Comparison Analysis of Object-Based Databases, Object-Oriented Databases, and Object Relational Databases

S. O. Ogunlere*, S. A. Idowu

Babcock University,
Computer Science Department
Ilishan-Remo, Ogun State, Nigeria
Tel: +2348067622845

Corresponding author's email: OgunlereS [AT] babcock.edu.ng

ABSTRACT--- ***The concepts of Object-Based Databases (OBD), Object-Oriented databases (OODB) and Object Relational Databases (ORDB) are of paramount importance in today's technological development. A comparison analysis of the concepts and usage of these variants of databases is presented in this paper with respect to areas of applications and features for more clarification. In addition, this paper also presents different data Models for these variants and discusses their relative strength and weaknesses which may serve as good resources for programmers***.

Keywords--- Object-Based Databases, Object-Oriented Databases, Object Relational Databases, Inheritance, Polymorphism and Dynamic binding.

1. INTRODUCTION AND BACKGROUND

Databases are being used more than ever before to store and to access information in data intensive applications mostly over World Wide Web. Due to the ease of maintenance and outstanding performance of databases, the growth of database technologies has of recent times increased rapidly. Many web-applications are database – driven, while records are retrieved and displayed in a formatted form using web languages like *eXtensible eMarkup Language* (XML). A Database Management System (DBMS) is designed around a particular data model that allows all system components (and humans) to understand the schema and data. A *data model* describes the possible schemas (essentially the *meta-schema).* Data Models can be categorized according to the data structures and operators they present to the user, (Bloor, 2003). Possible data models are:

i. E-R Model
ii. Hierarchical Data Model
iii. Network Data Model
iv. Semi Structured Data Model
v. Relational Data Model (RDBMS)
vi. Object-Based Data model (OBDMS)
vii. Object Oriented Data Model (OODBMS)
viii. Object Relational Data Model (ORDBMS)
ix. Structured, Unstructured and Semi-structured data (XML)

Among all these data models mentioned above, **relational model** followed by **object-based, object oriented** and **object relational models** enjoyed most popularity. **XML** is known as the de-facto standard for exchanging data between different systems, platforms, applications, and organizations (just as interface).

1.1 Brief Review of Object Oriented Databases

As a result of research in 1970s and 1980s into having database management support for graphic-structured objects, object database management systems was born. Notable research projects included:

- Encore-Ob/Server (Brown University)
- EXODUS (University of Wisconsin)

- IRIS (Hewlett-Packard)
- ODE (Bell Lab)
- ORION (MCC) Microelectronics and Computer Technology Corporation
- Vodak (GMD-IPSI),
- Zeitgeist (Texas Instruments)

Object-Based Database management systems add the concept of persistence to object programming languages (as could be seen in the early commercial products where various programming languages were integrated). They store data in a logical model that is closely aligned with an application program's object model. Though they will have a physical data model optimized for the kinds of logical data model it expects.

2. OBJECT-BASED DATABASE MANAGEMENT SYSTEMS (OBDMS) CONCEPT

In the Context of Software development, Object based Programming concept is a concept of programming by relating software to a real world object. In this case the objects become the key building blocks of the programs but the major factor in this model is that objects already exist and the programming must be based on the existing objects. The data store is constructed from structure of object, usually from existing data types. Programming or functionality is being built on already existing object. Hence, object based language browser enables users to manipulate and access the predefined objects, but cannot create owns custom objects.

Three major examples of Object Based Languages are Visual Basic (Version 6 and below), VB Script, JavaScript. For instance, in Visual Basic 6 which utilizes a Graphical User Interface (GUI), Programs or applications are built by placing Several Controls (Like Buttons, Textbox, LabelBox, Listbox, Combobox) on the form, and those controls are coded to interact and bring out the functionality of such application. All of these controls (Objects) exists by defaults and are not created by programmer. Instance of any of the control is however created whenever a copy is placed on the form. Those controls cannot be created but can be modified during software development. This is a good illustration of object based system. The programming is based or built on those objects. These languages all support the definition of an object as a data structure but lack polymorphism and inheritance

According to Wegner & Peter (1987) [14], Object-based languages need not support **inheritance** or **subtyping**, but those that do are also said to be **"object-oriented".** Object-based languages that do not support inheritance or subtyping are usually not considered to be true object-oriented languages; but by definition, all object-oriented languages are considered object-based hence the terms "object-based" and "object-oriented" are normally used as mutually exclusive alternatives, rather than as categories that overlap.

The term "object-based" sometimes may be applied to **prototype-based languages,** true object-oriented languages that do not have classes, but in which objects instead inherit their code and data directly from other "template" objects. An example of a commonly used prototype-based ***scripting language*** is ***JavaScript.*** Both object-based and object-oriented languages whether class-based or prototype-based, may be statically type-checked. Statically checking prototype-based languages can be difficult because these languages often allow objects to be dynamically extended with new behavior, and even to have their parent object (from which they inherit) changed, at run time.

Thus the term object-based database may be thought of having two different meanings described as follows:

(a) As a limited version of object-oriented programming, where one or more of the following restrictions applies:

- No implicit inheritance
- No polymorphism
- Only a much reduced subset of the available values are objects (typically the GUI components).

(b) As a prototype-based system (that is, those based on "prototype" objects that are not instances of any class).

Hence the main difference between object based and object oriented language is that the language which supports the concepts of ***inheritance*** and ***dynamic binding*** is known as object oriented while the language which does not support these two concepts is called object based language. But in real life scenario and in practice, both are often referred to as object-oriented databases using different language formats.

3. OBJECT ORIENTED DATABASE MANAGEMENT SYSTEMS (OODBMS) CONCEPT

A major trend in computer science is the object-oriented paradigm, an approach to program design and implementation using collections of interacting entities called objects. Objects incorporate both data and operations. In this way they mimic things in the real world, which have properties (data) and behaviors (operations). Objects that hold the same kind of data and perform the same operations form a class.

Object oriented databases store objects rather than data such as integers, strings or real numbers. Objects are used in object oriented languages such as Smalltalk, C++, Java, and others. Objects basically consist of the following:

- Attributes - Attributes are data which defines the characteristics of an object. This data may be simple such as integers, strings, and real numbers or it may be a reference to a complex object.
- Methods - Methods define the behavior of an object and are what was formally called procedures or functions.

Therefore objects contain both executable code and data. There are other characteristics of objects such as whether methods or data can be accessed from outside the object. Other term worth mentioning is classes rather than entity as the case may be for RDBMS. Classes are used in object oriented programming to define the data and methods the object will contain. The class is like a template to the object. The class does not itself contain data or methods but defines the data and methods contained in the object. The class is used to create (instantiate) the object. Classes may be used in object databases to recreate parts of the object that may not actually be stored in the database. Methods may not be stored in the database and may be recreated by using a class.

In software development cycle, Object oriented programming is also a software development concepts that relate programming or software to real life objects. This concept is however encompassing in the conceptualization of object. Object can be created, can be modified and can even be terminated. Examples of object oriented Languages are Java, C++, Objective C, PHP, C#, etc. Each of these languages have their specific ways of implementing object but they all support the three main features of object orientation. Object-Oriented Database Management System (OODBMS) support two concepts known as dynamic binding or subtyping and inheritance.

It is clearly obvious to ascertain that OODBMSs combine database capabilities with object-oriented programming language capabilities which 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. The programmer can maintain consistency within one environment because the database is integrated with the programming language in that both the OODBMS and the programming language will use the same model of representation.

It is therefore evidence that object-oriented database technology is a marriage of object-oriented programming and database technologies as illustrated in figure 1. This figure shows how these programming and database concepts have come together to provide what it is now refers to as object-oriented databases.

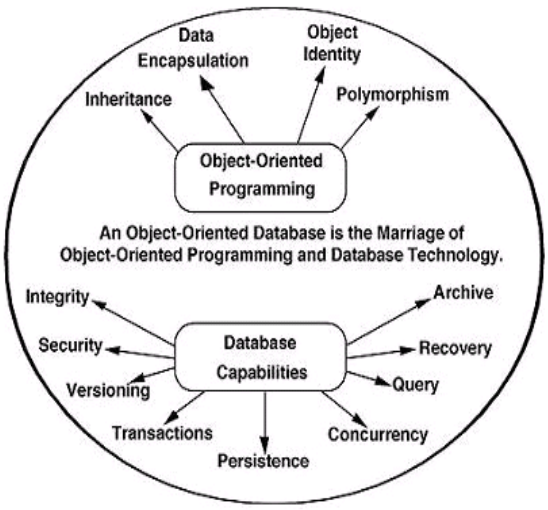

Figure 1: Makeup of an Object-Oriented database (Chaterjee, 2005)

The most significant characteristic of object-oriented database technology is that it combines object-oriented programming with database technology to provide an integrated application development system. It is important to know that the goal of object-oriented databases is to maintain a direct correspondence between real-world and database objects. The strengths and weakness of OODBMS are depicted in Table 1 below.

Table 1: Advantages and Disadvantages of OODBMS (Mam, 2014)

MODEL	ADVANTAGES	DISADVANTAGES
OODBMS	More semantic information Support for complex objects Extensibility of data types May improve performance with efficient caching Versioning and Reusability Inheritance speeds development and application Potential to integrate DBMSs into single environment.	Strong opposition from the established RDBMSs Lack of theoretical foundation Throwback to old pointer systems Lack of standard ad hoc query language Lack of business data design and management tools Steep learning curve and Low market presence Lack of compatibility between different OODBMSs.

3.1 OBJECT –ORIENTED DATABASE ADVANTAGES

- Objects don't require assembly and disassembly saving coding time and execution time to assemble or disassemble objects.
- Reduced paging and easier navigation
- Better concurrency control - A hierarchy of objects may be locked.
- Data model is based on the real world.
- Works well for distributed architectures.
- Less code required when applications are object oriented.

3.2 OBJECT –ORIENTED DATABASE DISADVANTAGES

- Lower efficiency when data is simple and relationships are simple.
- Relational tables are simpler.
- Late binding may slow access speed.
- More user tools exist for RDBMS.
- Standards for RDBMS are more stable.
- Support for RDBMS is more certain and change is less likely to be required.

4. OBJECT RELATIONAL DATABASE MANAGEMENT SYSTEMS (ORDBMS) CONCEPTS

O**bject-Relational Database Management System** (**ORDBMS**) is a database management system (DBMS) similar to a relational database, but with an object-oriented database model. Objects, classes and inheritance are directly supported in database schemas and in the query language. Object Relational Database Systems implements both the concept of Relational Database System and Object Oriented Database System in a single system. In addition, just as with pure relational systems, it supports extension of the data model with custom data-types and methods.

The basic goal for the Object-relational database is to bridge the gap between relational databases and the object-oriented modeling techniques used in programming languages such as Java, C++. However, a more popular alternative for achieving such a bridge is to use a standard relational database system with some form of Object-relational mapping (ORM) software. Whereas traditional RDBMS or SQL-DBMS products focused on the efficient management of data drawn from a limited set of data-types (defined by the relevant language standards), an object-relational DBMS allows software developers to integrate their own types and the methods that apply to them into the DBMS.

The ORDBMS (like ODBMS or OODBMS) is integrated with an object-oriented programming language. The characteristic properties of ORDBMS are:

i. Complex data
ii. Type inheritance and

iii. Object behavior.

Complex data creation in most SQL ORDBMSs is based on preliminary schema definition via the user-defined type (UDT). Hierarchy within structured complex data offers an additional property.

Type inheritance is when a structured type can have subtypes that reuse all of its attributes and contain additional attributes specific to the subtype.

Object behavior is related with access to the program objects. Such program objects must be storable and transportable for database processing. They are usually named as 'persistent objects.

Inside a database, all the relations with a persistent program object are relations with its object identifier (OID). All of these points can be addressed in a proper relational system, although the SQL standard and its implementations impose arbitrary restrictions and additional complexity.

In object-oriented programming (OOP), object behavior is described through the methods (object functions). The methods denoted by one name are distinguished by the type of their parameters and objects. The OOP language feature known as the polymorphism principle is defined as "one interface, many implementations". Other OOP principles, inheritance and encapsulation, are related both to methods and attributes. Method inheritance is included in type inheritance. Encapsulation in OOP is a visibility degree declared, for example, through the PUBLIC, PRIVATE and PROTECTED modifiers. Examples of Object relational Database are Postgres SQL, Illustra, Oracle and Microsoft SQL.

Of all the listed examples above, Postgres SQL is the most distinguished and relevant object relational database system in the context of the definition. Microsoft and Oracle SQLs are primarily Relational but with extension in capabilities to support object orientation in data management. Considering the merits of object database over relational database and vice versa, Object relational database leverage the two and removes all the demerits. In simple words, we can say that ORDBMSs synthesize the features of RDBMSs with the best ideas of OODBMSs. Table 2 below shows the advantages and disadvantages of object relational databases.

Table 2 Advantages and Disadvantages of ORDBMS (Mam, 2014)

MODEL	ADVANTAGES	DISADVANTAGES
ORDBMS	Ability to query complex applications and ability to handle large and complex applications Reduced Network Traffic…queries and complex instructions can be executed on the server (as opposed to clients) Application and Query Performance…...Parallel server technology employed for Software Maintenance…data and methods are stored on the server and makes maintenance easier Integrated Data and Transaction Management….The database engine handles all transaction integrity, backup, etc.	Modeling and processing support of complex objects and their versions, large objects, semantic-rich relationships, etc. is only rudimentary or even missing in current ORDBMSs ORDBMSs have to be complemented by adequate client-side data management and long-running design transactions encapsulating the client processing model, in order to provide satisfactory support for technical applications Low performance in web applications.

5. CONCLUSION

In view of all the discussions above, Understanding database and object concepts is key to the usage or deployment of any database system. Based on this, in database paradigm, the term object based and object oriented are logically the same since in database, objects, emphasis are not on all the three features of object orientation of Inheritance, Encapsulation and Polymorphism but on any of those characteristics of objects. Therefore, there are two main categories of object database systems.

1. Object Database (Object Based Database, Object Oriented Database or simply Object Database) Systems

2. Object Relational Database Systems which implements both the concept of Relational Database System and Object Oriented Database System in a single system.

6. REFERENCES

[1] Jan E. K., (2005), "XML Databases – do they really exist?", ELAG 2005 at CERN, Geneva, BIBSYS Lib Automation.
[2] Barbey, S; M. Kempe, and A. Strohmeier, (1993), "Object-Oriented Programming with Ada 9X". Draft Technical Report (Swiss Federal Institute of Technology in Lausanne Software Engineering Laboratory). Retrieved 15, December, 2013.
[3] Barry, Douglas K. "The Truth about Object Databases," Database Programming & Design, July 1997, pp. 44- 48.
[4] Communications of the ACM, Special Section: Next Generation Database Systems, October 1991.
[5] Davis, Judy. "Extended Relational DBMSs: The Technology, Part 1," DBMS Online, June 1997.
[6] Frank, Maurice. "Debating Databases at DB/Expo," DBMS Online, December 1996
[7] Hellerstein, Joseph M., Chapter 21: "Object-Database Systems," in Ramakrishnan, Raghu: Database Management Systems (McGraw Hill, 1997).
[8] Introduction to RDBMS, OODBMS and ORDBMS
[9] Jakobovits, Rex. "Comparison to Relational Databases".
[10] King, Nelson H. "Object DBMSs: Now or Never," DBMS Online, July 1997.
[11] Linthicum, David S. "Selecting a DBMS," DBMS Online, July 1996
[12] Matthias N. & Bert V. D.L., (2005)," Native XML Support in DB2 Universal Database", IBM Silicon Valley Lab,555 Bailey Avenue, San Jose, CA San
[13] Stonebraker, Michael. "Object-Relational DBMS, The Next Wave," Informix website.
[14] Wegner, Peter (December 1987), "Dimensions of Object-Based Language Design", In Meyrowitz, Norman. OOPSLA'87 Conference Proceedings 22 (12): 168—182.