Research of The Heavy Rain Basic Database System Based on VC++

Jintao Ye, Tao Peng, Junchao Wang, Guangliu Feng
Hubei Key Laboratory for Heavy Rain Monitoring and Warning Research,
Institute of Heavy Rain, China Meteorological Administration,Wuhan 430074,China
yejintao@whihr.com.cn

Abstract

Regarding the heavy rain basic database, the paper describes that the program of steps and methods to SQL Server 2000 database making use of the interface of ADO in the VC++. Meanwhile, according to the actual application status of the heavy rain basic database, the paper provides the program of batch meteorology data input. Besides, it detailed depicts some methods to optimize the capability of database, such as making use of the function of SQL Server procedure to realize the real time auto-calculate of cumulative rainfall, and utilizing the table of database to establish index. Consequently, they improve the efficiency of the whole system.

Keywords*:ADO; VC++; SQL Server; Database optimized*

1. Introduction

Our country is frequently hit by heavy rains in summer, and the storm mostly happened in Jiang-huai Area during the rainy season. Researching on storm is always the emphasis and difficulty [1-3] in weather research and forecasting work due to the serious damage caused by flooding. The building of heavy rain basic database has an important meaning for both the researching and forecasting business.

Manual operation couldn't accomplish the work with the emerging of the new observation method, more and more types of meteorological data and increasingly larger quantity of data, so we need a quick means to input the data to database. Database technology has been widely used with the development of computer technology. Served as a powerful development tool for the database theory, VC++ provides a faster access to database technology, and has a good graphical user-interface. This paper established a real-time heavy rain database on the basic of the rainfall database research by Liu Guizhi[4].In addition, the paper describes the specific methods to get access to SQL Server 2000 database technology making use of the ADO interface in VC++, whose purpose is to import meteorological data quickly.

2. the Heavy rain database

2.1. ADO overview

ADO (ActiveX Data Objects) is Microsoft's new interface for database applications, it is also a high-level database access technology building on the basis of OLE DB. Besides ADO enables your client applications to access and manipulate data in a database server through any OLEDB provider[5].

2.2. Brief introduction of the Heavy rain database

The heavy rain basic database comes from the subsystem of the special social commonweal research program "GIS-based analysis of rainfall-induced geological hazards warning system", which was supported by the Ministry of Science and Technology. Database mainly collect the meteorological monitoring data and forecast products, including weather information from ground station (rainfall every hour of the day throughout the Province, the latest rainfall, accumulative rainfall), information from Satellite (a real-time link of FY-2 satellite image), data from weather radar (real-time radar puzzle) and objective or subjective forecasting of sub-country. As the system needs to read latest information from the database, we have to build a real-time database, import real-time information to database every day , so that the early warning system can call database anytime. This paper realizes the automatic update of data records by combing VC++ programs and ADO technology.

3. An instance of batch input of rainfall data in VC++ using ADO

3.1. Create application frame and initialize OLE/COM library circumstance

Create a standard MFC AppWizard application hourlyrain.

3.2. Import ADO library

We use the #import statement to invoke ADO in the header file stdafx.h of the project before using ADO. Using that directive allows the complier to compile correctly. Below shows the correct way to use #import with Msado10.dll:

```
#import"c:\programfiles\common files\system\ado\msado15.dll
"no_namespaces rename("EOF" adoEOF")
```

This statement declares that the project uses ADO without its namespace, and renamed constant EOF adoEOF to avoid constant conflict. Now we can use ADO interface without other header files.

3.3. Initialize OLE/COM library environment

It's important to note that ADO library is a set of dynamic COM libraries. It means we must initialize the OLE/COM library environment for our MFC application. A better approach is to initialize OLE/COM library environment in the member function InitInstance in the main class of our application.

```
if(!AfxOleInit())
{
AfxMessageBox("OLE initialization is error!");
return false;}
```
[6]

3.4. Using the _ConnectionPtr interface to connect the database

First we establish a database named “MyDatabase” through SQL Server 2000. And the database have a user name “MyUserID”(you can name it as you wish), a password “MyPassWord” so that database access can be controlled . Finally we fill the ServerName with our local server name.

This paper uses an instance of the smart pointer _ConnectionPtr. First an object is defined in the header file hourlyrain.h as shown below:

```
_ConnectionPtr    m_pConnection;
```

Then add the following code to the member function InitInstance:

```
CString strSRC="Driver=SQL Server;Server=
ServerName;Database= MyDatabase;UID= MyUserID;PWD=
MyPassWord ";
if(FAILED(m_pConnection.CreateInstance("ADODB.Connect
ion")))
    {
        AfxMessageBox("Create Instance failed");
        return 0;
    }
if(FAILED(m_pConnection->Open(_bstr_t(strSRC),"","",-1)))
    {
        AfxMessageBox("Can not open Database!");
        m_pConnection.Release();
        return 0;
    }
```

You can check for the prototype of the Open function in MSDN comes with VC++.

```
HRESULT Recordset15::Open(const _variant_t & Source,
const_variant_t & ActiveConnection,enum CursorTypeEnum
CursorTpye, enum LockTypeEnum LockType, long Options)
```

3.5. Using the _RecordsetPtr interface to open database table

In order to get the result record set we defines an object of the smart pointer _RecordsetPtr in the header file hourlyrain.h as shown below:

```
_RecordsetPtr    m_pRecordset;
```

The following code is for creating an instance of the smart pointer:

```
CString strSQL="select * from dbo.mytable";
if(FAILED(m_pRecordset.CreateInstance("ADODB.Rec
ordset")))
{
    AfxMessageBox("Create Instance failed");
    return;
}
if(FAILED(m_pRecordset->Open(_variant_t(strSQL),_v
ariant_t(strSRC),adOpenDynamic,adLockOptimistic,adC
mdText)))
{
    AfxMessageBox("Open table failed!");
    m_pRecordset.Release();
    return;
}
```

3.6. Data entry of database table

We can use functions in MFC to read weather files, to import batch data to tables, to traverse, add, remove, or delete our tables after the connection of database and the opening of record sets through ADO technology.

Below is an example, add a new record to the table real_rain24_data (latest rainfall throughout the province):

First create a database table real_rain24_data through SQL Server:

```
CREATE TABLE [dbo].[real_rain24_data] (
[date_dat] [bigint] NOT NULL ,          /*information
date with the form yyyymmdd */
[time_dat] [smallint] NOT NULL ,        /*information
time， 05:00 Beijing time*/
[no_57249] [decimal](7, 1) NULL,/*information of the
  station with the number 57249*/
[no_57251] [decimal](7, 1) NULL ,
……....... /*information from other stations in HuBei
Province*/
[no_58501] [decimal](7, 1) NULL /* information of the
station with the number 58501*/
) ON [PRIMARY]
GO
```

Then add a record through VC:

```
void CHourlyrainApp::OnRealHourlyRain()
 {
int date_dat=20070307,station=57249, time_dat=8;
float rain=0.5;
CString   no_station;
no_station.Format("no_%d",station);
 try
 {
 m_pRecordset->AddNew();
m_pRecordset->PutCollect(_variant_t((long)0),_variant_t
((long)date_dat));
m_pRecordset->PutCollect(_variant_t((long)1),_variant_t
((long)time_dat));
m_pRecordset->PutCollect(_variant_t(no_station),_varia
nt_t((float)rain));
      m_pRecordset->Update();
 }
```

In practical applications, data_dat, station, rain are all variables whose value can be read from the meteorological data files; time_dat serves as the time of the live rainfall which is set to be 08:00 here.

If you want to import real-time data, you need to set the primary key for each table in case that you add repetitive data to the database.

3.7. Close and release the object in destructor function as follows:

```
if(m_pConnection!=NULL)
```

```
  {
 m_pConnection ->Close();
 m_pConnection.Release();
  }
```

4. Heavy rain data optimization

4.1. Creating Index

Due to the data of some tables in the heavy rain basic database coming from real-time data, the amount of data grows with each passing day .It will be very difficult to inquire if the storage of data is confused and disordered. Creating index is an effective method to accelerate inquiry, and we can create one or more indexes to accelerate inquiry optimization according to the demand. Based on good database design, efficient index design is paramount to achieve good performance in SQLSERVER[7].

For micaps_surf_data—the table of ground basic database, we find there exist lots of records of redundant data in partial historical data which is previously inputted ,so it's necessary to delete existed redundant data before creating index.

```
delete a from micaps_surf_data   a
inner join
(select V04001,V04002,V04003,V04004,V01000
from micaps_surf_data
group by V04001,V04002,V04003,V04004,V01000
  having count(*) > 1 )b
on (a.V04001=b.V04001 and a.V04002=b.V04002 and
a.V04003=b.V04003 and   a.V04004=b.V04004 and
a.V01000=b.V01000 )
and
autoID not in
(select min(autoID)
from micaps_surf_data
group by V04001,V04002,V04003,V04004,V01000
having count(*)>1)
```

Fields of the table, V04001, V04002, V04003, V04004 and V01000 represent year, month, day, hour, and station index number respectively. After deleting records of redundant data, we create indexes respectively for these fields of the table, V04001, V04002, V04003, V04004 and V01000, to optimize inquiry function. SQL Server will directly locate the data records inquired by user through following the indication of index, consequently improving query performance.

4.2. Calculation of accumulative rainfall utilizing stored procedures

Providing the accumulative rainfall data of 1, 2, 3, 7, 10 and 14 Days according to project requirement, we have to accumulate rainfall data per day at eight o'clock of Hubei Province. The adoptive method is calculating accumulatively the data from table real_rain24_data, and creating a procedure which is named cumulate rain by utilizing the cursor and trigger mechanism of SQL Server[8]. This procedure can run automatically to calculate rainfall data accumulatively by trigger Configuration.

Building the table of accumulated rainfall data in SQL Server:

```
CREATE TABLE [dbo].[cumulate_rain_data] (
[date_dat] [bigint] NOT NULL ,               /*Date of
data*/
[station] [bigint] NOT NULL ,                /* station
index number */
[day_1_rain] [decimal](7, 1) NULL ,      /*Rainfall
accumulation of 1 Day */
[day_2_rain] [decimal](7, 1) NULL ,      /*Rainfall
accumulation of 2 Days */
[day_3_rain] [decimal](7, 1) NULL ,      /*Rainfall
accumulation of 3 Days */
[day_7_rain] [decimal](7, 1) NULL ,      /*Rainfall
accumulation of 7 Days */
[day_10_rain] [decimal](7, 1) NULL ,     /*Rainfall
accumulation of 10 Days */
[day_14_rain] [decimal](7, 1) NULL        /*Rainfall
accumulation of 14 Days */
) ON [PRIMARY]
GO
```

Corresponding code of the storage procedure---cumulate_rain, is as follows:

```
CREATE PROCEDURE dbo.cumulate_rain   AS
declare @day1 bigint, @day2 bigint,@day3 bigint
select @day1=convert(char(8),getdate(),112)
select
@day2=convert(char(8),dateadd(day,-1,getdate()),112)
select
@day3=convert(char(8),dateadd(day,-2,getdate()),112)
insert into cumulate_rain_data     ( date_dat,station)
(select @day1, station from para_surf_sta )
declare @no_sta bigint
declare @cno_sta varchar(8),@str varchar(5000)

declare sta_cursor cursor
for select station from para_surf_sta
open sta_cursor

fetch next from sta_cursor into @no_sta
while @@fetch_status=0
begin
set @cno_sta='no_'+str(@no_sta,5)
set @str='
update cumulate_rain_data
set day_2_rain=(
select isnull(sum('+@cno_sta+'),0)
from real_rain24_data where time_dat=8 and
(date_dat='+str(@day1,8)+' or date_dat='+str(@day2,8)+')),
day_3_rain=(
select isnull(sum('+@cno_sta+'),0)
from real_rain24_data where time_dat=8 and
(date_dat='+str(@day1,8)+' or date_dat='+str(@day2,8)+' or
date_dat='+str(@day3,8)+'))
where date_dat='+str(@day1,8)+' and
station='+str(@no_sta,5)
EXEC(@str)
fetch next from sta_cursor into @no_sta
end

close sta_cursor
deallocate sta_cursor
```

5. Conclusion

This paper introduces some methods and techniques of data inputting and performance optimization of heavy rain basic database. Above-mentioned methods and programmer steps, which accomplish batch inputting utilizing ADO in VC, have passed the test under VC++6.0 environment. In addition to the above introduction of some basic functions, ADO can also get record set by utilizing the Command object to execute SQL commands directly. Consequently, Utilizing ADO interface technology flexibly in VC++6.0 programmer technology is the safest, the most reliable, rapid and effective method to accomplish batch inputting meteorological data.

Meanwhile, it is very effective that using Transact-SQL provided by SQL Sever 2000 to process some data of the database tables. With the increasing of data in the database table, it is very necessary to optimize its performance and improve the efficiency of SQL Sever data processing, so that the early warning system can inquire quickly. Except partial optimization function mentioned in this paper, there are many method to optimize database, such as memory allocation of database server and SQL statement optimization. These methods will be applied to system one by one in the future. After above-mentioned optimization, the inquiry and retrieval function has improve obviously, thus improve efficiency of the whole system.

6. ACKNOWLEDGEMENT

The study is supported by the National Natural Science Fou ndation of China (41205086) and the Special Research Project (meteorology) of the Ministry of Science and Technology of C hina (GYHY201206028, GYHY201306056 ,GYHY20130605 9).

7. References

[1] CHEN Zhong-ming, MIN Wen-bin CUI Chun-guang, "Diagnostic Analysis on the Formation and Development of Meso-scale Vortex Systems[J]" Torrential Rain and Disasters 2007(1)：29-34

[2] WAGN Li,JIN Qi, KE Yi-ming ,"Quantitative Evaluation of Short-term Strong Precipitation Forecasting of Three NWP Models [J]"Torrential Rain and Disasters 2007(1)：29-34

[3] LI Yin-e; SHEN Wei; ZHANG Ping-ping, "Diagnostic Study of a Heavy Rain Caused by Cold Vortex in North China" Torrential Rain and Disasters 2007(1)：29-34

[4] LIU Gui-zhi, WANG Zhi-bin, CHEN Bo, Building And Maintaining of Heavy rain Basic Database.[G].Wuhan NWP. heavy rain Disasters, Beijing : China Meteorological Press，2004(1):91-100.

[5] QiuShi Technology. Book of database management and developing technique for SQL Server 2000 [M]. Beijing: People's Post and Telecommunication Publishing House，2004.

[6] QiMin Studio. Development And Examples of database application systems combined Visual C++ and SQL Server [M]. Beijing: People's Post and Telecommunication Publishing House，2004.

[7] SA Shi-xuan，Wang Shan. Database System Introduction [M]. Beijing: Higher Education Press，2000.

[8] WU Jing-hui. Analysis and Research on Database Optimization Technologies [J]. Computer and Modernization ，2004(12)：90-92.