Binary data storage, retrieval and feature calculation on Netzeza PureData system for Analytics

Y Donon¹, R Paringer¹,² and A Kupriyanov¹,²

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

e-mail: yann.donon@ssau.ru

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 Netzeza 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 Netzeza 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
Netzeza 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. Netzeza doesn’t support alternative as MEDIUMBLOB or LONGBLOB for the storage of large binary data objects.

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. Data providers
Data providers are software components establishing a bridge between relational data sources and data exploitation layers in non-real time environments (IBM, 2019). It is in charge of mapping database’s physical table columns to application’s logical columns definition. Netezza offers data providers drivers for four architecture, ODBC, JDBC, OLE DB and .NET.

Tests were realized using different drivers in order to confirm our assumptions, but always using a Microsoft platform in order to normalize our approach and minimize noises, therefore the JDBC (Java Database Connectivity) architecture was excluded from the experiment.

2.1.1. ODBC
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 for the entirety of the experiments. (Microsoft, 2019)

2.1.2. OLE DB
OLE DB is a replacement for ODBC supporting a wider range non-relational database, it is based on a consumer-provider architecture, in opposition of ODBC’s client-server’s (Kyle, 1995). Concretely, OLE DB is a modern replacement solution for ODBC offering a wider range of applications (Blakeley, 1996). The difference in terms of performance, both drivers are difficult to compare, with regularly contradictive results (Li, 2009).

2.1.3. .NET framework
Microsoft’s .NET framework for software development implements the CLR for the execution (Common Language Runtime), the FCL (Framework Class Library) and the .NET framework Data Provider (SqlClient). .NET uses a managed client and middle-tier applications architecture. (IBM, 2019) (Microsoft, 2019)

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

Figure 1. Database Logical Data Model.

2.2.1. 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.

2.2.2. 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.
2.2.3. 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.

2.2.4. 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 (N.L. Kazanskiy, 2014). Features are any of the numerical characteristics calculated from the images such as: texture features (E. Biryukova, 2016), geometric features (R.A. Paringer, 2011), histogram features (Pratt, 1991), features of the spatial spectrum (N.S. Kravtsova, 2017).

2.2.5. 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 (Z. Zheng, 2011), annotation (Wang, 2011) of images, or object detection (S. Gotovac V., 2016) of object on an images. They are non-numeric characteristics.

2.2.6. 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 (N. Ilyasova, 2018)), or when annotated (for neural networks (Gershenson, 2003)).

2.3. 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 (Vester, 2018). Of course, we can temperate this as the price of the storage in DBMS, for example, still higher than in file systems (Kratochvil).

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 (Vester, 2018).

2.4. 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 \( \sum_{n=1}^{a} \left( \frac{2 \ln n}{a} \right) = P^2 \), in our case 963 bytes, that has been rounded up to 1000. This formula represents the intersection between 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 second 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 characters) | 500 | 1000 | 2000 | 3000 | 4000 |
|----------------------------------|-----|------|------|------|------|
| Time estimation ODBC            | 1,088 | 1,000 | 1,004 | 1,038 | 1,002 |
| Time estimation OLE DB          | 1,087 | 0,994 | 0,998 | 1,021 | 1,022 |
| Time estimation .NET            | 1,108 | 1,001 | 1,006 | 1,028 | 1,052 |
| Coefficient of variation, % ODBC | 0,039 | 0,106 | 0,041 | 0,046 | 0,054 |
| Coefficient of variation, % OLE DB | 0,072 | 0,072 | 0,108 | 0,089 | 0,091 |
| Coefficient of variation, % .NET | 0,042 | 0,036 | 0,089 | 0,070 | 0,051 |

For all of the 100 repetitions tested, the calculation time was registered in order to establish an average (V.I. Protsenko, 2016). From this list the coefficient of variation was calculated for each method as shown on table 1. This didn’t allow us to perceive any significant scheme, thus affirming that a fragment size or driver had more volatility in terms of data retrieval.

However figure 2 highlights a more informative feature of table 1, all methods have an estimation of feature calculation tendency at the lowest while working with 1000 bytes fragments, confirmed by the tendency lines, which is conveniently close our suggest fragment size based on the dataset features of 963 bytes. Thus allowing to encourage further the selection of this size for storage. This graphs also shows the .NET data provider being slightly slower than the two others, however in this study, given its scope, cannot extends this information to more than a contextual feature.
4. Conclusion
The structure of database presented allows storing large sets of images and the results of their processing in an optimal way. Applications 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, independently from the data provider used.

5. References
[1] Blakeley J A 1996 OLE DB: A Component DBMS Architecture (Redmond WA: Microsoft Corporation)
[2] Biryukova E, Paringer R and Kupriyanov A V 2016 Development of the effective set of features construction technology for texture image classes discrimination CEUR Workshop Proceedings 1638 263-269
[3] Gershenson C 2003 Artificial Neural Networks for Beginners Networks 8
[4] IBM 2019 NET architecture Retrieved from IBM Knowledge center URL: https://www.ibm.com/support/knowledgecenter
[5] IBM 2019 Data Provider Definition URL: https://www.ibm.com/support/knowledgecenter
[6] Kratovich M The move to store images in the database (Piction, Oracle)
[7] Kyle G 1995 Inside ODBC (Redmond, WA: Microsoft)
[8] Li C 2009 Database Adapter and Connector Encyclopedia of Database Systems (Springer)
[9] Microsoft 2019 What Is an ODBC Driver? URL: https://www.progress.com/faqs/datadirect-odbc-faqs/what-is-an-odbc-driver
[10] Microsoft 2019 Microsoft OLE DB Provider for ODBC Overview URL: https://docs.microsoft.com
[11] Ilyasova N, Kupriyanov A, Paringer R and Kirsh D 2018 Particular Use of BIG DATA in Medical Diagnostic Tasks Pattern Recognition and Image Analysis 28(1) 114-121
[12] Kravtsova N S, Paringer R A and Kupriyanov A V 2017 Parallel implementation of the informative areas generation method in the spatial spectrum domain Computer Optics 41(4) 585-587 DOI: 10.18287/2412-6179-2017-41-4-585-587
[13] Kazanskiy N L, Protsenko V I, Seraphimovich P G 2014 Comparison of system performance for streaming data analysis in image processing tasks by sliding window Computer Optics 38(4) 804-810
[14] Paringer R A, Kupriyanov A 2011 Methods For Estimating Geometric Parameters of The Dendrite’s Crystallograms Proceedings of 8th Open German-Russian Workshop “Pattern Recognition and Image Under-standing” 226-229
[15] Pratt W 1991 Digital Image Processing (California: Wiley)
[16] Protsenko V I, Seraphimovich P G, Popov S B, Kazanskiy N L 2016 Software and hardware infrastructure for data stream processing CEUR Workshop Proceedings 1638 782-787
[17] Gotovac S 2016 Analysis of saliency object detection algorithms for search and rescue operations 24th International Conference on Software Telecommunications and Computer Networks (SoftCOM) 1-6
[18] Vester J 2018 Can I Please Store Images in the Database Now? URL: https://dzone.com/articles/can-i-please-store-images-in-the-database-now
[19] Wang F 2011 A Survey on Automatic Image Annotation and Trends of the New Age Procedia Engineering 434-438
[20] Zheng Z 2011 Short-term traffic volume forecasting: a k-nearest neighbor approach enhanced by constrained linearly sewing principle component algorithm Procedia Engineering 434-438

Acknowledgments
This paper contains results of the project Geohub and multimodal multi-structure storages, which is carried out within the framework of the Program “Center of Big Data Storage and Analysis” of the National Technology Initiative Competence Center. It is supported by the Ministry of Science and
Higher Education of the Russian Federation according to the agreement between the M.V. Lomonosov Moscow State University and the Foundation of project support of the National Technology Initiative from 11.12.2018, No 13/1251/2018.