IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP5042

Extending UML for Database Design

Luiz Camolesi Jr.
Methodist University of Piracicaba - UNIMEP, Rod.do Açúcar 156, Piracicaba - SP - Brazil, E-mail: lcamoles@unimep.br

Edson Luiz Avanzi
Methodist University of Piracicaba - UNIMEP, Rod.do Açúcar 156, Piracicaba - SP - Brazil, E-mail: eavanzi@dglnet.com.br

ABSTRACT

The database design is made through some contexts from the conceptual level to the physical level. Nowadays many tools are available to assist the designers with the projects but UML (Unified Modeling Language) has become a standard language to describe the complete development of relational and object relational databases. However, some important aspects about the performance of database systems, on the physical level, are not enough expressed in UML. In this article we approach aspects such as indexing, file organization and the database server-computer configuration to propose an extension of the UML for the database physical design.

1. INTRODUCTION

With the exponential growth of information in several areas of business and knowledge, databases are bigger and more complex causing the enterprises to search for an adequate planning from a database engineering. The companies wish have many data but they want the information available within the market time. In addition to ERP (Enterprise Resources Planning), BIS (Business Intelligence System), MIS (Management Information System) and Data Warehouse systems that require a lot available space in disks, Internet brought new data types which have been stored on database, too. Then, the physical design to database systems needs to include aspects of performance to access the data, mechanisms for representing the file indexing, the table organizations and all the environment such as data storage technologies, database and application servers. The development of a good project of database system is important so that the data may be stored and recovered in an adequate way.

The project of a database system [8] is made through some contexts, which include from the highest level (conceptual) which represent the vision of the users about the data to the lowest level (physical) which represents how data will be stored and recovered on database.

Nowadays many technologies are available to assist the designers and architects to project the database and one of them is the UML [9] [11]. The UML has emerged as an effective modeling tool for database modeling and database design process [17]. It can facilitate the integration of database models with the rest of a system design [13].

This article proposes the database physical project in relational database using the UML. The rest of this paper is organized as follows. In section 2 we outline the database models. In section 3 we identify the deficiencies that can be overcome by using UML extension mechanisms. In section 4 and 5 we illustrate one possible technique for modeling a relational database in the UML. Some experience with relational database modeling as well as some familiarity with the UML is assumed.

2. DATABASE MODELS

In the conceptual project occurs the description about the contents on database but not of the structures of data. It is necessary to interview the potential users of system to describe the information relevance and data purposes and constraints. The result is the conceptual model presented with the modeling of the data such as ER (Entity-Relationship) model [2] [14].

The ER model consists of a collection of entities which are related by the ER diagram. The ER diagram do not consider the physical structures of databases assuming that this scheme will be used upon a model, as relational model which uses several two-dimensional tables to represent both data and the relationships among these data. This model has been selected for the implementation of many a database.

Although the IDEF1X [3] has become a standard for data modeling in relational databases it is mostly used for database logical design but it is not the best choice for non-relational system implementations.

In this way, the objective in the logical project is to translate the conceptual scheme to the database describing the structures of data in the relational scheme. Besides the CASE tools which were and continue being used by some designers to develop the logical model we can also count on the UML [15] [16].

In the physical project the logical model is converted in to structures of storage and the access paths to the files are detailed. The objective is to reach the performance goals.

The UML has elements that represent both conceptual and logical aspects of the databases [1] [12]. It represents a relationship as a dependence of any type between two tables using the stereotyped association and including a set of both primary and foreign keys. It can represent overall physical structures of the database and use a stereotyped component to represent a physical database. But it doesn´t detail the file indexing, type of index, table organizations and block sizes that are factors important to improve the performance of the database.

3. DATABASE PHYSICAL PROJECT

The database physical project involves the study and the knowledge of the physical features of the database to reach the strategies for accessing the data efficiently. These include the storage structures, index structures for files that enhance the search for and retrieval of records improving the performance and the form of file organization.

Most of the DBMS has flexibility for modeling of the elements offering some options as follows.

3.1 Indexing

During the physical project the best strategies must be chosen get fast access to records in a file on disk using an index structure [2] [7] [14]. The indexes speed up the recovery of records based in some conditions used for the search and they can use alternative paths to access the records without changing there of place. The most used types of indexes are based on ordered files and tree data structures.

- Ordered indexes

The most common types of ordered indexes are the primary, clustering and the secondary indexes.

The primary index is an index whose search key defines the sequential order of the file on disk and the search usually occurs using the primary key. Primary indexes can be dense or sparse. While a dense index has an index for every search key in the data file whereas the sparse index has index entries for only some of the search key. The clustering indexes look like the primary indexes but they are used when numerous records have the same value for the ordering field.

In the secondary index the records are not stored sequentially in the same as that of the primary index, being necessary to contain pointers to all the records. Although this improves the performance for the queries on non-primary keys it implies in performance degradation when some alteration of the database will be necessary.

This conference paper appears in the book, Innovations Through Information Technology, edited by Mehdi Khosrow-Pour. Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

• Tree data structures

As the file grows the performance of the ordered indexes goes degrading. This problem can be partially solved by periodic reorganization of entire file but it is not a good idea because during the reorganization no user can access the system or, that is, the system will need be unavailable. The alternatives for this situation are the tree data structures. The basic types of tree data structures are the B^+-Tree and B-Tree.

The B^+-Tree is an index structure in which every path from the root of the tree to the leaf of the tree is of the same length seeming a balanced tree.

The B-Tree is similar to B^+-Tree but allows search key values to appear only once eliminating redundant storage of search keys. Its number of nodes is reduced with the search key value being found ahead of the B^+-Tree. On the other hand, only a small piece of the search key values are found early and its implementation is normally more complicated than B^+-Tree.

3.2 File organization

The file organization [2] [7] [14] implies in the organization of the data of the file into records, blocks, and access structures including the way records and blocks are physically placed on the disk. The designer must choose a file organization that increases the systems efficiency upon records retrieval.

The main ways to organize the records in files are: unordered, ordered and hashed.

• Unordered (heap files)

This is the simplest and basic type of file organization and is also known as heap files. The records are placed anywhere where there is space, usually they are inserted to the end of file. There is no ordering of records. Typically, the deleting of rows create gaps in file which result in fragmentation, decrement of space on disk and decreasing of performance.

• Ordered (sequential files)

The sequential file organization or ordered records allows that records be stored in a sorted order based on the same search key of each record where this key does not need to necessarily be a primary key. In this organization type the file must be periodically reorganized to keep the sequential order on the disk.

• Hashed

In the hashed file organization it is used a hash function to determine the block in the file where the record can be stored. In this way the access to the block will be immediate without the access necessity to an index structure.

3.3 Data blocks

The different files that exist on a database are partitioned into storage units, known as data blocks [2] [7] [14] and may contain several data. In the physical storage the data are organized in terms of data blocks, extents and segments. The smallest level of granularity of storage is a data block which is of a fixed length and typically it is a multiple of the operating system block size.

The form of physical data organization determines the set of data that a block contains. The correct choice of the block size is important when reading and writing rows on the tables. If there are more than one server, the data blocks are defined in each server. As a baseline for performance, we also propose the definition of the block size in the UML extensions.

4. EXTENDING UML FOR DATABASE PHYSICAL DESIGN

This article proposes the Table 1 below for the extension of the database physical design using the UML. An advantage of using it is that it reduces the complexity in large database design [6]. We know that more indexes [2] [7] exist, therefore, this table is not complete and other indexes can be added. Some alternatives to use the Table 1 are:

Table 1. Table for stereotypes and icons

		Stereotype	Icon
Indexes	Primary Dense	<<PD>>	
	Primary Sparse	<<PS>>	
	Clustering	<<CS>>	
	Secondary	<<SC>>	
	B-Tree	<<BT>>	
	B^+-Tree	<<BT^+>>	
File Organization	Unordered	<<heap>>	
	Ordered	<<sequential>>	
	Hashed	<<hash>>	

- Only an icon to indicate which file organization it is;
- Default icon in the UML and of the stereotype to indicate which file organization it is;
- Only the default icon in the UML and to assume that the file organization is the DBMS default organization.

The Figure 1a shows an example of a Class in which we can see in details the file organization, the organization key and the indexing. In the class name there appear a stereotype and an icon that can be used to represent both indexes and file organization by using the UML. The icon in the class name indicates that the file organization is sequential, ordered by the TITLE key. Some file organizations do not need a key specification (as heap) whereas in others the specification is required. Then, in the database project the designers must be careful with the specifications of the keys (both simple and composite), when necessary. The attributes AUTHOR and ISBN are identified as indexes B^+-Tree (BT^+) which are being indexed by the ordered index IX_AUTHOR( ) and IX_ISBN( ).

Figure 1a. An example of a Class

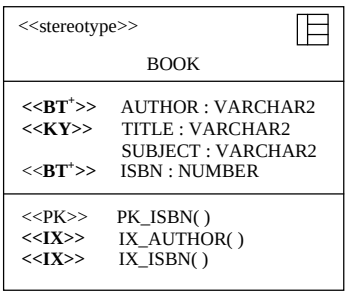
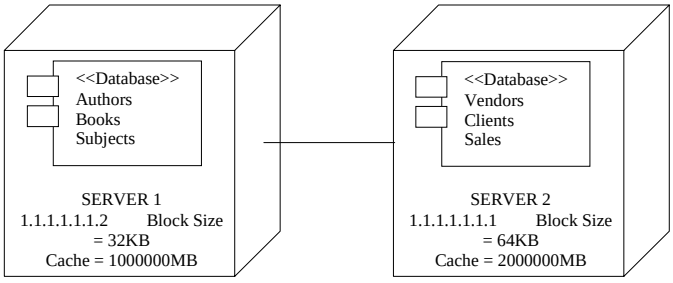
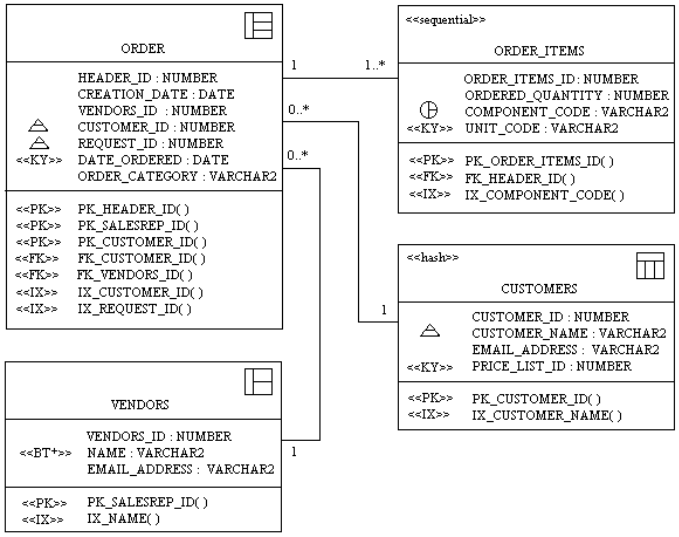


Figure 1b. An example of Component diagram



Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

Figure 2. An example of modeling



The Figure 1b shows part of a Component diagram in which is described the block size. Each server can have a different block size therefore, the block size is defined in each server for best performance when processing. This diagram also shows cache value which is another significant parameter for the system performance.

For the class with indexing the stereotype <<IX>> identifies a name (or method) of the used indexing. Many an indexing may exist for a class. The stereotype <<PK>> is obligatory in all classes which is used to identify the unique default indexing of the primary key. The stereotype <<IX>> identifies the key which is used in the organization of the class. There must be only one organization key which can be simple or composite by several attributes.

5. MODELING EXAMPLE

The Figure 2 illustrates a part of the table collection used for the management of order entry in some information systems. The tables Order, Order_Items, Vendors and Customers can be specified according to Table 1 facilitating the implementation on the physical level. Moreover, the UML will thus detail the structures of storage and access paths to the files with every information about the overall project being registered.

In Figure 2, the ORDER table has an icon in the class name which indicates that the file organization is sequential and it is ordered by the DATE_ORDERED key. The CUSTOMER_ID and REQUEST_ID have icons indicating that they are indexes B^+-Tree and they are indexed by IX_CUSTOMER_ID and IX_REQUEST_ID, respectively, according to stereotype <<IX>>. Besides, there are primary keys and foreign keys indicated by stereotypes <<PK>> and <<FK>>. The ORDER_ITEMS table is a sequential file organization indicated by the stereotype <<sequential>> and it is ordered by the UNIT_CODE key. The clustering indexing is indicated by the icon in the COMPONENT_CODE which is indexed by the IX_COMPONENT_CODE, according to stereotype <<IX>>. The clustering indexing was chosen in order that the components can be grouped by items. The VENDORS table is a heap file organization indicated by an icon without having the organization key. The CUSTOMERS table is a hash file organization. The organization key is the PRICE_LIST_ID and the CUSTOMER_NAME is an index which is indexed by IX_CUSTOMER_NAME index.

6. CONCLUSION

Nowadays the information systems can be distributed for many places and there are several data types and the database are becoming bigger and bigger. The Unified Modeling Language is an interesting and optimal tool for modeling and database design [12]. The physical project is not usually accomplished causing problems of performance in overall system. This paper proposes a set of extensions using the UML for database physical design so that the designers have more elements to architect the systems and represent pieces of important information to improve the performance of the database system [4]. These contributions are more relevant when applied to large databases such as multimedia databases because in these case the performance is more significant and the design is more complex [5].

This work tries to keep the UML conception bases and the simplicity and legibility of the extended diagrams. The UML 2.0 [10] brings significant improvements and the profile package upon which this work can be continued.

7. REFERENCES

[1] BJÖRKANDER, Morgan; KOBRYN, Cris. “Architecting Systems with UML 2.0”, IEEE Computer Society, July/August 2003.

[2] ELMASRI, Ramez; NAVATHE, Shamkant B. “Fundamentals of Database Systems, 3rd ed., Addison-Wesley, 2000.

[3] IEEE Computer Society. “IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEF object)”, IEEE Std 1320.2-1998, December 2003.

[4] KIVISTO, Kari. “Roles of Developers as Part of a Software Process Model”, In Proceedings of the 32nd Hawaii International Conference on System Sciences – 1999.

[5] MARTIN, Grant. “UML for Embedded Systems Specification and Design: Motivation and Overview”, In Proceedings of the 2002 Design, Automation and Test in Europe Conference and Exhibition (DATE’2002), IEEE Computer Society.

[6] MOK, Wai Yin; PAPER, David P. “On Transformations from UML Models to Object-Relational Databases”, In Proceedings of the 34th Hawaii International Conference on System Sciences – 2001.

[7] MOLINA, Hector Garcia; ULLMAN, Jeffrey D., WIDOM, Jennifer. “Database System Implementation”, Prentice Hall, 2000.

[8] MULLER, Robert J. “Database Design for Smarties”, Morgan Kaufman, 1999.

[9] NAIBURG, Eric J.; MAKSIMCHUK, Robert A.. “UML for database design”, 1st edition, Addison-Wesley Pub Co, 2001.

[10] OMG. “UML 2.0 Infrastructure Final Adopted Specification”, OMG document ptc/03-09-15, Object Management Group, 2003.

[11] OMG. “UML 2.0 Superstructure Final Adopted Specification”, OMG document ptc/03-08-02, Object Management Group, 2003.

[12] Rational Software. “The UML and Data Modeling”, Rational Software Corporation, http://www.rational.com/media/whitepapers/Tp180.PDF, November, 2003.

[13] SELONEM, Petri; KOSKIMIES, Kai; SAKKINEN, Markku. “How to Make Apples from oranges in UML”, In Proceedings of the 32nd Hawaii International Conference on System Sciences - 2001.

[14] SILBERSCHATZ, Abraham; KORTH, Henry F., SUDARSHAN, S.. “Database System Concepts”, 4th ed., McGraw Hill, 2002.

[15] SPARKS, Geoffrey. “Database Modelling in UML”, In Methods & Tools e-newsletter. http://www.martinig.ch/mt/index.html

[16] TERRASSE, Marie-Noëlle; SAVONNET, Marinette; BECKER, George. “A UML-based Metamodeling Architecture for Database Design”, In Proceedings of the IEEE International Database Engineering and Applications Symposium (IDEAS’2001).

[17] WHITE, Stephanie; CANTOR, Murray; FRIEDENTHAL, Sanford; KOBRYN, Cris; PURVES, Byron. “Panel: Extending UML from Software to Systems Engineering”, In Proceedings of the 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS’2003).

Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

0 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/proceeding-paper/extending-uml-database-design/32452

Related Content

Prediction System-Based Community Partition for Tuberculosis Outbreak Spread
Fatima-Zohra Younsiand Djamila Hamdadou (2022). *International Journal of Information Technologies and Systems Approach (pp. 1-20).*
www.irma-international.org/article/prediction-system-based-community-partition-for-tuberculosis-outbreak-spread/289998

Organizational Knowledge Sharing and Enterprise Social Networks: A Higher Education Context
Niall Corcoranand Aidan Duane (2019). *Educational and Social Dimensions of Digital Transformation in Organizations (pp. 78-114).*
www.irma-international.org/chapter/organizational-knowledge-sharing-and-enterprise-social-networks/215138

Communities of Practice from a Phenomenological Stance: Lessons Learned for IS Design
Giorgio De Michelis (2012). *Phenomenology, Organizational Politics, and IT Design: The Social Study of Information Systems (pp. 57-67).*
www.irma-international.org/chapter/communities-practice-phenomenological-stance/64677

An Efficient Source Selection Approach for Retrieving Electronic Health Records From Federated Clinical Repositories
Nidhi Guptaand Bharat Gupta (2022). *International Journal of Information Technologies and Systems Approach (pp. 1-18).*
www.irma-international.org/article/an-efficient-source-selection-approach-for-retrieving-electronic-health-records-from-federated-clinical-repositories/307025

Discovery of Sequential Patterns Based on Sequential Interestingness and Constraint Conditions
Shigeaki Sakurai (2015). *Encyclopedia of Information Science and Technology, Third Edition (pp. 1756-1766).*
www.irma-international.org/chapter/discovery-of-sequential-patterns-based-on-sequential-interestingness-and-constraint-conditions/112581