Global Journal of Computer Science and Technology: C
Software & Data Engineering
Volume 15 Issue 3 Version 1.0 Year 2015
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Object Oriented Database Management Systems-Concepts, Advantages, Limitations and Comparative Study with Relational Database Management Systems

By Hardeep Singh Damesha

Lovely Professional University, India

Abstract- Object Oriented Databases stores data in the form of objects. An Object is something uniquely identifiable which models a real world entity and has got state and behaviour. In Object Oriented based Databases capabilities of Object based paradigm for Programming and databases are combined due remove the limitations of Relational databases and on the demand of some advanced applications. In this paper, need of Object database, approaches for Object database implementation, requirements for database to an Object database, Perspectives of Object database, architecture approaches for Object databases, the achievements and weakness of Object Databases and comparison with relational database are discussed.

Keywords: relational databases, object based databases, object and object data model.

GJCST-C Classification : F.3.3

Strictly as per the compliance and regulations of:

Object Oriented Database Management Systems-Concepts, Advantages, Limitations and Comparative Study with Relational Database Management Systems

Hardeep Singh Damesha

Abstract- Object Oriented Databases stores data in the form of objects. An Object is something uniquely identifiable which models a real world entity and has got state and behaviour. In Object Oriented based Databases capabilities of Object based paradigm for Programming and databases are combined due remove the limitations of Relational databases and on the demand of some advanced applications. In this paper, need of Object database, approaches for Object database implementation, requirements for database to an Object database, Perspectives of Object database, architecture approaches for Object databases, the achievements and weakness of Object Databases and comparison with relational database are discussed.

Keywords: relational databases, object based databases, object and object data model.

I. Introduction

History of data processing goes through many different changes with different technologies along with the time. In decade there is huge increase in the volume of data that need to be processed due to which sometimes old technology do not work and need to come with new technology to process the data. History of database technology has used Unit Records & Punch Card, Punch Card Proliferation, Paper Data Reels, & Data Drums, File Systems, Database Systems, NoSQL and NewSQL databases. From last five decades, the mostly used technology is database management systems.

After some limitations of file systems, researchers come up with new technology known as Database Management Systems which is the collection of software or programs to maintain the data records. Initially, two models are proposed are hierarchical and network models, but these models don't get much popularity due to their complex nature. Then a researcher E.F. Codd comes up with a new data model known as relational model in which data items are stored in a table. Many DBMS's are developed on the basis of this model. This is the most popular model till now because it has conceptually foundation from relational mathematics.

Author: Lovely Professional University, Punjab.
e-mail: hardeep684@gmail.com

In mid-1980's ,no doubt RDBMS are very much popular but due to some limitation of relation model and RDBMS do not support for some advanced applications[1] OODB comes in the picture. At that time Object Oriented Programming paradigm is very much popular. Due to this researcher think to combine the capabilities of database and object based paradigm for programming. In Object databases data is stored in the forms of objects. These database management systems are not very much popular because due to the lack of standards.

Research is going on the database technology from 1960's up to this day. Many improvements are done in database technology by researcher in last decade and more technologies are coming to improve the database technology. The new database technologies include new transaction management and concurrency control methods and Redundant Array of Independent Disks (RAID) for efficient storage and Big Data and Cloud Computing.

II. Why Object Oriented Databases?

There are three reasons for need of OODBMS:

A. Limitation of RDBMS
B. Need for Advanced Applications
C. Popularity of Object Oriented Paradigm

A. Limitation of RDBMS

These limitations are in relational model. Due to this these limitations are reflected to all RDBMS [2]. These limitations are:

1. *Poor representation of real world entities:* The Relational model cannot represent real world in proper way because it has only one semantic that is table which can represent the real world entity in proper way.
2. *Normalization is necessary, but sometimes not useful:* Normalization in RDBMS to maintain the consistency of the database, but some broken relations is not related to real world.
3. *Overloading of semantic structure:* Relational Data Model has only one semantic structure for representing data and relationship that is table. Due

to this, sometimes it becomes very difficult to find out that which is going to model data or relationship?

4. *Poor support for integrity and enterprise constraints:* Constraints are very much needed for your database have to be desired data. RDM supports only limited number of constraints. The enterprise constraints are those which are defined by industry standards.
5. *Homogeneous data structure:* RDM requires homogeneous data structures like:
   - RDM assumes both horizontal and vertical homogeneity.
   - Relational mathematics algebra has only fixed number of operations due to which Relational Model operations cannot be extended.
6. *Tables can store only atomic/single value:* No doubt, this is property of RDM. But sometimes in many situations this property becomes its limitation.
7. *Normalization is strongly recommended:* Most of the situations, you have must normalize the relation make the data consistency inside your database.
8. *Difficulty in handling recursive queries:* There is very poor support to handle recursive queries in RDBMS. For this you must know:
   - Depth of recursive query must be known.
   - You can use the transitive closure operations to handle recursive queries in RDBMS.
9. *Impedance mismatch:* SQL Data Manipulation Language (DML) is lack computational completeness [10]. To overcome this situation, you must embed the SQL with any high programming language like C++, Java, and C #. Due to there will be impedance mismatch between two language SQL and higher programming language.
10. *Poor support for long duration transactions:* In RDBMS, generally transactions are short lived and concurrency control techniques or mechanisms are not good for .long duration transactions.
11. *Poor Schema Evolution support:* Schema Evolution means making changes to schema of database at runtime without interrupt the execution of the application.
12. *Poor Navigational Access:* There is very poor support for the navigational access in RDBMS.

There are some advanced applications need the database with deeper structural and functional foundation of capabilities that are not provided by conventional database [1]. These applications are:

B. Need for Advanced Applications

a) Computer Aided Design (CAD):

- In these types of applications, relevant data about buildings, airplanes and integrated circuit chips is stored and managed. In this type of applications, database design may be very large.
- Design in these types of applications is not static. This design is evolves through the times. Updates need to be propagated.
- These applications require version control and configuration management.
- These applications require complex objects for their development. For example, a car's component may be related to other components.
- Need long duration transactions because sometimes updates are for reaching.
- Support for cooperative engineering because most of the times many people work on same design.

b) Computer Aided manufacturing (CAM):

- These application data is very much similar to CAD, but needs discrete production.
- These applications must respond to real time events.
- Generally algorithms and custom rules are used to respond to a particular situation.

c) Computer Aided Software Engineering (CASE):

- These applications manage data about the phases of software development life cycle.
- Design may be extremely large.
- Involves cooperative work.
- Need to maintain dependencies among components.
- Versioning and configuration management.

d) Network Management Systems:

- Coordinates communication services across the network.
- These systems are used for such tasks as network path management, problem management and network planning.

e) *Other Applications:* The Object Oriented Database also used in Office Information Systems, Multimedia systems, Digital Publishing and Geographic information Systems.

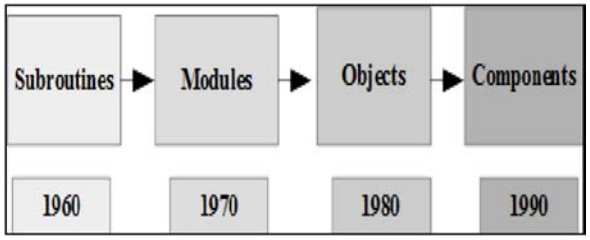
C. Popularity of Object Oriented Paradigm

Another domain that enforces the development of OODBMS is popularity of object oriented programming paradigm [4], [5], [6], [7], [8] because a real life situation can be model in best way by using object oriented programming.

OO Programming Aspects:

1. *Abstraction:* It is process of finding important aspects of an entity and ignoring unimportant aspects such as implementation details. The properties comprise two things state and behaviour. A state is models through the attributes of object and behaviour is models through operations executed on data by object.

2. *Object:* An object is something uniquely identifiable, models a real world entity and has got state and behaviour. The only big difference between entity and object is that entity has only state has no behaviour, but object contains both state and behaviour. Example: Student
3. *Encapsulation:* An object contains both current state (Attributes) and set of methods used to manipulate it. It is called encapsulation.
   Information Hiding: It is process of separates external properties of an object from its internal properties, which are hidden from external environment. These two concepts also related with abstraction.
   Importance: These two concepts support the facility that internal properties of an object to be changed without affecting applications that use it, provided external properties remain same. It also provides data independence.
4. *Attributes:* These represent the current state of an object. They can be two types:
   - Simple Attribute
   - Complex Attribute
5. *Object Identity (OID):* It is the unique identifier associated with every object in the system. It has following characteristics:
   - It is generated by system.
   - It is unique to that object in the entire system.
   - It is used only by the system, not by the user.
   - It is independent the state of the object.
6. *Methods and Messages:* These implement the behaviour of an object and involve encapsulation.
   Message: It is a request or call to an object to execute the method that is defined by message.
7. *Class:* It is Container/Template/Blue-print for objects. Objects inside a class called instances. It comprises many definitions in many different situations. For example: A Class behaves like an object with its own class data and operations.
8. *Inheritance:* It is the special type of relationship between classes: The inheriting class inherits the some or all properties of the base class depend which mode of inheritance is used. Special classes or inheriting classes are called subclasses and general classes are called super classes.
   Generalization: It is method to create a superclass is called generalization.
   Specialization: It is process of forming a sub class is called specialization.
9. *Polymorphism:* It means "many forms". It is dynamic feature which executes at run time of program. It involves the concept of overriding and overloading [10].
10. *Complex Objects:* An object is called complex object if it contains many sub objects and it is viewed as single object.
    Example: Country. Because it contains many states and again states contains cities.
11. *Relationships:* It is basically an association between two things. These are represented through reference attributes, typically implemented through OID's. Types of binary relationships are:
    - One to One relationship
    - One to Many relationship
    - Many to One relationship
    - Many to Many relationship
    - The different programming paradigm during the different decades:



    - Due to this researchers think to combine OO Paradigm and DB.

III. Approaches for OODBMS

A. Relational Extension Based DBMS
B. Object/Relational DBMS
C. Pure OODBMS

A. *Relational Extension Based DBMS*

This is the first approach that is adopted by industry and academia towards the implementations of OODBMS is to extend the relational model to provide the OO features [1], [2]. The advantages of this approach are:

- Stick to relational model
- Have to OO features like complex object and UDT (User Defined Types).

Design techniques for relational extensions: In mid-80's a researcher named Stonebraker in OODBMS field represent the design techniques in this field with different proposals for Extended Type System for an OODBMS should follow:

- Definition of new data types
- Definition of new operations of so defined data types.
- Implementation of access methods.
- Optimized query processing for the queries on new data types.

Other Extensions in RDBMS: The different techniques are adopted by different DBMS to support to support OO features:

- Support for variable length "undefined" data values. Using this support, generalized user defined data types can be represented. Like Oracle supported RAW, LONG and LONGRAW (65535 bytes).

- Sybase support TEXT and IMAGE up to 2GB and also others. These features were partial support for storing complex data. Such facilities were mainly used to capture non-text data like voice, medical charts and fingerprints.
- User defined procedure are associated with used defined data types.
- Example: POSTGRES

Postgres: It is developed at UC Berkeley in mid-80 by Prof. Stonebroker and his group. It is commercialized as ILLUSTRA. In this INGRES which is basically a relational database management system to support OO features. Basic idea in POSTGRES was to introduce minimum changes in the Codd's original relational model to achieve the objective. Advantage is the continuity with the previous product (INGRES) and provision of OO features in the new product. Design objectives of POSTGRES declared by Stonebraker were:

- To provide fully implemented functionality of complex objects.
- Support for User/Abstract Defined Types, operators and functions for accessing.
- To provide functionality of Active Databases and Inferencing.
- QUEL is the manipulation language in INGRES.
- POSTQUEL in POSTGRES :Most of QUEL commands are included in POSTQUEL:
  1. Time varying data (snapshots and historical data)
  2. Iterative queries
  3. Alerters, Triggers and Rules for Inferencing.

B. *Object/Relational DBMS*

These systems have relational and object based both features by the definition [1], [2]. They provide similar objectives as provided by the Relational Extension approach of RDBMS. In this approach, build an object layer on the top of relational system like Open ODB and ODAPTER. They are built on different architectures like Query Server or Client/Server.

Open ODB/ODAPTER: Open ODB is an ORDBMS from HP during mid's 90 and aims to support for broad base applications. It has following features:

- Based on Iris DBMS
- Based on Client/Server architectures
- Both data and applications can be shared by the user applications.
- Clients use Application Program Interface (API) to access information.
- OSQL is data manipulation language for Open ODB/ODAPTER.
- Open ODB uses relational techniques to support to OO features.
- Object Model is implemented by Object Manager.
- Mapping to OO schema and queries to relational ones.
- The underlying relational storage manager is ALLBASE/SQL.

C. *Pure OODBMS*

These type OODB's systems are not much popular because lack of standards [9]. There is no single definition for a single concept. For Example: An Object has many definitions, but in RDB there is a fixed standard for or single definition for each concept like table .Here defining some definitions which are mostly accepted but not standardize [2].

OODB Model: It is data model that capture semantics of objects suited in object based programming paradigm.

ZDONIK and MAIER give a threshold model that an Object database must have following features:

- Database functionality like transaction management, concurrency control etc.
- Facility of Object Identity (OID).
- Facility of encapsulation.
- Facility of complex objects.
- Inheritance not must but may useful.

OODB: It is permanently stored and sharable collection of objects suited with an Object Data Model.

OODBMS: It is system which contains application programs which are used to manage all object oriented database activities like manipulation of objects.

Some Commercial OODBMS [9]:

- Gemstone OODBMS is developed by Gemstone Systems Incorporation.
- Objectivity/DB OODBMS is developed by Objectivity Incorporation.
- Objectstore OODBMS is developed by Progress Software Corporation.
- Ontos OODBMS is developed by Ontos Incorporation.
- DB4O from Versant Corporation.

IV. Oodbms Manifesto

a) *Mandatory Features*

The OODBMS paradigm manifesto set the minimum fundamental directional basis for an OODBMS model [3], [4], [5, [8].These characteristics can be classified as mandatory and optional features:

1. *Support for complex objects:* A OODBMS must support for complex objects. Complex objects can be obtained by applying constructor on basic objects.
2. *Object Identity:* It is the unique identifier associated with every object in the system. It has following characteristics:
   - It is generated by system.

- It is unique to that object in the entire system.
- It is used only by the system, not by the user.
- It is independent the state of the object.

3. *Encapsulation:* An OODBMS should enforce encapsulation through access objects only.
4. *Types or Classes:* A OODBMS must support for one of them types or classes.
5. *Inheritance and Hierarchies:* A OODBMS must support for concept of super classes and subclasses. The types of heritance can be:
   - Substitution
   - Inclusion
   - Constraint
   - Specialization
6. *Dynamic Binding:* An OODBMS must support concept of dynamic binding in programming language such as:
   - Overloading
   - Overriding
   - Late binding
7. *Computationally Complete DML:* To provide a support for data processing database have use computationally completely language like SQL-3.
8. *Extensible set of data types:* A OODBMS must support for used defined data types.
9. *Data Persistence:* This is basic requirement for any DBMS.A OODBMS must provide persistent by storing object in proper way.
10. *Managing very large databases:* A OODBMS must support for large databases.
11. *Concurrent Users:* This is basic requirement for any DBMS. It must support for concurrency control.
12. *Transaction Management:* This is also basic requirement of any DBMS.
13. *Query Language:* This is also a basic requirement of any DBMS. This query language must be computationally complete.

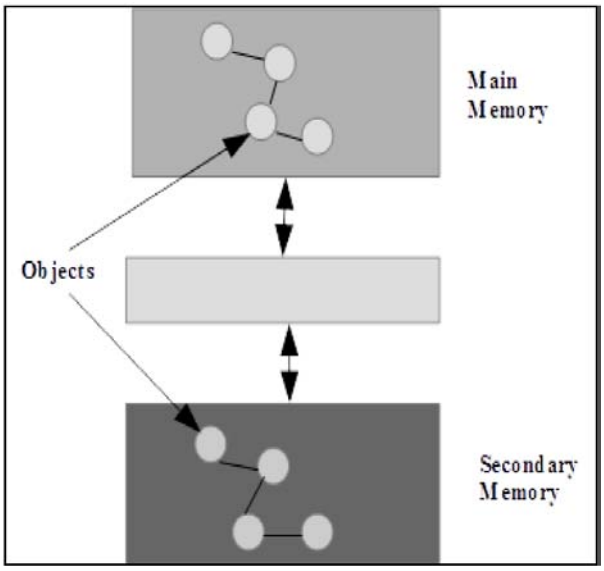
b) Optional Features

1. *Multiple Inheritance:* Multiple inheritance is not directly support by multiple objects oriented programming languages. An OODBMS can also support for multiple inheritance.
2. *Type checking and inferencing:* Type Checking and Inferencing features can be added to Object Databases.
3. *Long duration and Nested Transactions:* Relational database transactions are short-lived. An OODBMS can support for .long duration transactions and also for nested transactions.
4. *Distributed databases:* An object database may have support for distributed database which is a collection of multiple databases logically related and distributed over the network.
5. *Versions:* An OODBMS can support for version control and configuration management.

V. OODBMS Perspectives

Memory Management in OODBMS: There are different memory management techniques for different OODB systems [2]. For Extended Relational Model database Systems and Object/Relational (by definition) Database Systems (ORDBMS) Two Level Storage Model is used and for pure OODBMS is Single Level Storage Model is used.

a) Single Level Storage Model



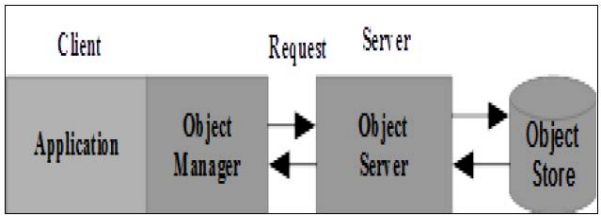
VI. OODBMS Architectures Approaches

The basic theme of OODBMS is to add persistence to OOPL as they provide object orientation. The major difference is that here database needs to store data as well as methods [1].

Client/Server Architecture: There three basic client/server architecture approaches [2]:

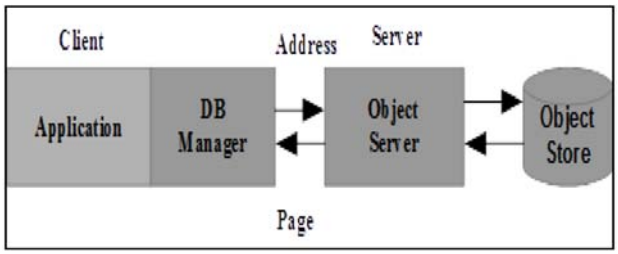
a) Object Server
b) Page Server
c) Database Server

a) *Object Server:* There is a distributed processing environment between the two parts client and server. Typically, server is responsible for other OODBMs functions. Transaction control management and interface to programming language is handled by Client.

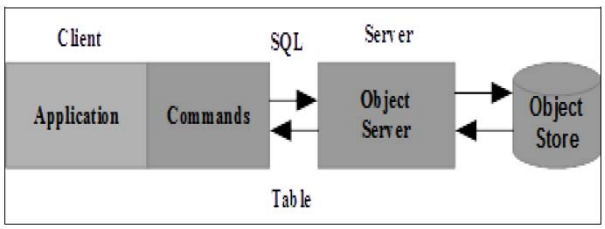


b) *Page Server:* In this client-server model, client is responsible for database processing most of the times. Secondary storage management and

response to requests is handled by server. A page can contain many complex objects or normal objects.



c) *Database Server:* In this approach, Client simply passes the request to the server, receives results and passes them to application. Most of database processing done at server .This approach is used mainly by RDBMS's.



VII. Achievements and Weaknesses of OODBMS

a) Achievements

1. *Support for User Defined data types:* OODBs provides the facilities of defining new user defined data types and to maintain them [9].
2. *OODB's allow creating new type of relationships:* OODBs allow creating a new type of relationship between objects is called inverse relationship (a binary relationship) [11].
3. *No need of keys for identification:* Unlike, relational model, object data model uses object identity (OID) to identify object in the system [8].
4. *Development of Equality predicates:* In OODBs, four types equality predicates [8], [10] are:
   - Identity equality
   - Value equality of objects
   - Value equality of properties
   - Identity equality of properties
5. *No need of joins for OODBMS's:* OODBs has ability to reduce the need of joins [9].
6. *Performance gains over RDBMS:* Performance gains changes application to application. Applications that make the use of object identity concept having performance gains over RDBMS's [9].
7. *Provide Facility for versioning management:* The control mechanisms are missing in most of the RDBMS's, but it is supported by the OODBMS's [9].
8. *Support for nested and long Duration transactions:* Most of the RDBMS's do not support for long and nested transactions, but OODBMS's support for nested and long duration transactions [9].
9. *Development of object algebra:* Relational algebra is based on relational mathematics and fully implemented, but object algebra has not been defined in proper way. Five fundamental object preserving operators are [12]: union, difference, select, generate and map.

d) Weaknesses

1. *Coherency between Relational and Object Model:* Relational databases are founded in every organization. To overcome relational databases, object databases have to be providing coherent services to users to migrate from relational database to object database. Architecture of Relational model and Object model must be provide some coherency between them [9].
2. *Optimization of Query Expression:* Query expression optimization is done to increase the performance of the system [13].Optimization of queries is very important for performance gains. But due to following reasons it is difficult to optimize queries in object databases:
   - User defined data types
   - Changing variety of types
   - Complex objects, methods and encapsulation
   - Object Query language has nested structure
   - Object Identity
3. *No fixed query algebra for OODBMS's:* Due to lack of the standards in OODBMS, there is no standard query algebra for OODB. Lack of standard query algebra becomes one of the reasons for problem of query optimization. There are different query languages for different object databases.
4. *No full-fledged query system:* Query system also not fully implemented. Some query facilities lacks in Object databases like nested sub-queries, set queries and aggregation function [9], [11].
5. *No facility for Views:* In relational databases, views are temporary tables. Object databases having no facility for views. An object oriented view capability is difficult to implement due to the features of Object Model such as object identity. Object oriented programming concepts like inheritance and encapsulation makes the difficult to implement views in object databases [11].
6. *Security problems in Object databases:* Security is related to authentication, authorization and accounting. Discretionary Access Control (DAC), Mandatory Access Control (MAC) security policies are implemented in object databases to secure data. In some systems provide security on the basis of object oriented concepts like encapsulation. Object database having to facilities for authorization [9].

7. *No support for schema evolution with OODBs:* Most object databases do not allow schema evolution. Schema Evolution is facility which allows changing the schema at run time such as adding a new attributes or methods to the class, adding new superclass to the class.
8. *Consistency constraints mechanisms are not fully implemented:* Only limited numbers of features are provided by OODBMS's for uniqueness of constraints, integrity constraints and other enterprise constraints [11].
9. *No full-fledged facilities to implement complex objects:* No doubt, object oriented databases provide some facilities to implement the concept of complex objects. But there is no full –fledged implementation of complex objects [11].
10. *Interoperability between OODB and Object Oriented Systems:* In Object Oriented Programming objects are transient in nature. To provide persistent to data, OODB and OO systems need to be interoperable. Many problems may arise during interoperable between OODB and OO systems [11].
11. *Limited performance gains over RDBs Decrease in performance:* Performance gains changes application to application. Applications that make the use of object identity concept having performance gains over RDBMS's. But application that requires bulk database loading and does not make use of OID then performance of OODBMS's is not good [9].
12. *Some basic features are not present:* Some basic features like triggers, meta data management [11] and constraints such as UNIQUE and NULL [9] not present in object databases.

VIII. Comparison of OODBMS and RDBMS

Parameter	OODBMS	RDBMS
Model	OODB Model	Relational Model
Standards	Lack of standards	Fully standardized
OO Programming	Direct and extensive	No Direct Support
Name Of Standards	ODMG-3.0	SQL2(ANSI X3H2)
Conceptually Foundation	No particular	Relational Mathematics
Relationships	Support only for simple and Complex relationships	Support only for simple relationships
User Defined Types	Full support for UDTs	Less Support for UDT's
Advanced Applications	Full support for advanced applications	No support for advanced applications
Complex Objects	Support for complex objects, but not to full extent	Less support for complex objects
Navigational Access	Good	Poor
Schema Evolution	Easy	Difficult
Transactions	Long-lived	Short-lived
Query Language	Depend upon product	SQL
Recursive Queries	Easy to handle	Difficult to handle
SQL	computationally complete	Not computationally complete
Normalization	No need	Strongly recommended
Store data in	Object	Table
Representation	Strong representation for real world entity	Poor representation for real world entity
Semantic nature	Semantically very strong	Semantically very weak
Constraints	Integrity constraints and Enterprise constraints are fully implemented	Integrity constraints and other Enterprise constraints are not fully implemented
Algebra	Object Algebra	Relational Algebra
Operations	Easily extension of operations	Fixed set of operations due to relational algebra
Multimedia data	Full support for multimedia data	Less support for multimedia data

IX. Summary and Conclusion

No doubt, relational databases are very popular and there are found everywhere. The object oriented database comes into action in mid-1985's to remove the limitations and to support some advanced database applications like CAD, CASE etc. Another point that provides the momentum to develop object based database is popularity of object based programming. So; researchers in the field of database think to combine the object oriented programming concepts with database to make powerful database management systems. Different approaches are adopted by industry to make the database with object oriented features. These approaches include relational extensions and pure object oriented are most popular to develop object oriented database systems. OODBMS's are made by many vendors by using different approaches. OODBMS's removes the limitations of RDBMS's, also provide support for advanced database application with some additional features. But due to the lack of standards, they do not get much popularity in the industry. Then after some time, some limitations are found in object oriented database management systems.

X. Acknowledgement

The Author would like to special thanks Sardar Chet Singh Damesha, Sardarni Sachiar Kaur Damesha and whole Damesha family for inspiring me everytime and always with me.

References

Books

1. Object Oriented Database Systems: Approaches and Architectures by C.S.R. Prabhu
2. Database System: A Practical Approach to Design, Implementation and Management by T. Connoly and C. Begg.

Research Papers

3. Atkinson M., Bancilhon F. ,Dewitt D. ,Dittrich K., Maier D. and Zdonik S. "The Object Oriented Database Manifesto" December 1989
4. Atkinson, M., et. al., "The object-oriented database system manifesto," in Proc. Int. Conf. On Deductive and Object-Oriented Databases, 1989.
5. Bancilhon, F., "Object Oriented database systems," in Proc. 7th ACM SIGART/SIGMOD Conf., 1988.
6. Kim, W., "A foundation for object-oriented databases", MCC Tech. Rep., N.ACA-ST-248-88, Aug. 1988.
7. Stefik, M. and Bobrow, D.G, "Object-oriented programming: Themes and variations," The AI Mag., Jan 1986.
8. Zdonik, S.B. and Maier, D., eds., Readings in Object-Oriented Database Systems, San Mateo, CA: Morgan Kauffman, 1989.
9. Kim, W., "Object-Oriented Database Systems: Promises, Reality, and Future," in Modern Database Systems: The Object Model, Interoperability and Beyond, pp.255-280, Kim, W., ed., ACM Press, Addison Wesley, 1995.
10. Bertino, E., Negri, M., Pelagatti, G., and Sbattella, L., "Object-Oriented Query Languages: The Notion and the Issues," IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 3, 1992.
11. Dittrich, K.A., and Dittrich, K.R., "Where Object-Oriented DBMSs Should Do Better: A Critique Based on Early Experiences," in Modern Database Systems: The Object Model, Interoperability and Beyond, pp. 238-252, Kim, W., ed., ACM Press, Addison Wesley, 1995.
12. Scholl, M. and Schek, H., "A relational object model," in Abiteboul, S. and Kanellakis, P.C., eds., in Proc. 3rd Int. Conf. On Database Theory, volume 470 of Lecture Notes in Computer Science, pp. 89-105, Springer Verlag, 1990.
13. Erlingsson, U., "Object-Oriented Query Optimization," unpublished manuscript.