International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Binary Data Packing Method for Database Optimization

Yevgeniy Borodavka[1], Mykola Tsiutsiura[2]

[1, 2] Kyiv National University of Construction and Architecture, 31, Povitroflotskiy Ave, Kyiv 03680, Ukraine

Abstract: ***Modern databases which are used in many object oriented applications have a large amount of data. Due to this fact the databases optimization problem is topical. In this paper we propose a method for a database optimization. The method based on the binary data packing. The method allows reducing a number of tables in a database and encrypting data at the same time. As example we consider the database fragment designed for our own CAD application. Two type of the binary data packing were considered – a simple and a complex. The advantages and the shortcomings of the method were considered. The results of the method usage are smaller size of database and faster data processing.***

Keywords: binary data packing, database, optimization, encryption**.**

1. Introduction

Nowadays it is too hard to imagine any enterprise or CAD application without database using. Today we have a lot powerful solutions from many developers [1]. On the other hand most popular developing paradigm is the Object-Oriented Modeling (OOM) and Programing (OOP) [2]. Many developers are faced with problem of reliable and convenient mapping objects to relational databases (OR/M). There are several basic methods to do the mapping but each of them has some disadvantages and restrictions [3]. We try to combine different mapping strategies and after that to optimize database structure with suggested method.

2. The Problem

The ideal structure of database should to satisfy the follows:

- The small size
- Fast data access
- Low space wasting
- Simple restoring of the objects connections.

None strategies which were described in [3] satisfy all these conditions. We can try to use combinational method – for different parts of the database apply different strategies, but it only will make database is fragmented. Each fragment of the database will be close to ideal state but in general whole database will not satisfy to all conditions of ideality.

For enterprise application the most popular strategies are "map hierarchy to a single table" and "map each class to its own table" [3]. The disadvantage of first one is the space wasting; the disadvantage of second one is the great amount of tables in a database and as a result slow data access. But on the other hand the first strategy counteracts disadvantages of the second one – data access time is very fast. And the second strategy counteracts the first one – the space wasting is lowest. It is sounds logically to mix up these two strategies and using some kind of the technology with a purpose of removing disadvantages of both strategies.

There is some opinion about database structure: "the more tables the better, because in this case the database is well structured". Another opinion about the database structure tells us that the single table in the database is a bad style. Now it is obvious both these opinions are not correct in general and the truth somewhere between.

As an example we consider the object model (Fig. 1).

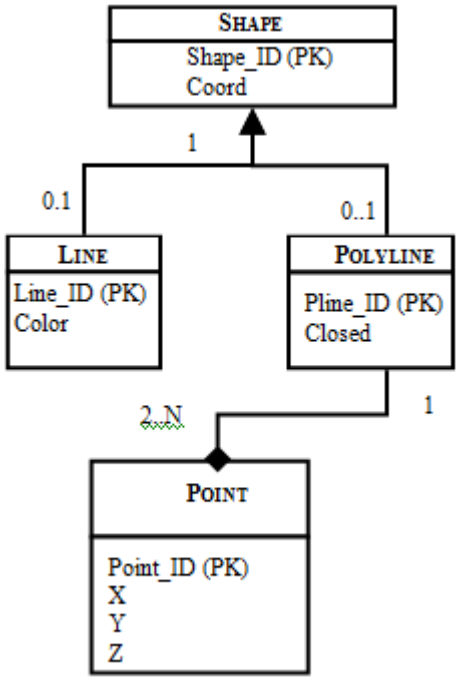

Figure 1: Class diagram of the example object model

If we use strategy "map hierarchy to a single table" we will have well-structured database and for three classes the number of the tables is not critical. But if we extend the object model the number of tables will grow dramatically. The efficiency of database will go down and time consumption for data access will increase as a result of the huge number of the tables and relations between its.

In our example we have aggregation relation, so we can try to find out the way to join two classes in one table, but not in

two. And this way should be universal and acceptable for using in any of such kind of cases.

According to class diagram (Fig.1) each object "Polyline" occupied one row in correspond database table and aggregate some amount of the "Point" objects (from 2 to N) which take place in another table. This is classical approach. In this case we need to add foreign key field to points table intended for keeping relation "one-to-many" (Fig. 2).

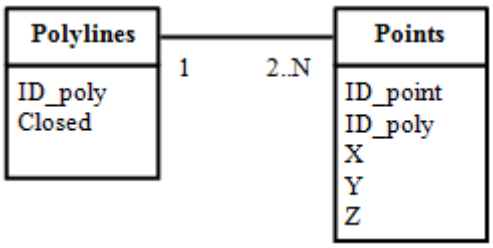

Figure 2: "Polylines" and "Points" tables' structures

3. The Method of Optimization

3.1 Direct Approach

Let's try to write points' coordinates in the one field of the table "Polylines". The easiest way is to create text field with maximum length and saving all points' coordinates there using some kind of a separator. For example it could be implemented as shown in Tab. 1. There symbol "," is used as separator between coordinates X, Y, Z and symbol "|" is used as separator between points.

Table 1: Fragment of the table "Polylines" with additional text field

POLYLINES		
ID	**Closed**	**Points**
1	false	23,45,-25|-34.6,65.4,0|85.3,76.9,-45.2
2	true	-3.4,0.5,56|12.4,66,0.3|83,-6.2,25

The implementation is presented above has some disadvantages. Firstly, it can be done only for simple structure of aggregated table. If aggregated table has more complicated structure this approach doesn't work, because the time consumption for complex structure parsing will exceed all benefits. Secondly, this approach leads to space wasting. For example, to save one negative two-digit number with one digit precision, we need to use at least 5 byte of information for ANSI or UTF-8 text coding. If we use UTF-16 it will be twice more. Thus, in case of six-digit numbers, data size will be inadequate to necessary, because for description of wide variety of float numbers 4 byte is enough.

The approach was described above is suitable in limited cases only. For example, in case when saving of positive two-digit numbers array is needed.

3.2 Simple Binary Data Packing

The more sophisticated approach to data packing is using binary field to packing whole table. Modern databases grant up to 2 gigabyte information to be saved in binary field. In the most cases it is enough.

The advanced approach pack the table on the "many" side in one binary stream and save to single binary field of the table on the "one" side. For example at Fig. 2 the rows of the table "Points", which correspond to single polyline, should be packed into binary stream and saved in proper fields (Tab. 2).

Table 2: An example of binary packaging

POLYLINES			POINTS				
ID_PL	**Points**		**ID_PL**	**ID**	**X**	**Y**	**Z**
1	BLOB		2	1	-34	2.4	7.2
2	**BLOB**	←	2	2	-1.2	35	4.6
3	BLOB		2	3	0.3	-2.5	2.7
			2	4	4.4	3.5	-4.2
			2	5	0.3	-2.3	-3.7

The example above show how the polyline which consist of 5 points can be represented in one table row. The polyline with *ID_PL=2* has special binary field which contains 5 rows of the "Points" table.

To use this approach each object must have methods for packing and unpacking itself to/from binary stream. Any OR/M class already has methods for mapping to database, so it only needs to improve these methods according to presented approach. The binary data packing should be used carefully and correctly – the sequence of write and read must be strongly followed. If it is not kept and the sequence is shifted even on 1 byte then all information is read after it will be invalid.

The considered approach to binary data packing is useful only for object in relation "aggregation". In this case all aggregated object are saved in binary field of aggregator table. The binary field of each row has the same structure and processed by the same methods. This type of the binary data packing we call *simple*.

3.3 Complex Binary Data Packing

Now let is considering top part of the example class diagram (Fig. 1). There we have hierarchy structure with inheriting. Abstract class "Shape" is the parent for concrete classes "Line" and "Polyline". Thus, each row in the "Shapes" table can correspond to the different child tables.

For the extending the binary data packing to this hierarchy structure we need add 2 fields to the parent table – one for the child object type saving and another binary field for the child object data saving. In this case each row in the parent table will have the different structure depend of the child object type (Fig. 3).

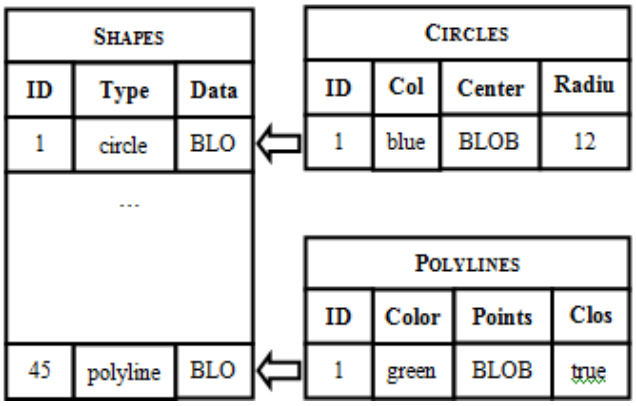

Figure 3: The strategy of the hierarchy binary data packing

As the simple type of the binary data packing, current modification is demanding each object must have methods for packing and unpacking itself to/from binary stream. The management of whole data processing is making by the parent class. It calls proper method of the child class depends of the object type in correspond field. This type of the binary data packing we call *complex* or *hierarchical*.

In general case it is possible the situation with deep hierarchy of objects and the aggregation relations at the same time. Such difficult models can include a lot of objects with the binary field and some binary fields can include other objects with binary fields and so on. But whole hierarchy will be processing automatically with using inheritance and polymorphism of OOM.

4. Conclusion

We suggest the method of database optimization is based on the binary data packing. There are two type of this method implementation – simple and complex. The advantages of the method are:

- Database size reduction;
- Data access speed enhancing;
- Complex hierarchy object read and write simplification;
- Automatic data encryption, because without knowledge about the binary data structure it is impossible to get access to the data.

The main shortcoming of the method is that the binary fields cannot be used in SQL statements. Thus, only not indexed data may be packed by presented method.

References

[1] T.M. Connolly, C.E. Begg, Database Systems: A Practical Approach to Design, Implementation, and Management, Pearson Education, 2005.
[2] S.W. Ambler, The Elements of UML(TM) 2.0 Style, Peperback, 2005.
[3] S.W. Ambler, "Mapping Objects to Relational Databases: O/R Mapping In Detail". Aviable: http://www.agiledata.org/essays/mappingObjects.html. [Accessed: Oct. 31, 2014]

Authors Profiles

Mr. Yevgeniy Borodavka received B.Sc. and M.Sc. degrees in computer science and Ph.D. degree in computer aided design from Kyiv National University of Construction and Architecture, Ukraine, in 2002, 2003 and 2009, respectively. Since 2005 assistant professor and since 2009 associate professor of the Information Technologies of Design and Applied Mathematics department in Kyiv National University of Construction and Architecture, Ukraine. Since 2004 to 2009 – lead engineer and since 2009 to 2012 – senior scientific staff in State Enterprise “State Research and Development Institute of Computer Aided Design in Construction”. Since April 2013 is lead engineer in Samsung Research and Development Institute Ukraine (SRK).

Mr. Mykola Tsiutsiura, Since 2012 is master in "Information control systems and technologies." Since 2013 is master in "Project Management". Since November 2012 is a graduate student of the Kiev National University of Construction and Architecture in the direction of information technology and project management. Since 2013 is assistant professor Information Technologies department in Kyiv National University of Construction and Architecture, Ukraine.