 TOP 语句将在查询结果集中返回的行数限制到指定的行数或行的百分比. 一般应将 ORDER BY 子句与 TOP 语句结合使用, 以避免返回随机行.

关系数据库中的操作是面向集合的, 结果集一般包含若干数据行. 但应用程序并不总能将整个结果集作为一个单元来有效地处理的, 有时候, 应用程序需要一种机制以便每次处理一行或一部分行. 游标正是这种机制, 它支持对结果集的这种扩展处理要求. 在存储过程使用 Transact-SQL 游标的典型过程为[2]:

a.使用 DECLARE CURSOR 语句声明 T-SQL 游标.

b.使用 OPEN 语句打开游标.

c.使用 FETCH 语句提取单个行.

d.使用 CLOSE 语句及 DEALLOCATE 语句关闭并释放游标.

3 分页功能的实现

本文讨论的数据表及存储过程都在关系数据库管理系统 SQL Server 2008 中实现. 这里仅考虑查询所有学生信息, 简化的表结构如表 1 所示.

表 1 学生表(Student)

字段名	数据类型	描述	备注
sno	varchar(10)	学号	主键
sname	nvarchar(10)	姓名	
sex	char(2)	性别	
age	int	年龄	
dept	nvarchar(10)	系别	

分页存储过程的实现可以采用多种方法, 这里我们重点讨论三种方法, 分别采用通过 row_number()方法、select top 方法、游标方法来实现. 在前两种方法中, 因为表名、每页记录数作为参数的缘故, 我们采用 EXEC 执行动态 SQL 的方法. 这三种方法实现的存储过程名统一命名为 SplitPages, 版本号分别为 1、2、3, 参数都设定了默认值, 以便于测试. 若要便于定位到最后一页, 可以在存储过程中返回总页数. 下面的存储过程实现的是不带条件的查询, 若需要进行指定条件的查询, 则可以给存储过程添加参数, 在动态 SQL 或游标声明时带上条件即可.

3.1 通过 row_number() 方法实现存储过程

存储过程 SplitPages 的设计思想是用 row_number()函数给每条记录添加行号并生成临时表 temp, 然后通过行号选择所需要的记录. 该方法需要的辅助空间代价与 student 表大小相同; 时间代价主要耗费在产生临时表及在该表上按行号进行查询. 存储过程的参数为: 表名@tn, 每页记录数@cnt, 当前页号@index.

```
create proc SplitPages(@tn nvarchar(30)='student',
        @cnt int=3, @index int =1) as
begin
  set nocount on
  if object_id('temp') is not null drop table temp
  --若临时表存在, 则先删除
  set @tn=quotename(@tn)
    --quotename 函数使字符串成为有效的标识符
  if object_id(@tn) is null --表名参数的检查
begin
  raiserror('please check table name', 16, 1)
    --错误提示
  return
end
exec('select (row_number() over (order by sno)) as
  rowIndex, * into temp from '+ @tn)
--执行动态 SQL 产生临时表 temp, 包括行号
select sno, sname, sex, age, dept from temp
where rowIndex between (@index-1)*@cnt+1 and
@index*@cnt --在 temp 表中按行号查询结果记录
end
```

3.2 通过 select top 方法实现存储过程

存储过程 SplitPages;2 的设计思想是按字段 sno 升序生成临时表 temp, 然后通过 delete 语句中结合 select top 删除所需要的记录之前的记录, 最后在剩余记录中选择所需要的记录. 该方法需要的辅助空间代价与 student 表大小相同; 时间代价主要耗费在产生临时表、删除所需记录之前的记录、查询结果记录, 一般情况下, 时间效率低于存储过程 SplitPages;1. 存储过程的参数为: 表名@tn, 每页记录数@cnt, 当前页号@index.

```
create proc SplitPages;2(@tn nvarchar(30)='student',
  @cnt int=3, @index int =1) as
begin
  set nocount on
  declare @sql nvarchar(max), @cnt1 int,
  @maxSno varchar(10)
  if @index=1 --第一页直接用 select top 查询出结果
begin
  set @sql='select top ' + convert(varchar(10),
    @cnt) + ' * from ' + @tn + ' order by sno'
  exec (@sql) --执行动态 SQL 查询数据
  return
end
--不是第一页则先删除所需记录之前的记录,
--然后再查询结果记录
if object_id('temp') is not null drop table temp
exec ('select * into temp from ' + @tn + ' order by sno') --产生临时表 temp
set @cnt1=@cnt*(@index-1)
  --计算需要删除的记录数
set @sql='delete from temp from (select top ' + convert(varchar,@cnt1) +' * from temp order by sno) as t where temp.sno=t.sno'
exec (@sql)
--执行动态 SQL 删除所需记录之前的记录
set rowcount @cnt --设置下面查询返回的记录数
select * from temp order by sno--也可用 select top
end
```

3.3 通过游标方法实现存储过程

存储过程 SplitPages;3 的设计思想是通过游标方法使用循环语句逐行访问数据, 先忽略所需记录之前的所有记录, 然后把所需记录逐行插入表变量中, 最后从结果表变量中选择所需记录. 该方法需要的主要辅助空间代价是与结果集相同大小的表变量的空间; 时间代价主要耗费在声明游标时查询 student 表、通过游标进行遍历数据及查询结果表变量, 一般情况下, 时间效率低于存储过程 SplitPages;1. 存储过程的参数为: 每页记录数@cnt, 当前页号@index.

```
create proc SplitPages;3(@cnt int=3, @index int =1) as
begin
  set nocount on
  --声明游标, 若有表名参数,
  --则可采用 exec 执行游标声明的动态 SQL 语句
  declare curSelect cursor for select sno, sname, sex, age, dept from student order by sno
  --声明变量
  declare @i int, @sno varchar(10)
  declare @sname nvarchar(10), @sex char(2)
  declare @age int, @dept nvarchar(10)
  declare @t table(--声明表变量
      sno varchar(10),
      sname nvarchar(10),
      sex char(2),
      age int,
      dept nvarchar(10))
  open curSelect --打开游标
  --从游标提取一条记录到变量中
  fetch next from curSelect
    into @sno, @sname, @sex, @age, @dept
  set @i=0
  declare @start int
  set @start=(@index-1)*@cnt
  --下面的循环忽略所需记录之前的记录
  while @@fetch_status=0 and @i<@start
  begin
    set @i=@i+1
    fetch next from curSelect
      into @sno, @sname, @sex, @age, @dept
  end
  set @i=0
```

```
--下面的循环提取所需记录并插入到表变量中
while @@fetch_status=0 and @i<@cnt
begin
  insert into @t
  values(@sno, @sname, @sex, @age, @dept)
  set @i=@i+1
  fetch next from curSelect
    into @sno, @sname, @sex, @age, @dept
end
select * from @t --从表变量中查询所需记录
close curSelect --关闭游标
deallocate curSelect --释放游标
end
```

3.4 通过 ADO.NET 调用存储过程实现分页

我们在 Visual Studio 2010 中用 C#编写的 SplitPage 函数, 通过 SqlCommand 对象调用存储过程 SplitPages 来实现数据分页; 调用该函数即可达到分页效果.

```
private void SplitPage(string tn, int cnt, int index)
{//参数分别表示表名、每页记录数、当前页号
  string cnnStr =ConfigurationManager.
  ConnectionStrings["CnnStr"].ConnectionString;
  SqlConnection cnn = new SqlConnection(cnnStr);
  cnn.Open(); //打开连接
  SqlCommand cmd = new SqlCommand
("SplitPages", cnn); //用存储过程名创建命令对象
  cmd.CommandType = CommandType.
  StoredProcedure; //指定命令对象的命令类型
  cmd.Parameters.AddWithValue("@tn","student");
  cmd.Parameters.AddWithValue("@cnt", cnt);
  cmd.Parameters.AddWithValue("@index",index);
  SqlDataReader dr = cmd.ExecuteReader();
        //取得结果集放到 SqlDataReader 中
  gvData.DataSource = dr; //数据绑定
  gvData.DataBind();//gvData 为 GridView 控件
  dr.Close();
  cnn.Close();//关闭连接
}
```

4 结语

我们把测试网站和数据库管理系统部署在同一台计算机上, 测试的结果如表 2 所示. 其中, 运行时间单位为毫秒. 系统基本配置是 Windows Server 2003 企业版操作系统, 2G 内存, IE 8.0 浏览器.

表 2 测试结果

方法 \ 记录数	30	300	3000	30000
自动分页	2	28	51	361
嵌入式 SQL	5	24	36	155
存储过程 SplitPages	6	16	29	127

通过在数据库服务器端直接编写存储过程来实现海量数据的分页, 只传给客户端需要的数据, 而不是每次查询都传递所有数据, 减少了网络流量, 提高了系统的性能. 文中讨论的分页存储过程的各种实现方法及其相关知识, 对于数据库编程的学习者及研发人员而言, 具有一定的参考价值.

参考文献

1 刘红坤.基于 Ajax 和 PHP 数据分页的实现.计算机系统应用,2012,21(2):218–220.

2 Watson K.C#2005 数据库编程经典教程.北京:人民邮电出版社,2007:234–265.

3 闪四清,邵明珠.SQL Server 2008 数据库应用实用教程.北京:清华大学出版社,2009.204–256.

(上接第 181 页)

参考文献

1 李金艳,郭威,王道文,等.JasperReport 在 SaaS 物流平台报表中的设计与实现.第十五届全国青年通信学术会议.166–170.

2 张震,张曾,邓丽曼,等.用 ActiveX 技术实现 WWW 环境下的报表打印.微型计算机与应用,1999.9:18–25.

3 李红,何婧,李浩,等.基于 SOA 的物流服务平台设计.计算机科学,2009,36(8):186–190.

4 陆歌皓,李什金,吴超凡.Drools 规则引擎在现代物流信息平台的应用.计算机科学,2011,38(10):447–40.

5 单东林,张晓菲,巍然.锋利的 jQuery.北京:电子工业出版社,2008.

6 http://www.jatools.com/.

7 叶阿勇.报表动态生成的思想与实现.福建电脑,2003,2:20–21.