Research Journal of Recent Sciences ISSN 2277-2502
Vol. **1 (ISC-2011),** 419-421 **(2012)** *Res.J.Recent Sci.*

Mini Review Paper

Performance Interoperability between RDBs and OODBs

Shukla Brahma Datta and Gupta V.K.
Department of Computer Science, NIMS University, Jaipur, Rajasthan, INDIA

Available online at: www.isca.in
(Received 11th October 2011, revised 27th February 2012, accepted 30th March 2012)

Abstract

Object-oriented databases and relational database are becoming more and more popular for applications to support the complexity and the irregularity of the real-world entities. Object-Oriented Databases (OODBs) have been designed to support large and complex programming projects. The data accuracy, consistency, and integrity in OODBs are extremely important for developers and users. In Object Oriented Data Model each record is represented by object. The basic element of an object-oriented database is the object. A relational database allows the definition of data structures, storage and retrieval operations and integrity constraints. In such a model the data and relations between them are organized in tables. A table is a collection of records and each record in a table contains the same fields. Object-oriented database systems began developing in the mid-80's out of a necessity to meet the requirements of applications beyond the data processing applications which were served by relational database systems. In this paper, the Achievements and weaknesses of both database models and the Special problems found in the both model are discussed. This paper deals with different constraints in object-oriented databases and relational database.

Keyword: Distributed database system, relational model, objects oriented model, database, security, homogeneous, heterogeneous, constraint etc.

Introduction

For the last several years the most used database[1] model has been relational. While the relational model has been useful, its utility is reduced if the data does not fit into a relational table. Many organizations have data requirements that are more complex than can be handled with these data types. Multimedia data, graphics, and photographs are examples of these complex data types.

As advances in computer-related technology improve, increasingly larger files can be created, transmitted, and stored electronically. It thus becomes more apparent that object-oriented technology, and object-oriented databases in particular, are needed to warehouse the files or "objects." In contrast to the relational database model, an object-oriented database stores objects, which consist of data as well as procedures that are used to perform operations on that data. About 88 percent of organizations use relational databases[1], yet about 55 percent plan to acquire object-oriented databases at some point in the future. Main discussion of this paper is object-oriented database and compared with relational database. In this paper, the strengths and weaknesses of both database models and the Special achievements found in the object-oriented database are discussed.

Relational Database: A relational database management system (RDBMS)[1] is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd. Most popular databases currently in use are based on the relational database model. A short definition of an RDBMS is: a DBMS in which data is stored in tables and the relationships among the data are also stored in tables. The data can be accessed or reassembled in many different ways without having to change the table forms.

A relational database is a set of tables containing data fitted into predefined categories. Each table (which is sometimes called a relation) contains one or more data categories in columns. Each row contains a unique instance of data for the categories defined by the columns. For example, a typical business order entry database would include a table that described a customer with columns for name, address, phone number, and so forth. Another table would describe an order: product, customer, date, sales price, and so forth. A user of the database could obtain a view of the database that fitted the user's needs. For example, a branch office manager might like a view or report on all customers that had bought products after a certain date. A financial services manager in the same company could, from the same tables, obtain a report on accounts that needed to be paid.

Object-oriented Databases: The basic element of an object-oriented database[2] is the object. A class defines an object. In essence, classes are the blueprints for objects. In the object-oriented model, classes are arranged in a hierarchy. The root class is found at the top of the hierarchy. This is the parent class for all other classes in the model. We say that a class that is the descendent from a parent inherits the properties of

the parent class. As needed, these properties can be modified and extended in the descendent class. An object is composed of two basic elements: variables and methods.

An object holds three basic variables types: (1) Object class (2) Object ID (OID) (3) Data Stores.

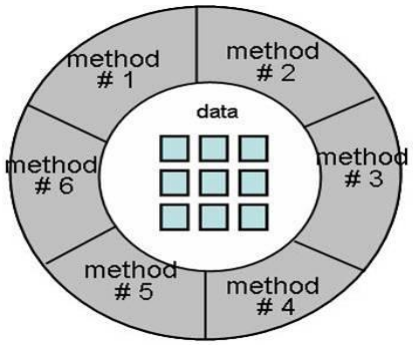

Figure-1
Object-oriented database

Methods perform two basic functions: They communicate with other objects and they perform reads and updates on the data in the object. Methods communicate with other objects by sending messages. Methods perform all reading and writing of the data in an object. For this reason, we say that the data is encapsulated in the object. This is one of the important differences between object-oriented and relational databases. All control for access, modification, and integrity start at the object level. For example, if no method exists for updating a particular object's variable, then the value of that variable is constant. Any change in this condition must be made at the object level.

When database capabilities are combined with object-oriented programming language capabilities, the result is an object-oriented database management system (OODBMS). OODBMS[2] allow object-oriented programmers to develop the product, store them as objects, and replicate or modify existing objects to make new objects within the OODBMS. Because the database is integrated with the programming language, the programmer can maintain consistency within one environment, in that both the OODBMS and the programming language will use the same model of representation. Relational DBMS projects, by way of contrast, maintain a clearer division between the database model and the application.

The class composition hierarchy is represented by IS-PART-OF relationship, and class inheritance hierarchy is represented by IS-A relationship. The Object-Oriented Data Model (OODM) can support three types of relationships between classes, which are:

Composition hierarchy (logical or physical composition) is a relationship between two classes where the instances of one class are in someway attributes, methods, and constraints of the other.

Inheritance hierarchy (single or multiple inheritance) is a relationship between superclasses and subclasses. A superclass may have any number of subclasses, which subclasses inherit attributes and methods of superclass. This means all global attributes, methods, and constraints in a superclass exist in subclasses. In addition, subclasses may have additional attributes, methods, and constraints. Class association is a relationship between classes that can be in the form of 1:1, 1:M, or M:N. Composite objects are grouping of inter-related objects that can be viewed logically as a single object.

Achievements of OODBs over RDBs: OODBs allow representation of complex objects in a more straightforward way than relational systems. In this section we will discuss some of the achievements of OODBs so far: OODBs allow users to define abstractions, facilitate the development of some relationships, eliminate the need for user defined keys, have developed a new set of equality predicates, eliminate the need for joins in some cases, have performance gains over the RDB model in some situations, and have support for versioning and long duration transactions. Finally, object algebra has been developed, although it may not be as developed as relational algebra yet.

OODBs allow users to define abstractions: OODBs have the ability to define new abstractions and to control the implementation of these abstractions.

OODBs facilitate development of some relationships: OODBs offer the feature of inverse relationships to express a mutual reference between two objects (a binary relationship).

OODBs eliminate need for user-defined keys: The OODB model has an OID that it is automatically generated by the system and that guarantees uniqueness to each object.

OODBs reduce need for Joins: OODBs have the ability to reduce the need of join.

Performance gain using OODBs: Most current OODBs are not full-fledged database systems comparable to current RDBs, OODBs have a few sources of performance gain over RDBs:

Support for versioning or long-duration transactions: Versioning and long-duration transactions are missing in RDBs. Few OODBs offer versioning and long-duration transactions, though with limited facilities only. Development of Object Algebra: Though not as developed and mature as relational algebra, object algebra has been developed that defines five fundamental object-preserving operators: *union*, *difference*, *select*, *generate* and *map*.

Weaknesses of OODBs: The expectation was that object-oriented technology would bring a quantum jump to database technology. But, in spite of the achievements of OODBs discussed above, OODBs have not been able to make a major impact because of weaknesses still present in OODB model and technology.

In OODBs there is a lack of basic features that users of database systems have become accustomed to, and therefore expect. The features include, lack of interoperability between RDBs and OODBs, minimal query optimization, lack of standard query algebra, lack of query facilities, no support for views, security concerns, no support for dynamic class definition changes, limited support for consistency constraints, limited performance tuning capabilities, little support for complex objects, limited integration with existing object-oriented programming systems, limited performance gains, among others.

Need for OODBs: Relational data model has several restrictions. Object-oriented data model, based on the object oriented paradigm for programming and data management gained popularity in last few decades because of the advantages it provides in supporting complex object and multi-valued attributes.

Object-oriented databases support inheritance, object-identity, encapsulation, rich type system (including structured and collection types) and have also been studied in detail. Two chief trends emerged:

Extending object-oriented languages to support database operations: Object-relational model: extends relational model and provides rich type system of object-oriented databases, combined with relations as basis of for storage of data.

Interoperability between RDBs and OODBs: For OODBs[2] to make a major impact on the database market, following has to be done:

OODBs have to be made full-fledged database systems, sufficiently compatible with RDBs – a migration path is needed to allow the coexistence and the gradual migration from the current products to new products; Application development tools and database access tools have to be developed for such database systems; Architectures of the RDBs and OODBs have to be unified; The data models of the RDBs and OODBs have to be unified.

Conclusion

We have discussed object oriented database and relational database. We also discussed various constraints of both database models. We have seen that Object-Oriented Databases[3] have been designed to support large and complex programming projects. Object Oriented Databases generally provide persistent storage for objects. In addition, they may provide one or more of the following: a query language; indexing; transaction support with rollback and commit; the possibility of distributing objects transparently over many servers. These features are described. Main part of this paper is comparison of object-oriented database with relational database. An object-oriented database is an entirely different beast from a relational database.

For future work, I plan to examine security level in object oriented Database System. We are in the process of investigating schemes by which the performance of high security level transactions can be improved without compromising with the security. Further we are looking to secure real time object oriented distributed systems by which the performance of high security level transactions can be improved without compromising the security.

References

1. Codd E. F., A Relational Model of Data for Large Shared Data Banks, Communications of the ACM, **(1970)**

2. Bertino, Elisa, Data Hiding and Security in Object-Oriented Databases, In proceedings EighthInternational Conference on Data Engineering, 338-347, **(1992)**

3. Rothnie J.B. and Goodman N., A Survey of Research and Development in distributed database Management Systems, Proc. Third Int. Conf. on VLDB, **(1977)**

4. Bernstein P. and Goodman N., Concurrency Control in Distributed Database Systems, ACM Computing Surveys 13/2, **(1981)**

5. Tamer M. , Ozsu and Patrick Valduriez. Principles of Distributed Database Systems, Second Edition. Prentice-Hall, **(1999)**

6. Boot G.M.,The distributed System Environment, McGraw-Hill, **(1981)**

7. Jeffrey D. Ullman, Principles of Database and Knowledge-based Systems, 2nd edition, Computer Science Press, **Vol.1**, **(1982)**

8. Bell, David and Jane Grisom, Distributed Database Systems, Workinham, England: Addison Wesley, **(1992)**

9. Woo, Thomas Y.C., and Simon S. Lam, Authorization in Distributed Systems: A Formal Approach, In Proceedings 1992 IEEE Symposium on Research in Security and Privacy, 33-51**(1992)**

10. Elmasri Navathe, Database Concepts By Pearson Education **(2011)**

11. Colloly., Data base Concepts By Pearson Education **(2011)**

12. Coronel Rob, Introduction to Database Concepts **(2011)**