Image storage optimization and feature calculation on Netezza Database system

Y. Donon[1], R. Paringer[1,2], A. Kupriyanov[1,2]

[1]Samara National Research University, Moskovskoe Shosse 34A, Samara, Russia, 443086
[2]Image Processing Systems Institute of RAS - Branch of the FSRC "Crystallography and Photonics" RAS, Molodogvardejskaya street 151, Samara, Russia, 443001

Abstract. The storage of images in databases has long been a delicate matter but comes now more and more in use. This kind of data is in most DBMS limited in storage size to 64kb, making necessary to divide large objects into fragments; moreover, programming languages also present limitations in data retrieval. In this paper, we experimented the retrieval, reconstitution, feature calculation of pictures using different fragment size and determined the optimal one. In this work, the server used was a Netezza model from the N2001 line. In this work, we present the architecture of the database, our research and bring recommendation about the optimal fragmentation size to store images into a database.

1. Introduction

In the context of our research, the use of a database system for an accelerated storage of our picture dataset showed itself the most appropriate alternative. Having a Netezza server at our disposition and a dedicated network, we evaluated it as the most appropriate way to store and access our datasets.

The datasets used in that research is constituted of about two hundred-thirty thousand pictures taken from social networks; we downloaded each of them by fragments, calculated a feature that we then loaded again on the database.

2. Database

Netezza supports two types of binary data for both internal and external tables. VARBINARY and ST_GEOMETRY. Both support a length's field of 1 to 64000 bytes, the second being optimized for spatial or geometric analysis functions. VARBINARY and BLOB storage are of the same data type, BLOB being equivalent in other DBMS. Netezza doesn't support alternative as MEDIUMBLOB or LONGBLOB for the storage of large binary data objects.

Open Database Connectivity (ODBC) is a DBMS access interface that allows using SQL for data access. In most of the commonly used languages, the buffer size of the ODBC driver is 4096 bytes. This apposes some additional difficulty to the retrieval of blobs on the DBMS. Realizing the potential importance of this limitation, we limited ourselves in our experiments to a value close to the maximum buffer size.

On an important notice, once converted to hexadecimal value and thus removing the compression operated by the file system to pictures, our dataset ended up about twice bigger than in its original form.

During our experiment, the data where first converted to hexadecimal and stocked into external tables, then the external tables got inserted into the database.

2.1. Tables description

Sections should be numbered with a dot following the number and then separated by a single space:

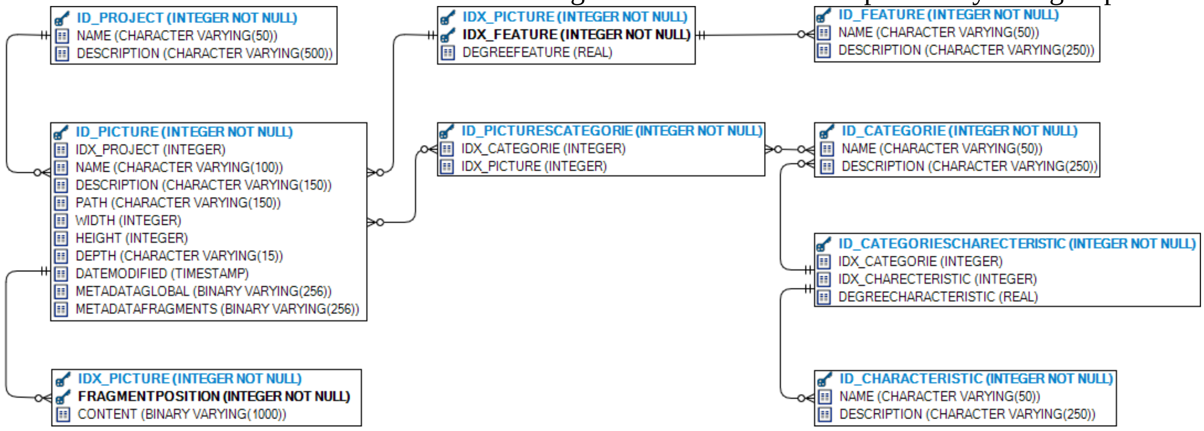
ID_PROJECT (INTEGER NOT NULL)
NAME (CHARACTER VARYING(50))
DESCRIPTION (CHARACTER VARYING(500))
IDX_PICTURE (INTEGER NOT NULL)
IDX_FEATURE (INTEGER NOT NULL)
DEGREEFEATURE (REAL)
ID_FEATURE (INTEGER NOT NULL)
NAME (CHARACTER VARYING(50))
DESCRIPTION (CHARACTER VARYING(250))
ID_PICTURE (INTEGER NOT NULL)
IDX_PROJECT (INTEGER)
NAME (CHARACTER VARYING(100))
DESCRIPTION (CHARACTER VARYING(150))
PATH (CHARACTER VARYING(150))
WIDTH (INTEGER)
HEIGHT (INTEGER)
DEPTH (CHARACTER VARYING(15))
DATEMODIFIED (TIMESTAMP)
METADATAGLOBAL (BINARY VARYING(256))
METADATAFRAGMENTS (BINARY VARYING(256))
ID_PICTURESCATEGORIE (INTEGER NOT NULL)
IDX_CATEGORIE (INTEGER)
IDX_PICTURE (INTEGER)
ID_CATEGORIE (INTEGER NOT NULL)
NAME (CHARACTER VARYING(50))
DESCRIPTION (CHARACTER VARYING(250))
ID_CATEGORIESCHARECTERISTIC (INTEGER NOT NULL)
IDX_CATEGORIE (INTEGER)
IDX_CHARECTERISTIC (INTEGER)
DEGREECHARACTERISTIC (REAL)
IDX_PICTURE (INTEGER NOT NULL)
FRAGMENTPOSITION (INTEGER NOT NULL)
CONTENT (BINARY VARYING(1000))
ID_CHARACTERISTIC (INTEGER NOT NULL)
NAME (CHARACTER VARYING(50))
DESCRIPTION (CHARACTER VARYING(250))

Figure 1. Database Logical Data Model.

2.1.1. Projects

Our dataset uses only one project; however we designed our database in order to store several as the features calculation and storage of picture is commonly used in our works.

2.1.2. Pictures

This table stores the characteristics of a picture. It is mostly used in our researches to filter pictures obeying to certain given characteristics and allows the retrieval and calculation of some features without the need to download the whole picture through, among other things, the use of metadata.

2.1.3. Fragments

This table is one of the central parts of our experiments; it contains fragments of pictures in binary. It is the size of the column content that we dimensioned in the framework of this experiment. As every picture can have an important number of fragments, an integer identifier (ID) for the table wouldn’t have been sufficient and a longer type of ID would have taken a lot of space. A simple and efficient solution was to create a double primary key constrain on both the foreign key of the picture and the fragment position, the second representing the position of the fragment in the image, making their combination unique.

2.1.4. Features & FeaturesDegree

Each picture may have several features with a given degree, each features can appear with several pictures with different degrees. A picture only has one feature of a kind, as such the ID is composed of both foreign keys of pictures and features. Features are any of the numerical characteristics calculated from the images such as: texture features [1], geometric features [2], histogram features [3], features of the spatial spectrum [4] and others…

2.1.5. Categories & PicturesCategories

The most important difference in terms of architecture between FeaturesDegree and PicturesCategories is in de multiplicity of characteristics a category shall have. Image categories can be the results of classification [5], annotation [6] of images, or object detection [7] of object on an images. They are non-numeric characteristics.

2.1.6. Characteristics & CategoriesCharacteristics

Every category may have many characteristics, each coming with different degrees. This explains why pictures on the other side shall have several times the same characteristic. The category characteristic allows you to further describe a non-numeric characteristic using numeric values. For example, in the case of multiple identical categories on images for classification (stage of disease [8]), or when annotated (for neural networks [9]).

2.2. Database storage versus file systems

It has been long frowned upon to store images in a database, however, in the latest years, technical developments allowed it to become more and more of a sensitive option, some experts going as far as declaring that there is no reasons not to do so. Of course, we can temperate this as the price of the storage in DBMS, for example, still higher than in file systems [10].

However DBMS presents other advantages such as a complete historization of changes, with facilitate management. The absence of need for an external backup strategy as the DBMS should already include his own. Facilitated access control, the DBMS is by essence fit to control access to users. Moreover, DBMS ensure the data consistency, in particular regarding the preservation of original metadata, often regarded as problematic in file systems [11].

2.3. Theoretically recommended storage size for optimization

We consider an optimal ratio between the Size of our fragments and their amount, depending on their size and to limit the storage space lost while limiting to a reasonable amount of fragments to be following. Considering$i$, the size of an image and$F$, the size of fragments, and$a$, the amount of pictures to store the most appropriate fragment size is where $\sum_{n=1}^{a}\left(2\frac{i_n}{a}\right) = F^2$, in our case 963 bytes, that has been rounded up to 1000. This formula represents the intersection between an the average size of pictures and the amount of fragments it is divided in.

3. Results

The experiment was conducted on a data set consisting of more than 200,000 images with a total volume of about 50 GB.

At the first stage of the experiment, images were prepared for uploading to the databases using a conversion program developed for the experiment. The pictures have been segmented in different fragment size sets. The sets were of 500, 1000, 2000, 3000 and 4000 byte fragments were obtained and loaded into the corresponding databases.

In the secon stage, using a script also developed for the needs of our experiments, we estimated the time needed by the database to retrieve all the fragments corresponding to an image. During the experiment, the time was measured by the search for the fragments of 1000 images. Each search was repeated 100 times and the average time calculated. The image sets were chosen randomly but were similar for all fragments sizes. In accordance with the previous assessment of the optimal fragmentation size, the average search time for images with a fragment size of 1000 bytes appeared as the optimal. The results of the time estimates are presented in table 1 / in figure 1.

Table 1. Time of treatment depending on fragment size.

Fragment size (Binary cahracters)	500	1000	2000	3000	4000
Time (estimation)	1,088	1,000	1,004	1,038	1,002
Coefficient of variation, %	0,039	0,106	0,041	0,046	0,054

4. Conclusion

The structure of database presented allows storing large sets of images and the results of their processing in an optimal way. Application of the specified structure together with the database system mentioned in our paper can be used for fast search of initial images and results of classification or annotation of images. Studies have shown that the use of a fragment size (1000) optimize reasonably the image reserved space and provides the best performance compared to other fragmentation sizes studied.

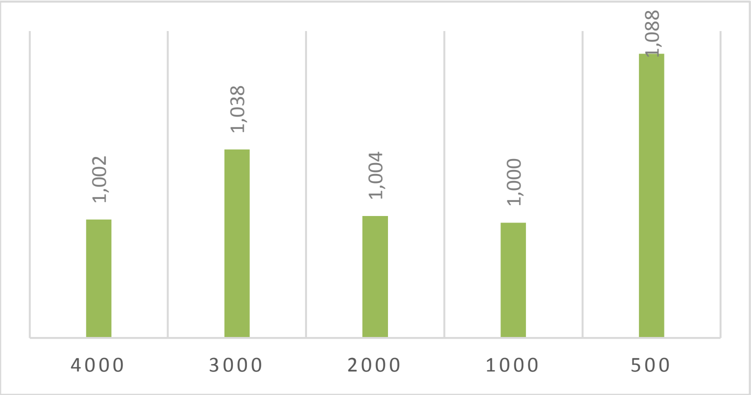

Figure 1. Relative values of searching time of images depending on the of the fragment size.

5. References

[1] Biryukova, E. Development of the effective set of features construction technology for texture image classes discrimination / E. Biryukova, R. Paringer, A.V. Kupriyanov // CEUR Workshop Proceedings. – 2016. – Vol. 1638. – P. 263-269.

[2] Paringer, R.A. Methods For Estimating Geometric Parameters of The Dendrite's Crystallograms / R.A. Paringer, A.V. Kupriyanov // Proceedings of 8th Open German-Russian Workshop "Pattern Recognition and Image Under-standing" OGRW-8-11. – 2011. – P. 226-229.

[3] Pratt, W. Digital Image Processing. – California: Wiley, 1991. – 734 p.

[4] Kravtsova, N.S. Parallel implementation of the informative areas generation method in the spatial spectrum domain / N.S. Kravtsova, R.A. Paringer, A.V. Kupriyanov // Computer Optics. – 2017. – Vol. 41(4). – P. 585-587. DOI: 10.18287/2412-6179-2017-41-4-585-587.

[5] Zheng, Z. Short-term traffic volume forecasting: a k-nearest neighbor approach enhanced by constrained linearly sewing principle component algorithm / Z. Zheng, D. Su // Transportation Research Part C: Emerging Technologies. – 2014. – Vol. 43. – P. 143-157. DOI: 10.1016/j.trc. 2014.02.009.

[6] Wang, F. A Survey on Automatic Image Annotation and Trends of the New Age // Procedia Engineering. – 2011. – Vol. 23. – P. 434-438.

[7] Gotovac, S. Analysis of saliency object detection algorithms for search and rescue operations / S. Gotovac, V. Papić, Ž. Marušić // 24th International Conference on Software Telecommunications and Computer Networks (SoftCOM), 2016. – P. 1-6.

[8] Ilyasova, N. Particular Use of BIG DATA in Medical Diagnostic Tasks / N. Ilyasova, A. Kupriyanov, R. Paringer, D. Kirsh // Pattern recognition and image analysis. – 2018. – Vol. 28(1). – P. 114-121.

[9] Gershenson, C. Artificial Neural Networks for Beginners // Networks. – 2003. – Vol. 0308. – P. 8.

[10] Kratochvil, M. The move to store images in the database [Electronic resource]. – Access mode: https://www.oracle.com/technetwork/database/database-technologies/multimedia/overview/why-images-in-database-1-134712.pdf. (01.06.2018).

[11] Vester, J. Can I Please Store Images in the Database Now? [Electronic resource]. – Access mode: https://dzone.com/articles/can-i-please-store-images-in-the-database-now. (01.06.2018).