A Novel Approach to Accessing Large Images in the Database

Yong Ren

Applied Technology College of Soochow University

Suzhou, Jiangsu, 215325, P.R.China

Keywords: OOP; Database; BLOB; Image; Big data

Abstract. This paper describes several ways to achieve access to images in a database and on the field of application of these methods have drawbacks as well as a detailed comparison, we propose a feasible implementation. Finally, this approach will be promoted, making it suitable for a variety of media file formats

Introduction

Database provides a BLOB, CLOB, BFILE and NCLOB four types of data to store large objects. BLOB is a binary large object that is used in the binary data stored in the database without structure, up to 4GB. BLOB type is sufficient to meet the general requirements for multimedia storage, so in this paper we use the BLOB type to store an image resource in the database.

System analysis and design

In the database development process, often need to store information in a database of some notes, and the information content of these notes generally larger, diverse formats - if possible, voice files, video files, image files, text files, and how to achieve these different access and preview format memo file, has been more concerned about a problem developer, this article describes three systems access Remarks binary information.

Method One: file is stored in a fixed path, the database access can save the file path and name of the data space, to avoid excessive expansion of the database, but the memo file in a certain directory must not be lost, and the same catalog file can not be re-the name of the file management may cause some difficulty, in addition, in the OLE control displays memo file browser, as every time the call server, so slow.

Method Two: In a database blob type or varbinary type field storing memo file, if the file is stored in a database for later, you can delete the original temporary files on the hard disk, no complex binary file management, and the database can be stored on a network server on sharing of data is very convenient.

Method three: storage structure to store the memo file with OLE locally. You can put all the information is stored in a binary file OLE storage file management more convenient. When the binary information stored, you can delete the original temporary files; because the file does not need to open the store for each execution server program to display the stored information, so access is faster.

Image database technology has been committed to solve efficiently store and manage large volumes of digital images of the problem. It is the continuation and development of database technology, on the one hand, the image data and text data, there are essential differences in the text data can be successfully applied in the field of traditional database technology, if the static field copied to the image database, the result is often inefficient, or ineffective; on the other hand, many of the results of traditional database, such as SQL language, image database indexing techniques and so worthy of reference. The combination of these two aspects of image database technology become mainstream development.

BLOB is a very large indefinite binary or character data, usually documents (.txt, .doc) and pictures (.jpeg, .gif, .bmp), which can be stored in the database. In SQL Server, BLOB can be text, ntext or

image data types. Image data type storage is uncertain length binary data, the maximum length is 2GB. BLOB data in SQL Server system stored data is different from the ordinary type, store data directly on the user-defined fields for common types of data system values, and for BLOB type data, the system opens a new store page to store these data the table stored BLOB type data fields is only a 16-byte pointer, the pointer to store BLOB data which records the page.

BLOB data is the data type of large amounts of data, it will take a lot of hard disk space, memory and network resources, so the rational design contains BLOB data type attribute table, to improve storage efficiency, query speed has a great influence. BLOB general design principles are as follows:

Binary large object does not have to be stored as text, ntext, or image data types, which can be used as varchar or varbinary data type village in the table. Select the type of data to be stored according to the actual size of the BLOB. If the data does not exceed 8K, then use Varchar or varbinary data types. If the size of these large objects more than 8K, then use text, ntext, or image data types.

Common design problem is the image stored in a database or file system exists. In most cases, the best image file stored in a database together with other data. Because the image data file is stored in the database has many advantages:

Easy to manage when the BLOB and other data stored in the database together, BLOB and tabular data backup and recovery together. This reduces the chance of data tables with BLOB data is not synchronized, but also reduces other users accidentally delete a file system path and risk BLOB data locations. In addition, the data storage insert, update and delete BLOB and other data are implemented in the same transaction in the database. This ensures consistency between data consistency and file with the database. There is little benefit to the file system does not need to file a separate security.

Although the file system scalability is designed to handle a large number of objects of different sizes, but the file system can not be optimized for a large number of small files. In this case, the database system can be optimized.

The availability of a database with more than a file system availability. Database replication allows replication in a distributed environment, and potentially modify data distribution. In the case of primary system failure, the log transfer provides a way to retain a copy of the database backup.

Storing images into the database

The image stored in the database has three steps. First, the content of the image read stream object. Then, the image is copied from the stream object to the parameter in the insert statement. Finally, the implementation of the insert operation.

Load the image content. When read and write, we use the concept of a unified stream to operate a variety of different types of resources, and the resources to be saved by the flow on different media. In the first step towards the image is stored in the database will be loaded with the contents of the image to the stream object. C ++ Builder offers a variety of stream objects, such as TMemoryStream and TBlobStream.

TMemoryStream usually used as an intermediate target, which provides reservation information, generic object stream I / O as well as from other storage media functions to read and write, but also made a number of methods and properties to manage dynamic memory cache, we use to store TMemoryStream dynamic data stored in the cache.

If you want to save the image file in the file system, you can use TMemoryStream LoadFromStream class methods. If you have already read in the image (such as loading the TImage in) from the file system, you can use the TBitmap class SaveToStream method to copy the image content to the memory stream. Whether an image where the image can be read into the stream object.

Parameters Setting. After completion of the read, it is necessary to copy images from the stream object to the parameter SQL insert statement. Is based on the use of ADO or BDE, this step is slightly different. If you use the Table component or components rather than TClientDataset Query component, then it should be copied from the stream to the TField rather than argument.

Perform an insert operation. Execute SQL statements and call the Post method is the real execution.

Assumes that the target table named image, and contains two. The first column is a string, string_field. The second column is BLOB, image_field. Images saved to BLOB column.

① using ADO or BDE's TQuery component and then ② using the Table component

Read image from the database

Reads an image from a database also has three steps. In fact, reading the similar images and store images, but just the opposite step. First, the implementation of SQL statements or open the Table component, the returned result set should include an image object. VCL open when the BLOB data set that contains the image, create a subclass TBlobField or TBolbField to store images. Then, extract the contents of the BLOB into memory stream. Once the image is in the memory stream, you can store images: If you want the image to a file system, you can use the SaveToFile method TMemoryStream; if you want to store the image into the TImage control, you can use the TBitmap :: LoadFromStream, similar images into the database code.

If you only want images stored in the database from the file system, the above method can also be used for reading and writing JPEG files. But if read from the database and display JPEG images, you need to add a bit of code, which is due after reading JPEG images from the database and reads from the memory stream to execute a JPEG decompression steps, we can use JPEG related classes to implement, and use similar code TMemoryStream implementation.

How to save a JPEG image to the database is only dependent on the existing JPEG files are saved in the file system or before saving the bitmap file compressed into a JPEG file. If you only want to save the file in the file system, you can write a JPEG file directly to the database, as BMP files is similar. If you need to compress the image before saving it to the database, you need to use TJPEGImage class again.

If the source image is a file system BMP images, BMP files to JPEG images saved to the database. First, the BMP file into TBitmap object, and then use the Assign method TJPEGImage assigned to TJPEG object, this step also completed assignment JPEG compression step. Then create a stream object, writing compressed JPEG images, the last written to the database.

If the source image has been loaded into the TImage component, the TImage object to JPEG images saved to the database. First, use the Assign method TJPEGImage TImage object assigned to TJPEGImage object, and then create a stream object, writing compressed JPEG images, the last written to the database.

If you read a bitmap, icon, or Windows metafile from the database, you can use the data-aware controls TDBImage reduce some code. By TDBImage TDataSource components connected to a BLOB field. But it has some limitations, such as Windows Metafile images can only browse for JPEG and other image you can not browse.

VCL provides a class called TBlobStream. TBlobStream and TFileStream similar. TFileStream a file is packaged in a stream object. When read TFileStream, actually read the corresponding file. In addition to reading and writing BLOB fields packages, TblobStream and TFileStream do the same thing

Use TBlobStream class implementation can make out a lot of code to streamline, but TBlobStream has many deficiencies. First, TBlobStream only be used BDE datasets. If you are using ADO dataset, you have to be replaced with TADOBlobStream. Obviously TmemoryStream versatility to be better. Second, in the manipulation of the existence of a record of all invalid BLOB stream, which stream to the field on a data set. When you want to manipulate a record, you must delete or rebuild each BLOB stream.

Conclusion

The technique used in this article is not only an image, you can store any type of field codes, including sound files, video files, Word documents, Excel files. Obviously, this type does not need to be loaded

into the object TImage. But reading a file from the file system and saved to the database method is applicable to all types of files.

Acknowledgements.

This work was sponsored by Qing Lan Project of Jiangsu Province.

References

[1]. Pietroy D, Baubeau E, Faure N, et al. Intensity profile distortion at the processing image plane of a focused femtosecond laser below the critical power: Analysis and counteraction[J]. Optics and Lasers in Engineering, 2015:138–143.

[2]. Yang M, Chao H. ECISER: Efficient Clip-art Image SEgmentation by Re-rasterization ☆[J]. Computer-Aided Design, 2015:105–116.

[3]. Das N, Acharya K, Sarkar R. A benchmark image database of isolated Bangla handwritten compound characters[J]. International Journal on Document Analysis and Recognition (IJDAR), 2014, 17:413-431. DOI:10.1007/s10032-014-0222-y.

[4]. Nagy G. Image database[J]. Image and Vision Computing, 1985, 3:111–117.

[5]. Chang S K, Yan C W, Dimitroff D C. An intelligent image database system[J]. Software Engineering, IEEE Transactions on, 1988, 14:681 - 688. DOI:10.1109/32.6147.

[6]. Zheng Z, Du Z, Li L. BigData Oriented Open Scalable Relational Data Model[C]. //Big Data (BigData Congress), 2014 IEEE International Congress on. IEEE, 2014:398 - 405.

[7]. Zhang J, Huang M L. Density approach: a new model for BigData analysis and visualization[J]. Concurrency and Computation: Practice and Experience, 2014. DOI:10.1002/cpe.3337.

[8]. Binnig C, Salama A, Zamanian E. XDB - A Novel Database Architecture for Data Analytics as a Service[C]. //Big Data (BigData Congress), 2014 IEEE International Congress on. IEEE, 2014:96 - 103.

[9]. He W, Zhou M, Gong X. Massive Parallel Join in NUMA Architecture[C]. //Big Data (BigData Congress), 2013 IEEE International Congress on. IEEE, 2013:219 - 226.

[10]. Zeng W, Yang Y, Luo B. Access control for big data using data content[C]. //Big Data, 2013 IEEE International Conference on. IEEE, 2013:45 - 47.

[11]. Dupac M. An object-oriented approach for mechanical components design and visualization[J]. Engineering with Computers, 2012, 28:95-107. DOI:10.1007/s00366-011-0220-3.