IJCSC 0973-7391 Volume 7 • Number 1 Sept 2015 - March 2016 pp.83-88 available online at www.csjournalss.com

Empirical Performance Comparison of OODBMS over RDBMS

Deepika Shukla , Neha Singh, Nidhi Jagudaniya, Ami Nirmal
Comp. Science and Engineering Department, Institute of Technology, Nirma University
deepika.shukla@nirmauni.ac.in

Abstract: the paper illustrates the emergence of Object-oriented databases as a better alternative to the traditional relational databases. Even though they are yet to gain a strong footing commercially, they have laid down a very firm foundation for a database management system that will efficiently cater to the needs of fully automated object oriented database management systems in the future. The paper compares empirically the performance of OODBMS and RDBMS. DB4O and SQLserver have been adopted as the candidate databases for the experiment purpose. The paper also describes the benefits of Object-oriented databases and in addition to this, it also addresses some concerns which are preventing them from gaining upper hand in the world of Database Management Systems.
Keywords: Object Oriented Database management system (OODBMS), Relational Database management system (RDBMS), Object Manager Enterprise, DB4O, SQL server, impedance-mismatch.

Introduction

Currently most of the software systems follow client-server architecture wherein they depend on one or the other database on server for storage of their data. These servers cater the requests of multiple clients connected to the server for their data needs. In today's world, most of these client server applications use a Relational Database Management System (RDBMS) as their data store and an object oriented programming language as development platform. As a result main programming construct is considered as 'objects' whereas database management system supports storage of data in form of relations and tuples. This brings in the inconsistency between storage model and programming model. i.e objects must be mapped to tuples while performing storage or retrieval of data by the application. This mapping induces indispensable performance penalty. OODBMS is the remedy to this performance penalty.

With the object model now in the picture, the developers aim to reduce the overhead of translating information representation in the database to an application specific depiction. Contrary to the traditional databases (using relational models); an object model facilitates data persistence and storage by storing objects in the databases. The relationships among various objects are ingrained in the structure of the objects. This is mainly applicable for complex data structures such as 2D and 3D graphics which must otherwise be flattened before they are eligible for storage in the relational databases.

An OODBMS combines object-oriented programming paradigm with traditional database management principles. That is, it has to inevitably support abstraction, encapsulation, polymorphism and inheritance and at the same time it should provide means for database management concepts such as the ACID properties (Atomicity, Consistency, Isolation and Durability) which lead to system integrity, support for a query language and secondary storage management systems which in turn allow for managing very large amounts of data.

Therefore, Classes, Objects, support for class hierarchy, complex objects, persistent object storage, overloading , overriding, secondary storage management, concurrency , recovery are some of the key concepts among others which are to be enforced by a database management system to be called as an OODBMS.

This paper discusses many aspects of Object Oriented Database Management System. After introducing the objective of the paper, manuscript talks about the way evolution has taken place in the area of database management system over the period of time. Though, introduction section discusses the objective of OODBMS , we fully establishes the need of OODBMS in the section after that. Next section throws some light on basic standards of an OODBMS. The paper also well discusses the characteristics of OODBMS and various application areas of OODBMS. Thereafter we have focused on Architecture of OODBMS. The main contribution of the paper is to present comparison between RDBMS and OODBMS and finally we draw conclusions and mention the future directions of work.

Evolution of Database Management Systems

Before discussing about OODBMS, we must have a brief glimpse of how the database management system evolved over time. The evolution of databases can be traced back to 1960's, where disks and drums acted as a means of information storage. After a decade it was felt that data should be made independent of the logic of the application program. The first ever databases were navigational in nature, where record pointers were used to access data by concerned applications. The record pointer was made to point to the record being accessed. This was the predecessor of the IBMs hierarchical model (IMS System) and network model. Following the aforementioned models, came the relational model which emphasized more on the storage content rather than on the links to retrieve the data and till date it is the most widely used type of database.

With the rapidly progressing industrial sector and the growing population the need to store a large variety of data surfaced. Soon it was found that the Relational models were somewhat limiting as they had a very rigid

structure which was to a great extent different from the real world scenario, also it provided no support for new types of data such as graphics, XML, 2D and 3D spatial data.
In the 1980s with the advent of Object Oriented methodologies and languages, the thought of meshing the database capabilities with object oriented programming language provided a new breakthrough in this domain, for this idea could provide data storage that was in line with the real world objects. This led to the development of Object Oriented Database Management Systems.
When we integrate the capabilities of a conventional database management system (atomicity, integrity, consistency, durability) with the OOP concepts, we get Object-oriented database management system. It is very different from the database using query sub-language like SQL or call interface used by ODBC and JDBC. It facilitates the integration of database along with the programming languages and thus, they provide a way of storing objects while encapsulating their behavior as well as methods associated with them in a single unit

Need and Standards For Object-Oriented Databases
The following enumerate the need of OODBMS:

- Process Integration: With the automation of many plants, there arises a need for process integration, which can be best handled by OODBMS.

- Complex Structures: Complex object structures like that of CAD/CASE applications which need to store digital circuits in a way that the information remains intact along with its respective operations, we require OODBMS. In no way can RDBMS serve this need.

- Storage: Some applications demand the need for storing classes as well as objects along with associations and methods. OODBMS is the most apt solution for this problem as it inherently stores the data in the form of objects that are nothing but attributes + methods.

OODBMS Standards
The Object Data Management Group has laid certain standards pertaining to data schema, programming language bindings, data manipulation and query languages required for the development of applications which sit well with object databases.
One of the core elements of the above mentioned standards is the Object Definition Language which is a standardized language to design the structure of the databases with respect to the object data model. It basically defines three components of the object oriented data model, Abstraction, Inheritance and Encapsulation, which is discussed later on in this paper.
The ODMG (Object Data Management Group) has also developed OQL (i.e. Object Query Language) which can be considered as a counterpart of SQL. It is also a declarative language just like SQL with features similar to SQL but with additional provisions meant for making interaction with the object databases simpler.

Characteristics and Application Areas of OODBMS
The Object-oriented databases comprises of the following characteristics:

A. Integrated development system:

As mentioned earlier it successfully manages to overcome the problem of "impedance mismatch" hence providing a single development environment in which the data storage model is maps with and equivalent to the programming model

B. Encapsulation:

OODBMS wraps up the object attributes along with the relevant methods in a single unit called the class.

C. Extendable data types:

Besides the commonly used data types like text, number, memo etc., OODBMS provides a way of readily handling complex data types like graphics, videos, BLOBs etc with utmost ease.

D. Inheritance:

A very useful feature provided by OODBMS for instance; if there are two objects Employee and Manager in RDBMS what one can do is enter the records of both manager and an employee along with their designation to differentiate between the two. In real world even the manager is an employee, but RDBMS stores the data that does not map the real world scenario. On the other hand with OODBMS we can create two classes Employee and Manager and let the Manager derive its primary functionalities from the Employee class through the feature of inheritance.

E. Persistence:

The object persistence refers to the capability of objects to preserve their state across different invocations. If this feature is not present the scope of an object's state lasts until the time application goes out of RAM. If one desires the objects' state to be preserved, persistence is the way.

Application Areas:

A. Computer-Integrated Manufacturing

CIM is employed by many a producers to achieve timely and supreme quality products. OODBMS provides process integration framework that allows individual processes to exchange information with each other hence enhancing the overall production process.

B. Advanced Office Automation Systems

OODBMS in such systems handles hypermedia very efficiently as there lays commonalities between database and the hypermedia model. With the direct implementation of many a data structures with utmost ease, OODBMS fits such applications like a glove.

C. Hospital Patient Care Tracking Systems

These applications include RFID-based patient tracking. The location of patients can be monitored constantly so as to prevent any harm to them. Object database plays a vital role in handling such a delicate system with its capability to work with a wide variety of data.

D. CAD, CAM, CASE

All the above mentioned systems i.e. Computer-Aided Design, Computer-Aided Manufacturing, Computer-Aided Software Engineering, have one thing in common i.e. complex graphical data that has to be stored, worked with, and even to be modified. This is where OODBMS comes into picture. As mentioned earlier it makes storing such complex structures very easy.

Architecture of OODBMS

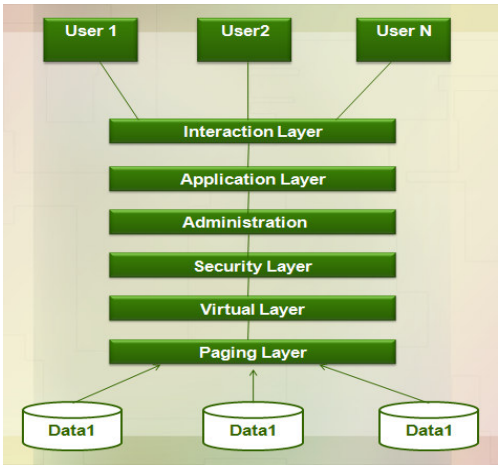

Figure: 1. Architecture of OODBMS

The architecture of OODBMS is divided into six layers [1], each with a functionality of its own Fig:1. Interaction layer is the top-most layer in the OODBMS architecture; it enables the users to interact with the database. Then follows the Application layer which acts as the interface between the user and the data store. The Administration layer which is the third layer in the architecture controls the flow of data as well as provides the permission to access certain data. The following layer i.e. the Security layer, is responsible to provide the full security to data and also provide to the security of application used to manage the data. The Virtual layer which is the next layer helps in managing large data. The Paging layer is the last layer of the architecture, is responsible for dividing the data into pages so that they can be managed easily.

OODBMS v/s RDBMS

The Table 1enumerates the points of difference between OODBMS and RDBMS. Basically the difference lies in the fact that RDBMS only allows the preservation of state while OODBMS retains not only the state of the object but also its behavior encapsulated in the form of a class [2].

Table 1: Dimensions differentiating RDBMS and OODBMS (adapted from [2])

Basis of Difference	OODBMS	RDBMS
Primary Objective	Data encapsulation and Data Independence	Data Independence from application programs
Reorganization	Classes can be reorganized without affecting the mode of using them	Data can be reorganized without affecting the mode of using them
Storage	Data and Methods	Only Data
Encapsulation	Data accessible through class methods only	No such provision
Adhesion	Data can be chained	No such provision
Representation	Class of objects directly model the real life applications	Application will have to be converted in accordance with the relational tables
Data access speed	Access to data is faster as no joins are needed	Joins are needed so data access is slower

When it comes to data independence in the relational databases, it mainly aims at protecting the applications from the changes in the storage structure and access mechanisms of the data. To realize the same, mainly views are used. But in reality no relational database provides full data independence because views are only used as shorthand in queries as they cannot be fully updated, so views only provide logical independence. While the advanced applications such as CAD/CAM would require being able to dynamically manage a database schema without losing the data. The schema change arises due to a number of reasons such as changed real world domain, reusability and extendibility of classes or cooperation with other systems. The solution to this is provided by OODBMS wherein we can propose a view language to define and update virtual schema enhancing view capabilities with schema evolution facilities for use as a uniform framework to manipulate both the schema and the database, therefore providing full data independence.

The next instance where OODBMS differs from RDBMS is the storage content; the RDBMS only stores data while OODBMS not only stores data but also methods to manipulate the said data.

If we were to compare the degree of encapsulation provided by OODBMS and RDBMS; RDBMS falls short of such facilities, OODBMS on the other hand by the virtue of its encapsulation feature, only allows class methods to access the data.

Another significant difference lies in the fact that OODBMS introduced data chaining whereas there is no such provision in RDBMS. And last but not the least OODBMS successfully bridges the gap between the real world model and the conceptual model while in case of RDBMS every real entity has to be mapped to a tuple, which results in “impedance mismatch”. Figure 2 shows the diagrammatic representation of the differences between RDBMS and OODBMS:

Benefits and Current Users of OODBMS

Thus, the benefits of Object-oriented database can be summarized as follows [3]:

- OODBMS provides persistent storage of objects
- Overcomes the “impedance mismatch” by providing a single development environment
- Enforces class hierarchy through inheritance
- Removes the stringent need for need for a query language as in case of RDBMS
- No concept of primary keys as each object is assigned a unique OID by the system itself.
- Easy navigation across multiple objects as no joins required.
- Data model closely resembles the real world entities.

- Works well for distributed architecture.

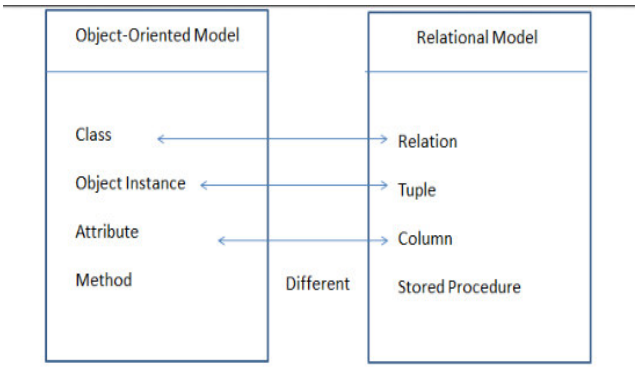

Figure 2. Points on which RDBMS and OODBM Differ

The organizations using OODBMS at a large scale are listed as follows. This exemplary list proves the potential of OODBMS in case of big commercial software system

- Chicago Stock Exchange
- Radio Computing Services
- Motorola in The Iridium System
- South-West Airline's Home Gate

Experiment and Results

For comparing both the databases empirically, instances of SQL server as RDBMS and DB4O as OODBMS were used. Here, for the sake of completion we present the procedure for installing the instance of DB4O. First step is to install a plug-in called DB4O. The DB4o distribution includes Object Manager Enterprise (OME) which allows you to browse your DB4O database in a graphical interface[10]. With OME you can check which objects were stored and examine the database structure. Using the OME ad-hoc query builder, you can create queries to validate business logic in your application or handle simple reporting requirements. The step that follows next is adding the said plug-in to Eclipse IDE and the following are the steps to do so[11]:

- create a folder named "lib" under your project directory, if it doesn't exist yet.
- copy db4o-*.jar to this folder.
- Right-click on your project in the Package Explorer and choose "refresh".
- Right-click on your project in the Package Explorer again and choose "properties".
- select "Java Build Path" in the tree view on the left.
- select the "Libraries" tab page.
- click "Add Jar".
- the "lib" folder should appear below your project.
- choose db4o-*.jar in this folder.

Insert, simple select , select with where clause, update and delete were fired on both the databases.

TABLE 2: Statistical Comparison Of OODBMS & RDBMS

QUERY	EXECUTION TIME: OODBMS	EXECUTION TIME: RDBMS	Approx No. of times OODBMS is faster that RDBMS
Insert	13ms	120ms	9.23
Retrieval	43.2ms	165ms	3.81
Retrieval(where clause)	20.6ms	321ms	15.58
Update	2.6ms	250ms	96.15
Delete	7ms	126ms	18
Average Speed up achieved			28.55

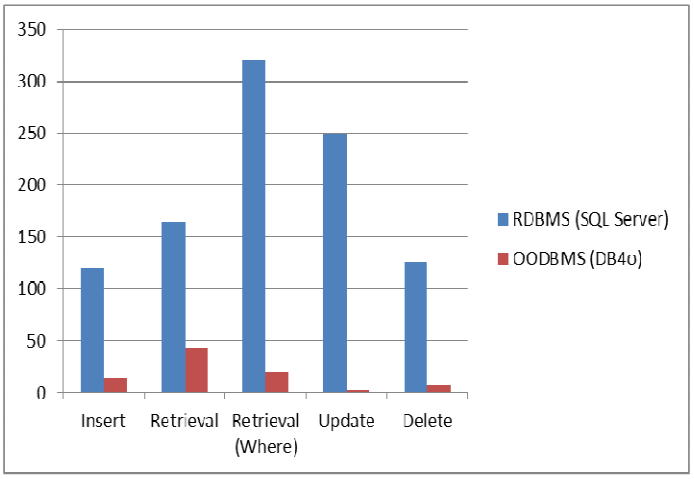

Figure 3 : Graphical Representation of the Execution time taken by various queries on both the databases

Conclusion/Future Work

As apparent from the given statistical data; performance-wise OODBMS is much better than RDBMS. On an average it takes only more than about 1/10th of the time to execute the queries as opposed to RDBMS, therefore providing a much optimized alternative to RDBMS. However, more number of queries and having higher complexity (containing complex joins) can be further used to measure the performance of both the databases.

But through this study we have found that, Object-oriented database surpasses traditional DBMS in many aspects yet it still lacks popularity in the commercial sector because of certain issues that is faced by OODBMS like it lacks maturity. There is a need for better tools to implement it, and also requires the establishment of certain standards.

Currently OODBMS and RDBMS are working supplementary to each other leading to a new concept called the ORDBMS. But in the near future OODBMS will become the new standard for the database.

References

[1] Alam, Mansaf. "Six Layers Architecture Model for Object Oriented Database." International Journal of Computer Science Issues (IJCSI, Republic of Mauritius (2011).

[2] SABĂU, Gheorghe. "Comparison of RDBMS, OODBMS and ORDBMS."Informatica Economica 11 (2007): 83-85.

[3] Luthra, Sunanda. "Architecture in Object Oriented Databases." Proc. The National Conference on Challenges and Opportunities in Information Technology. 2008.

[4] http://www.service-architecture.com/articles/object-oriented-databases/object-oriented_database_oodbms_definition.html

[5] en.wikipedia.org/wiki/Object_database

[6] Bellahsene, Zohra. "View mechanism for schema evolution in object-oriented dbms." *Advances In Databases*. Springer Berlin Heidelberg, 1996. 18-35.

[7] Dhande, Sheetal S., and G. R. Bamnote. "QUERY OPTIMIZATION IN OODBMS: IDENTIFYING SUBQUERY FOR QUERY MANAGEMENT."International Journal of Database Management Systems 6.2 (2014): 49.

[8] Saxena, Vipin, and Ajay Pratap. "A Framework for Performance Estimation of Object-Oriented Databases." methodology 4.2 (2014).

[9] Gorla, Narasimhaiah. "An object-oriented database design for improved performance." Data & Knowledge Engineering 37.2 (2001): 117-138.

[10] http://marketplace.eclipse.org/

[11] http://www.db4o.com