Preventing Code Injection Attack by Inheritance in Access Specifiers Model to Secure Data in Object Oriented Databases

A Dissertation Proposal

Submitted By

Hardeep Singh

To

Department of CSE/IT

In fulfilment of the Requirement for the Award of the Degree of

Master of Technology in CSE

Under the guidance of

Roshan Srivastava

May/2015

ABSTRACT

A vast domain of new application area is emerging in the market. These application area are too complex, sophisticated and demand database system with a much deeper, structural and functional foundation of capabilities than conventional databases offer. This acts as a motivating force for object orientation of database in which the capabilities of object oriented languages are integrated with databases system, therefore the problems involved in providing security for object oriented databases have been receiving very great attention from researchers because data are most sensitive part of system and its loss can have disastrous consequences. Object Oriented Databases are databases which stores the data in form of objects and very different from conventional databases. Different OODBMS's having different security models to secure the data depends upon their implementation approaches. There are main two security policies are found in OODB's are Discretionary Access Control (DAC) and Mandatory Access Control (MAC) to secure the data. Third approach is based on concepts of Object Oriented Programming paradigm like data hiding, encapsulation etc. In this research work, third approach security model Access Specifier Model is used to provide security to data inside object database. This model uses access specifier's like public, private and protected to provide security. One limitation is found in this model is code injection by inheritance. To remove this limitation, another access specifier 'sealed' is used to restrict inheritance and to stop code injection attack by inheritance.

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of research work depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this research work.

I would like to show my greatest appreciation to my research work mentor, **Mr. Roshan Srivastava** .I can't say thank you enough for the tremendous support and help. I felt motivated and encouraged every time I attended his meeting. Without his encouragement and guidance this research work would not have materialized.

I'm highly grateful to **Mr. Dalwinder Singh,** Head of Department, for his thorough guidance right from day 1 to the end of research work. He actually laid the ground for conceptual understanding of research work.

Hardeep Singh

DECLARATION

I hereby declare that the dissertation-2 entitled, **Preventing Code Injection Attack by Inheritance in Access Specifiers Model to Secure Data in Object Oriented Databases** submitted for the M.Tech Degree is entirely my original work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree or diploma.

Date:

Name: **Hardeep Singh**

Reg. No. : **11000410**

CERTIFICATE

This is to certify that **Hardeep Singh** has completed M.tech dissertation-2 proposal titled **Preventing Code Injection Attack by Inheritance in Access Specifiers Model to Secure Data in Object Oriented Databases** under my guidance and supervision. To the best of my knowledge, the present work is the result of his original investigation and study. No part of the dissertation proposal has ever been submitted for any other degree or diploma.

The dissertation-2 proposal is fit for the submission and the partial fulfilment of the conditions for the award of M.Tech Computer Science & Engineering.

Date:

Signature of Advisor:

Name:

UID :

TABLE OF CONTENTS

LIST OF FIGURES

S. No. **Page No.**

LIST OF TABLES

Chapter 1
INTRODUCTION

In beginning of 21st century, there are tremendous changes in every field such as Engineering, Science, Banking, Education and Business. These changes happened due to the use of computer technology in every field. This computer technology includes many fields in it like database, networking and artificial intelligence and many more. Database technology is very much important to the evolution of computer technology. The mostly used terms in database technology are defined below:

1.1 Database

A database is collection of related data in some organized form. The data is organized so that it can fulfil the requirements of processing the information. Some examples of databases are:

a) Dictionary

b) Student Register Record

c) Address Book

1.2 Database Management System (DBMS)

These are the computer programs or software applications are used to capture, store, manage and analyse the data by interacting with applications and users. Some examples of DBMS's are:

a) DB2 from IBM Corporation

b) SQL SERVER from Microsoft Corporation

1.3 Three Views of Data

The concept of abstraction is very important in every field.The meaning of abstraction amount of information to be visible or amount of information to be hide.Abstraction in database is also very important for many different purposes.Abstraction helps us to understand the different concepts with less complexity because it hides some information. For Example: a country shows minimum of its internal details and hides its details. This is

upper level of the abstraction which shows minimum details and hides maximum information from outside. This is also least complex level and easy to understood. A country has states which represents the next level of abstraction. This level not hide more details than upper level and somewhat difficult to understand. The lower level of abstraction in country is cities and villages where no detail is hidden from outside. This level involves maximum details and difficult to understand. Similarly, abstraction is applied to database architecture to understand it in simple way.

Before defining the different levels of database, here need to understand the definition of schema and instance of database

a) Schema: It refers to the entire structure of database in which it is defined that what kind of data can be stored in database and information about attributes and other information etc.

b) **Instance:** At a particular time, data present in the database is called the instance of the database. It can be changed by insertion, deletion and updating to the database.

Three levels of database are:

a) **Physical Level:** This is the lowest level of database .This level tells how data (123, LPU) can be stored on physical storage like Hard Disk, CD-DVD and Tape Drive. It also defines the size of data items on physical storage and location of data items on storage device .This level is helpful to developer of the database and the administrator of the database.

b) **Conceptual Level:** This is the middle level of database architecture known as conceptual level which lies between physical level and user level. This level is used to define the structure of the database. This level is used to define what kind of data can be input to the database and information of the attributes and so on.

c) **User Level:** This is highest level of database and also of abstraction. This level mainly interacts with users of the database. A user may be a may not interested in structure of database like what are the attributes of the table in particular database because it can confuse user with so many details. A user interacts with database through applications which provides interface to end users. Sometimes, different views are provided to different users due some reasons like security and only provide required information to the users. For this purposes, Virtual tables are used in the database.Thsese are also called custom tables because their according to the requirement of the database.

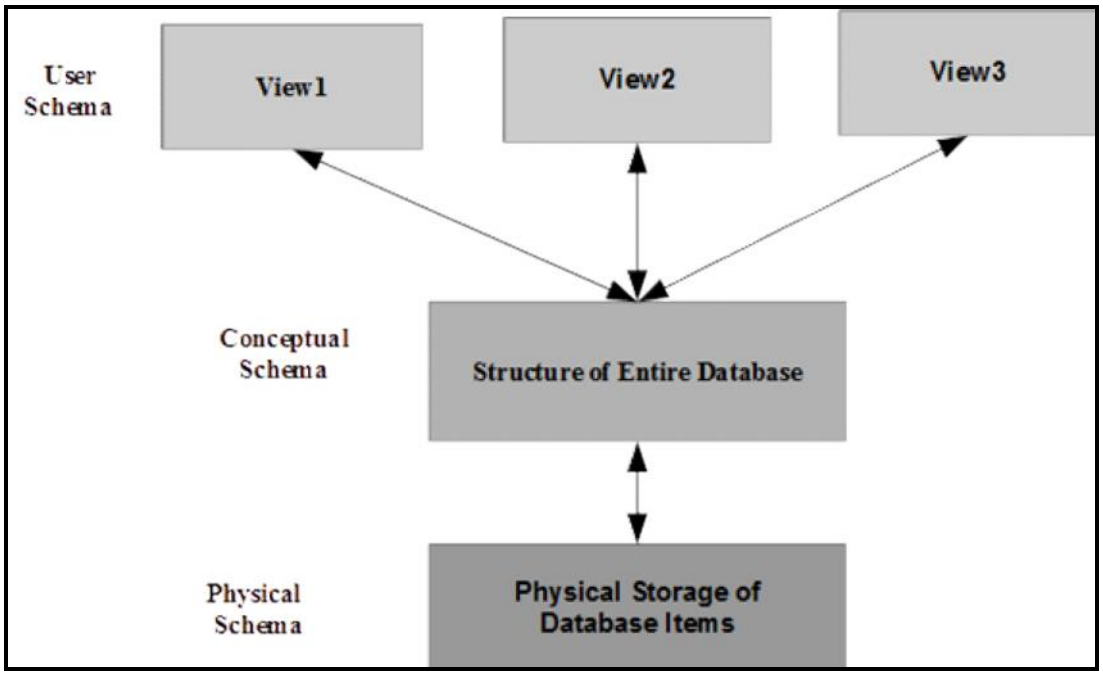

Figure 1: Three Views of Database

1.4 History of Databases

History of database 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. Now, here list of different technology to process data till now:

a) Unit Records & Punch Card Databases

The growing amount of data gathered by the 1880 US Census (which took human tabulators 8 of the 10 years before the next census to compute) saw Herman Hollerith kick start the data processing industry. He devised “Hollerith cards” (his personal brand of punchcard) and the keypunch, sorter, and tabulator unit record machines. The latter three machines were built for the sole purpose of crunching numbers, with the data represented by holes on the punch cards. Hollerith’s Tabulating Machine Company was collaborating with three other companies into International Business Machines (IBM), an enterprise that casts a long shadow over this history of databases.

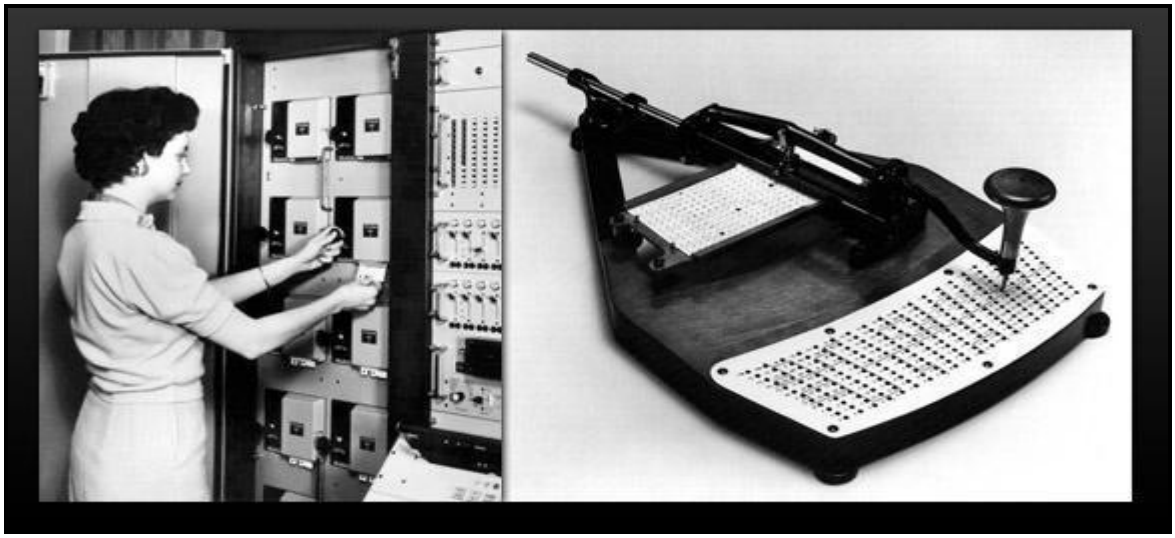

Figure 2: Punched card reader (L) and writer (R)

b) Punch Card Proliferation, Paper Data Reels, & Data Drums

After this revolution in data processing started by the punch cards. Then all companies become interested to revolutionize their administrator and services to customers. From 1910 to mid-1960, punch cards and tabulating mechanism are used for data processing. In this era IBM Company started using reels of punched tapes with use of punch cards. Then they also started using magnetic tapes in which data is recorded magnetically along the tape. This magnetically stored data can be easily accessed.

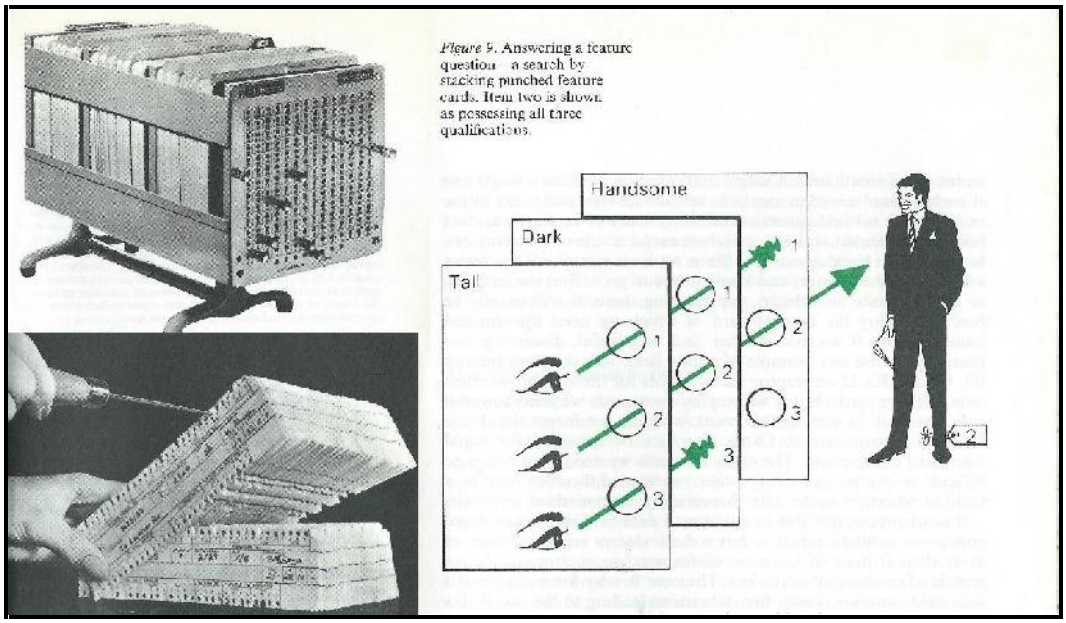

Figure 3: Punch Card Proliferation, Paper Data Reels, & Data Drums

c) File Systems

A file system is used to organize the data in the form of files and directories. In file systems, a hierarchy is used to maintain files and directories. One of the early file systems, Electronic Recording Machine Accounting (ERMA) mark1 is made to keep track of banking and libraries records.

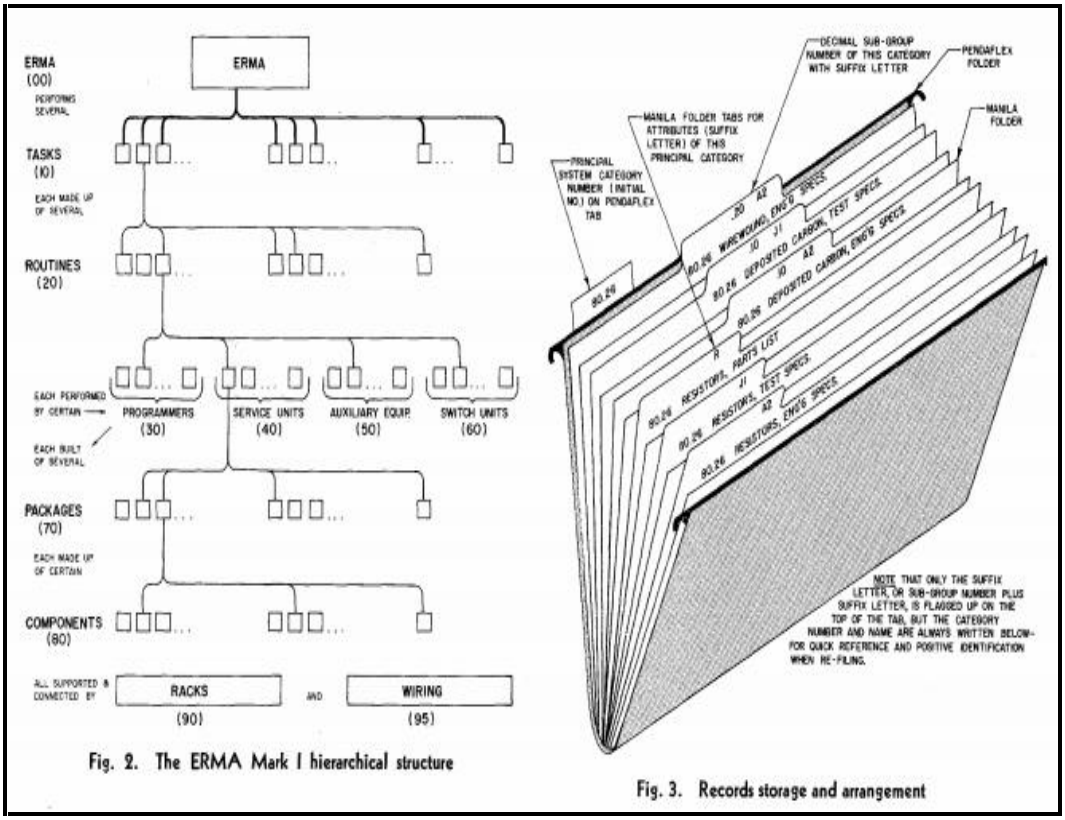

Figure 4: File Systems

d) Database Management System

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.

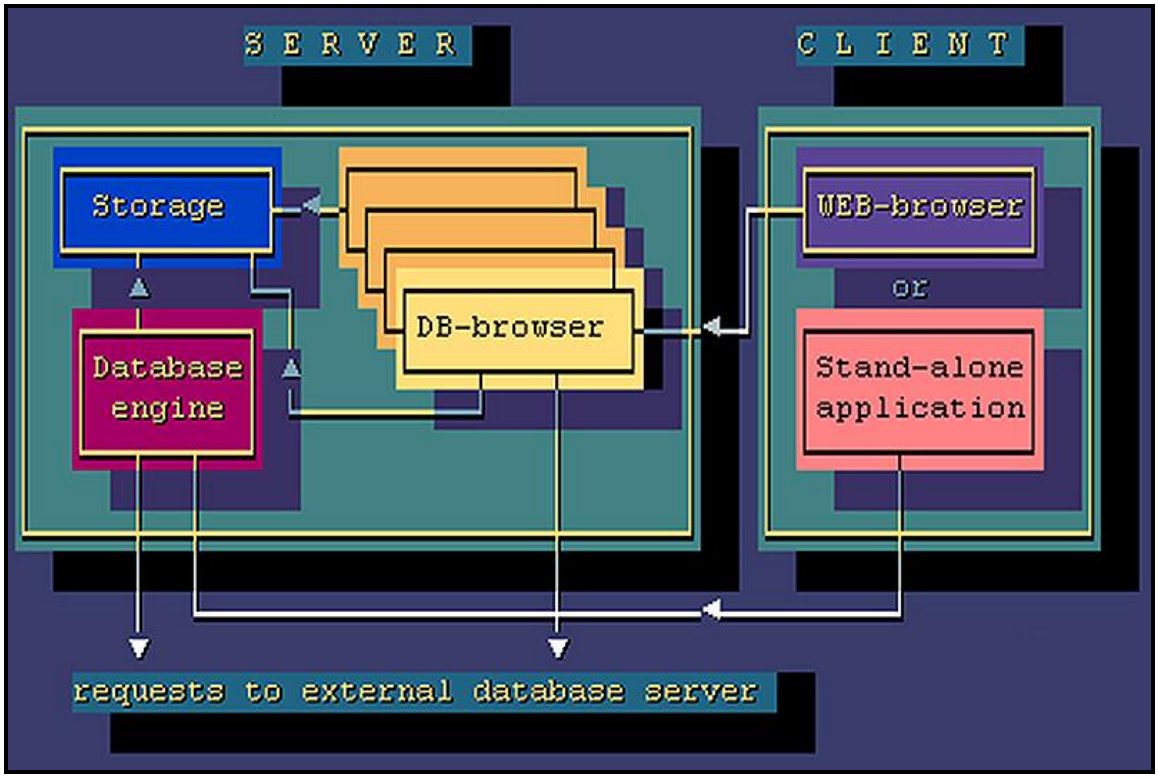

Figure 5: DBMS server client interaction

e) Object Oriented Database Management Systems

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 OODB comes in the picture. At that time Object Oriented Programming paradigm is very much popular. Due to this researcher think that to combine the capabilities of database and object oriented programming paradigm. In Object oriented 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.

f) NoSQL and NewSQL

Relational model is very much popular due to its mathematical foundation. Almost everywhere there is existence of relational database management systems.Now days organization are moving towards technologies of NoSQL and NewSQL due the reason is that It makes a better friend of the ad-hoc changes and dynamism demanded by a growing enterprise than the relational database does. Conventional databases having no support for Big data. That's why industry needs to be come up with these technologies.

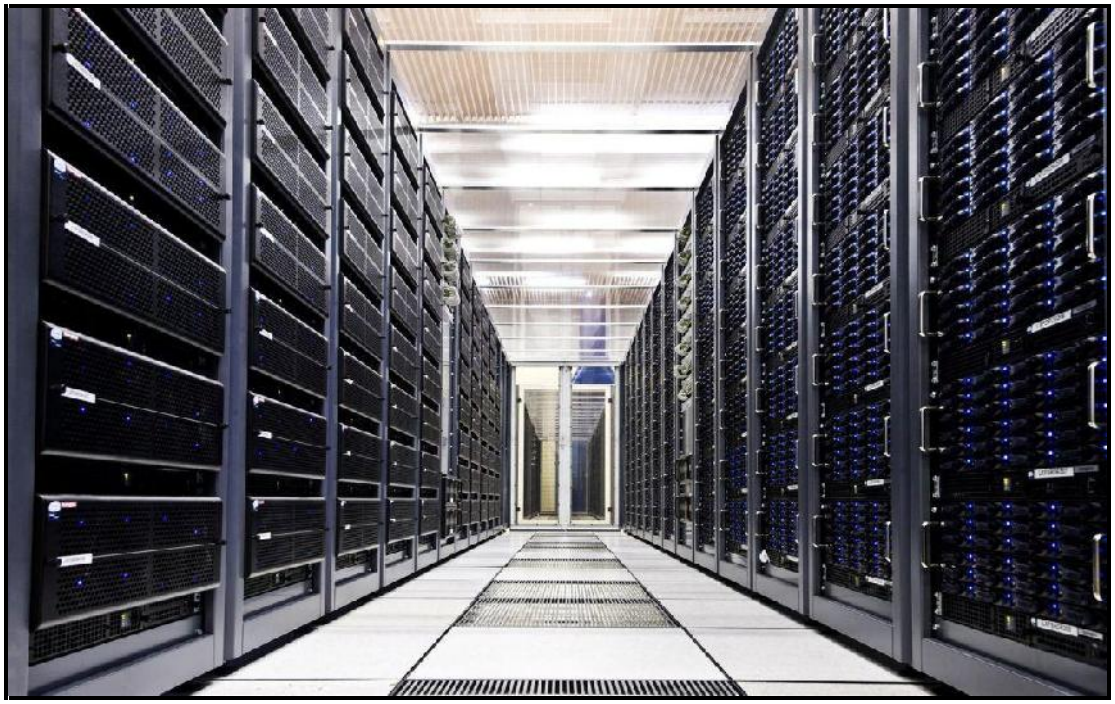

Figure 6: Data Centre

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 etc.

1.5 Object Oriented Database Management Systems

Why there is need of object oriented databases as industry have already relational databases? Basically there are three reasons for this:

a) Limitations of RDBMS

b) Need for Advanced Applications

c) Popularity of Object Oriented Paradigm

a) Limitations of RDBMS

These limitations are in relational model. Due to this these limitations are reflected to all RDBMS .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 sometimes creates relations that do not correspond to entities in "real world": Need for normalization in RDBMS to maintain the consistency of the database, but some broken relations are not related to real world.

3. Semantic overloading: RDM has only one construct that is table for representing data and relationship that is table. Due to this, sometimes it becomes very difficult to find out that what is model by relation. Is it data or relationship?

4. Poor support for integrity and enterprise constraints: Constraints are very much needed for database have to be desired data. RDM supports only limited number of constraints. The enterprise constraints are the industry standard constraints.

5. Homogeneous data structure: RDM requires homogeneous data structures like:

a) RDM assumes both horizontal and vertical homogeneity.

b) RDBMS having only a limited set of operations. These operations cannot be extended because due to relational mathematics algebra.

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 database**.**

8. Difficulty in handling recursive queries: There is very poor support to handle recursive queries in RDBMS. For this you must know:

a) Depth of recursive query must be known.

b) You can use the transitive closure operations to handle recursive queries in RDBMS.

9. Impedance mismatch: SQL Data Manipulation Language (DML) is lack in computational completeness. To overcome this situation, sometimes it must need to 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 of short durations and concurrency control methods, protocols or mechanisms are not suitable 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.

b) Need for Advanced Applications

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

1) **Computer Aided Design (CAD)**

a) In these applications, relevant data about buildings, airplanes and integrated circuit chips is stored. In this type of applications, database design may be very large.

b) Design in these types of applications is not static. This design is evolves through the times. Updates need to be propagated.

c) These applications require version control and configuration management.

d) These applications require complex objects which are not supported by conventional database management systems. For example, a car's component may be related to other components.

e) Need long duration transactions because sometimes updates are for reaching.

f) Support for cooperative engineering because most of the times many people work on same design.

2) Computer Aided manufacturing (CAM)

a) These application data is very much similar to CAD, but needs discrete production.

b) These applications must respond to real time events.

c) Generally algorithms and custom rules are used to respond to a particular situation.

3) Computer Aided Software Engineering (CASE)

a) These applications store data about the different phases of software development life cycle.

b) Design may be extremely large.

c) Involves cooperative work.

d) Need to maintain dependencies among components.

e) Version control support is required with configuration management.

4) Network Management Systems

a) Coordinates the delivery of communication services across the network.

b) Performs different tasks like network path management, network planning and problem management.

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

c) Popularity of Object Oriented Paradigm

Another domain that enforces the development of OODBMS is popularity of object oriented programming paradigm. This is due reason that you can model real life situations in best way by using object oriented programming.

OO Programming Aspects:

1. Abstraction: It is process of identifying essential properties of an entity and ignoring unimportant details such as implementation details. The property comprises 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 and information hiding: An object contains both data items and set of methods used to manipulate it. It is called encapsulation.

Information Hiding: It is process of breaking external aspects from its internal details of an object, so that it can be hidden from outside. These two concepts also related with abstraction.

Importance: These two concepts provides the facility to change internal details of an object to be without affecting applications that use it by providing external details remain same. It also allows data independence. Encapsulation and data hiding can be achieved by access modifiers.

Access Specifiers: These are special keywords which are to set the accessibility of a class, variable and methods etc. in Object Oriented Programming paradigm. The types of access modifiers are:

- **Public:** This type of member can be accessed from anywhere.
- **Private:** it can access by only same class or friend class.
- **Protected:** This type of members can be accessed in same class or derived class.
- **Internal:** This type of members can be accessed in same project only.
- **Protected Internal**: This type of members can be accessed in same assembly or derived assembly.
- **Sealed:** This is used to restrict inheritance from class.
- **Read-only:** No value can be assigned to these types of variables.
- **Static Fields:** A static variable in cannot have an uninitialized value.

4. Attributes: Attributes models the current state of an object. They can be two types:

a) Simple Attribute:

b) Complex Attribute:

5. Object Identity (OID): It is the unique identifier associated with every object in the system. It has following characteristics:

a) It is generated by system.

b) It is unique to that object in the entire system.

c) It is used only by the system, not by the user.

d) It is independent the state of the object.

6. Methods and Messages: This implements the behaviour of an object and involves encapsulation.

Message: It is used to execute one of its methods of an object.

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 can behave like an object with its own class attributes and methods.

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: The process by which super class is formed 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. 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:

a) One to One relationship

b) One to Many relationship

c) Many to One relationship

d) Many to Many relationship

The different programming paradigm during the different decades:

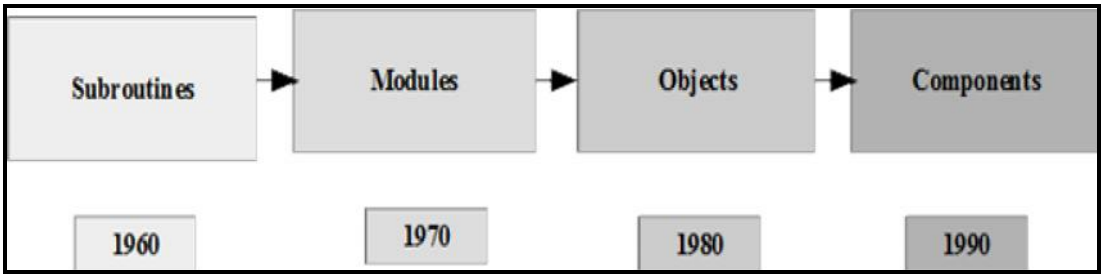

Figure 7: Different Programming Paradigms during Different Decades

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

1.6 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. The advantages of this approach are:

a) Stick to relational model

b) 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:

a) Definition of new data types

b) Definition of new operations of so defined data types.

c) Implementation of access methods.

d) 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:

a) 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).

b) 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.

c) User defined procedure are associated with used defined data types.

d) Example:POSTGRES

b) Object/Relational DBMS (ORDBMS)

These systems provide both object oriented features by the definition. They provide similar objectives as provided by the Relational Extension approach of RDBMS. In this approach, an object layer is built on the top of relational system like Open ODB or 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- 90 and aims to support for broad base applications. It has following features:

a) Based on Iris DBMS

b) Based on Client/Server architectures

c) Both data and applications can be shared by the user applications.

d) Clients use Application Program Interface (API) to access information.

e) OSQL is data manipulation language for Open ODB/ODAPTER.

f) Open ODB uses relational techniques to support to OO features.

g) Object Model is implemented by Object Manager.

h) Mapping to OO schema and queries to relational ones.

i) The underlying relational storage manager is ALLBASE/SQL.

c) Pure OODBMS

OODB systems are not much popular because lack of standards. 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 some definitions are given which are mostly accepted but not standardize.

OODB MODEL: A data model that capture semantics of objects provided by object oriented programming.

A threshold model is presented by ZDONIK and MAIER that an OODBMS must follow:

a) It must provide the facility of basic database functionality like transaction management and concurrency control etc.

b) OODB must support concept of Object Identity (OID).

c) OODB must provide the feature of encapsulation.

d) OODB must support for complex objects.

e) Inheritance not must but may useful.

OODB: An OODB is sharable and persistent collection of objects defined by 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:

a) Gemstone from Gemstone Systems Incorporation.

b) Objectivity/DB from Objectivity Incorporation.

c) O2 from O2 Technology Corporation.

d) Objectstore from Progress Software Corporation.

e) Ontos from Ontos Incorporation.

f) Versant from Versant Corporation.

g) DB4O from Versant Corporation

1.7 The Object Oriented Database Manifesto

Object Oriented Databases not much popular due to lack of standards. This is a de facto type of standard for object oriented database systems. An object oriented database system follows this manifesto to acquire mandatory and optional features which are following:

Mandatory 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:

a) It is generated by system.

b) It is unique to that object in the entire system.

c) It is used only by the system, not by the user.

d) 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 inheritance can be:

a) Specialization

b) Substitution

c) Inclusion

d) Constraint

6. Dynamic Binding: An OODBMS must support concept of dynamic binding in programming language such as:

a) Overloading

b) Overriding

c) Late binding

7. Computationally Complete DML: To provide a support for data processing database use a 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.

Optional Features:

1. Multiple Inheritance: Multiple inheritance is not directly support by multiple object oriented programming languages. An OODBMS can also support for multiple inheritance.

2. Type checking and inferencing: Type checking and inferencing feature can be supported by OODB.

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 OODB's can distributed database features.

5. Versions: An OODBMS can support for version control and configuration management.

1.8 Achievements and Weaknesses of OODBMS

There are some achievements and weaknesses of object oriented database systems as given follow:

Achievements:

1. OODBs provide the facility to users to define abstractions: An OODB has facility to define new abstractions and to control the implementation of these abstractions.

2. OODBs provide the facility of development of some relationships: OODBs offers the feature the inverse relationship to represent a mutual reference between two objects (a binary relationship).

3. No need of user defined keys: There is no need of any user defined keys due the presence of OID in OODB model.

4. OODBs reduce need for joins: OODBs has ability to reduce the need of joins.

5. Performance gains over RDBMS: OODBMS's are not fully implemented due this it has only some situations where OODBMS gains over RDBMS.

6. Provide facility for versioning: The control mechanisms are missing in most of the RDBMS's, but it is supported by the OODBMS's.

7. 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.

8. Development of object algebra: Relational algebra is fully established concept, but object algebra has in its initial stage and provides five basic object operations: select, generate, union, difference, and map.

Weaknesses:

1. Interoperability of OODBs with RDBs: Relational databases are fully implemented systems and mostly used in every industry. But object oriented database systems are not fully implemented systems due to lack of standards. For object oriented databases to establish in the industry, these systems must be compatible with already relational database systems. For this models of relational and object database should be combined.

2. Minimal Query Optimization: Query optimization is very much difficult in object oriented database systems. This difficulty comes into action due the use of OO data model because this model .provide following facilities which make query optimization more difficult:

Additional new data types

a) Changing variety of types

b) Complex objects, methods and encapsulation

c) Object Query language has nested structure

d) Object Identity

3. No standard algebra for querying: For relational databases, there is standard relational algebra for querying. But OODB systems lacks in standards. No Doubt, many languages are made for querying OO databases systems. These query languages are changes from one product to another product due to lack of standards

4. Lack of some features in query facilities: OODB systems do not provide nested sub-queries, aggregation function and set queries in querying facilities.

5. Provide no facility for Views: There is no provision of views in object oriented database systems due to feature object identity present in OODB model and another reason is that inheritance and encapsulation make view definitions unnecessary.

6. Lack of some security features: Most OODBs does not support authorization. This is due to because these are not fully implemented systems.

7. Lack of feature to dynamic class definition changes: No doubt, some OODBs provide the facility of schema evolution which means changes can be made to database schema when it running. But it is not fully supported in OODBs.

8. Provide some features to support for consistency constraints: In relational databases, primary and foreign keys are used for constraints but their few methods to define consistency constraints explicitly.

9. Complex object feature is not fully implemented: No doubt, OODBs support the concept of complex objects. But, it is not fully implemented in OODBs yet.

10. Sometimes difficulty in integration with existing OO programming systems: Sometimes it becomes difficult to integrate OODBs with some OO programming systems. Several problems arise while integrating are:

a) Some Naming conflict may occurs

b) Sometimes need for rewritten of class hierarchies

c) Overwritten of system operations by OODB systems

11. In some situations no performance gains over RDBs: Performance of OODB systems are better in those applications which require OID to access databases and updating operations. For other applications, OODB systems do not shows better results in performance because there is no use of OID in these applications.

12. Lack of other features in OODBs: OODBs systems are not fully implemented systems. These systems do not provide any facilities for meta data management, triggers and some constraints like UNIQUE and NULL.

1.9 **COMPARISON OF OODBMS WITH RDBMS**

Table 1: Comparison of OODBMS with 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
Parameter	**OODBMS**	**RDBMS**
Model	OODB Model	Relational Model
Standards	Fully standardized	Lack of standards
OO Programming	No direct Support	Direct and extensive
Name Of Standards	SQL2(ANSI X3H2)	ODMG-3.0
Conceptually Foundation	No particular	Relational Mathematics

Relationships	Support only for simple relationships	Support only for simple and Complex relationships
User Defined Types	Less support for UDTs	Full Support for UDT's
Advanced Applications	No support for advanced applications	Full support for advanced applications
Complex Objects	Less support for complex objects	Support for complex objects, but not to full extent.
Navigational Access	Poor	Good
Schema Evolution	Difficult	Easy
Transactions	Short-lived	Long-lived
Query Language	SQL	Depend upon product
Recursive Queries	Difficult to handle	Easy to handle
SQL	Not computationally complete	computationally complete
Normalization	Strongly recommended	No need
Store data in	Table	Object
Representation	Poor representation for real world entity	Strong representation for real world entity
Semantic nature	Semantically very weak	Semantically very strong
Constraints	Integrity constraints and other Enterprise constraints are not fully implemented	Integrity constraints and Enterprise constraints are fully implemented

Chapter 2
REVIEW OF LITERATURE

Dr. James Cannady in **"Security Models for Object Oriented Database Systems"** [1] gives a brief description on the different security models used in object oriented database. There are three secrecy related problems:

- Confidentiality
- Integrity
- Denial of Service

Following Security models are discussed in this paper are:

(a) Discretionary Access Control Models: These security models define access restrictions based on the identity of users (individual or groups), type of access (e.g., select, update, insert, delete), the specific object being accessed and other factors like time of day, which application program is used etc. Following DAC models used in OODB for security:

(1) The ORION authorization model: This model is based on providing authorization to users or group of users on basis of explicit authorizations. There are two types of authorizations in this are:

- Positive Authorizations: These authorizations are provided to user or group of users to access an object.
- Negative Authorizations: These authorizations are provided to user or group of users not to access an object.

(2) Data Hiding Model: This model is based on the assumption of providing authorization to users to execute methods on objects. A user can only execute those actions inside a method for which he/she is authorized to do so. Two categories of methods are:

Public Methods: These are those methods on which authorization is granted to users to execute the methods.

Private Methods: These are those methods on which authorization is not granted to users to execute the methods.

(3) Other DAC Models: Joel Richardson gives a model to secure Object Oriented databases which is similar to GRANT/REVOKE Statement model in RDBMS. In this model, creator of object can provide access to other users or group of users to execute specific methods on specific objects.

Rafiul Ahad gives a model which is based on access of execution of methods. Here access control is applied on the methods, not objects. This means a user has access to execute different methods on the database or not.

A final authorization dependent model is given by Dr. Edaurdo B. Fernandez proposed a model having positive authorizations and negative authorizations.

Another DAC Model for securing object oriented databases presented by Dr. Naftaly H. Minsky which is very similar to view access control mechanism in RDBMS. In this model, different laws and rules are given about to access objects.

(b) Mandatory Access Control Models: These security models security policies are used to define access control restrictions on users on the basis of different security levels access and share a database consisting of data at different sensitivity levels. Following MAC models used in OODB for security:

(1) The SORION AUTHORIZATION Model: This is a MAC model for securing Object Oriented databases. In this model, security is provided on basis of subjects and objects and access modes. Different Security levels are provided to Subjects and different sensitivity levels are provided to Objects and then access is allowed on the basis of access modes.

(2) SODA Model: This model works according to different security levels are given to subjects in the system and sensitivity levels are given to objects. After this access is provided on basis of access matrix in which security level of user or subject is checked and compared to sensitivity level of object and then decision is taken on whether allow to access or not.

(3) Polyinstantiation: SODA model is extended to remove the problem of multiparty update conflict. SODA model use the concept of polyinstantiation to remove update

conflict problem. Multiparty Update Conflict problem arises when users of different security levels attempts to access same information. Here a conflict arises about the accessing and modification of objects. This problem is solved by distributing information to different locations with different levels.

Dr. Elisa Bertino in **"Data hiding Model for Securing Object Oriented Database Systems"** [2] discussed about a Discretionary Access Control (DAC) model which is based on categories of methods. These categories of methods are important because they provide different approaches to access the data due to which there is an access control on data. Two categories of methods are:

- Public Methods
- Private Methods

Public Methods: These are those methods on which authorization is granted to users to execute the methods.

Private Methods: These are those methods on which authorization is not granted to users to execute the methods.

This model is based on the assumption of providing authorization to users to execute methods on objects. A user can only execute those actions inside a method for which he/she is authorized to do so. This means that access control is also applied upto to the instructions of the methods. In this way, better security can be provided. In model is somewhat similar to GRANT and REVOKE statements in Relational Databases. In this model, user who has created the object has privileges to grant authorizations to other users. Creator of the object has also ability to revoke the authorizations from different users to execute methods. User who has created the object has privileges to show only specific information to different users not all the information about a particular object.

Dr. Thomas F. Keefe in **"Secure Object Oriented Database (SODA) Model "** [3] tells that it is first model that is different from other models like Discretionary Access Control(DAC),Mandatory Access Control (MAC) for securing Object Databases because it is based on the concept of Object Oriented Paradigm. This model is integrated with MAC to provide better security in multilevel security system. In MAC, some classification levels are assigned to the data. In this model, these classification levels of

data are assigned by making the use of core concept of object oriented programming that is inheritance. Here multiple inheritance is not included in SODA model.

This model works like DAC and MAC models, different security levels are given to subjects in the system and sensitivity levels are given to objects. After this access is provided on basis of access matrix in which security level of user or subject is checked and compared to sensitivity level of object and then decision is taken on whether allow to access or not.

Premchand B. Ambhore , B.B. Meshram, V.B. Waghmare in **"An implementation of Object Oriented Database security"** [4] said that unlike conventional databases, secure OODBMSs having some specific properties that make them unique. There are many standardized security models are available for RDBMSs, but only a less number of models have been specifically designed for OO databases from which there is no standardization. The proposed security models make use of the concepts of the encapsulation, inheritance, information hiding, methods and ability to model real world entities that are present in object oriented environments. OODBMSs use unique characteristics that allow these models to control the access of data in the database. They incorporate features such as flexible data structure, inheritance and late binding. Access control models for OODBMSs must consistent with such features .Users can define methods, some of which are open for other users as public methods. The OODBMS can encapsulate a series of basic access commands into a method and make it public for users, while keeping basic commands themselves away from users. Some security models for OODBMSs are explicit authorization, Data hiding model and DAC (Data Access Control) models. Some other models are like MAC models. The object oriented database model also permits the classification of an object's sensitivity through the use of class and instance. When an instance of class is created, the object can automatically inherit the level of the sensitivity of superclass. Although, the ability to pass the classification through inheritance is possible in object oriented databases, class instances are usually classified at higher level within the object's class hierarchy. This prevents the flow control problem, where information passes from higher to lower level classifications.

Security Policies in OODB

a) Discretionary Access Control Policies

b) Mandatory Access Control Policies

a) Discretionary Access Control Policies

Roshan K. Thomas and Ravi S. Sandhu in **"Discretionary Access Control in Object Oriented Databases"** [5] study that these type of security policies define access restrictions based on the identity of users (individual or groups), type of access (e.g., select, update, insert, delete), the specific object being accessed and other factors like time of day, which application program is used etc. Discretionary Access Control (DAC) models can be represented by the access matrix model developed by Lampson in 1971 and further refined by the Graham and Denning.This model defines the an access matrix in which the rows represent subjects (Users, Processes), the columns represents objects (Files, Records, Programs, Subsystems) and the interaction of row and column contains the access types that the subject has authorization for with respect to the object. There are three access control issues in the Discretionary Access Control Model are:

a) **Subject to Object:** Interested to know that how a subject establishes an initial authorized point of contact with an object.

b) **Inter-Object:** Inter-Object access control is concerned with issues such as the visibility and the propagation of authorization across object boundaries, as consequences of an initial subject to object authorization.

c) **Intra-Object:** Deal with access control within in the internal structure and behaviour of an individual object.

For example: Table 2: Access Matrix for Access Rights

Subject/Object	**File**	**Record**
Administrator	Create, Read, Update, Delete	Create, Read, Update, Delete
User	Read, Update	Read, Update
Guest	Read	Read

b) Mandatory Access Control Policies

M.B. Thurisingham in **"Mandatory Security in Object Oriented Database Systems"** [6] study that these types of security policies are used to define access control restrictions on users on the basis of different security levels access and share a database consisting of data at different sensitivity levels. The sensitivity levels may be assigned to the data depending on the content, context, aggregation and time. These MAC are based on multilevel security. This policy ensures that users only can access that information for which they are authorized. The earliest of MAC security policies is The Bell and LaPadula security model is stated below:

- Active entities are called Subjects (such as users and processes) and passive entities (files and records) are called Objects.
- Subjects and Objects are assigned security levels. The set of security levels form a partially ordered lattice:

Unclassified<Confidential<Secret

The mandatory access control requirements are formalized by two rules, the first of which protects data from unauthorized disclosure and the second of which protects data from contamination:

a) **A subject S has no provision to access class c for read data operation unless class(S) ≥ c:** A subject can only read an object if the subject's security level dominates the security level of the object.

b) **A subject S has no provision to access class c for write data operation unless class(S) ≥ c:** A subject can only write an object if the subject's security level dominates the security level of the object.

Mansaf Alam in **"Access Specifiers Model for Data Security in Object Oriented databases"** [7] proposed a security model for Object Oriented Databases .OODB is the concept of both Object Oriented languages and Database. The concept of access specifier in Object Oriented languages used for data hiding. This feature can be applied to Object Oriented Databases to provide security at conceptual level and physical level. In this model three access specifiers public, protected and private are used to secure the data from unauthorized access. Figure 8 shows the flow of the data from one structure

(schema) to another schema. In this type modelling of database gives more security to the database by controlling the access of data flow in this way no one can access data without changing the access mode of data. The data founded in private section is highly confidential. The data in protected section can be accessed with permission of the owner of that data and private data can be accessed by anyone. A database is basically designed with protected access specifiers.Then User is allowed to access the data from the base database to another database with the help of friend method. Data can write, read and executed with feature of friend method. In Figure 8 four schema's are taken as DEPT, EMP1, EMP2 and EMP3.Schema's DEPT is main schema from which other two schema's are inherited like EMP1 inherited publicly and EMP2 privately. Fourth Schema EMP3 is inherited from EMP1 and EMP2 in mixed mode (Public, Protected).Which attribute will be public, private or protected in which schema depends upon the type of inheritance. The visibility of base class and derived class and type of derivation show below in given table.

Table 3: Base and Derived Class visibility and types of Derivations

Base Class Visibility	**Derived Class Visibility**		
	Public Derivation	**Private Derivation**	**Protected Derivation**
Private	Not Inherited	Not Inherited	Not Inherited
Protected	Protected	Private	Protected
Public	Public	Private	Protected

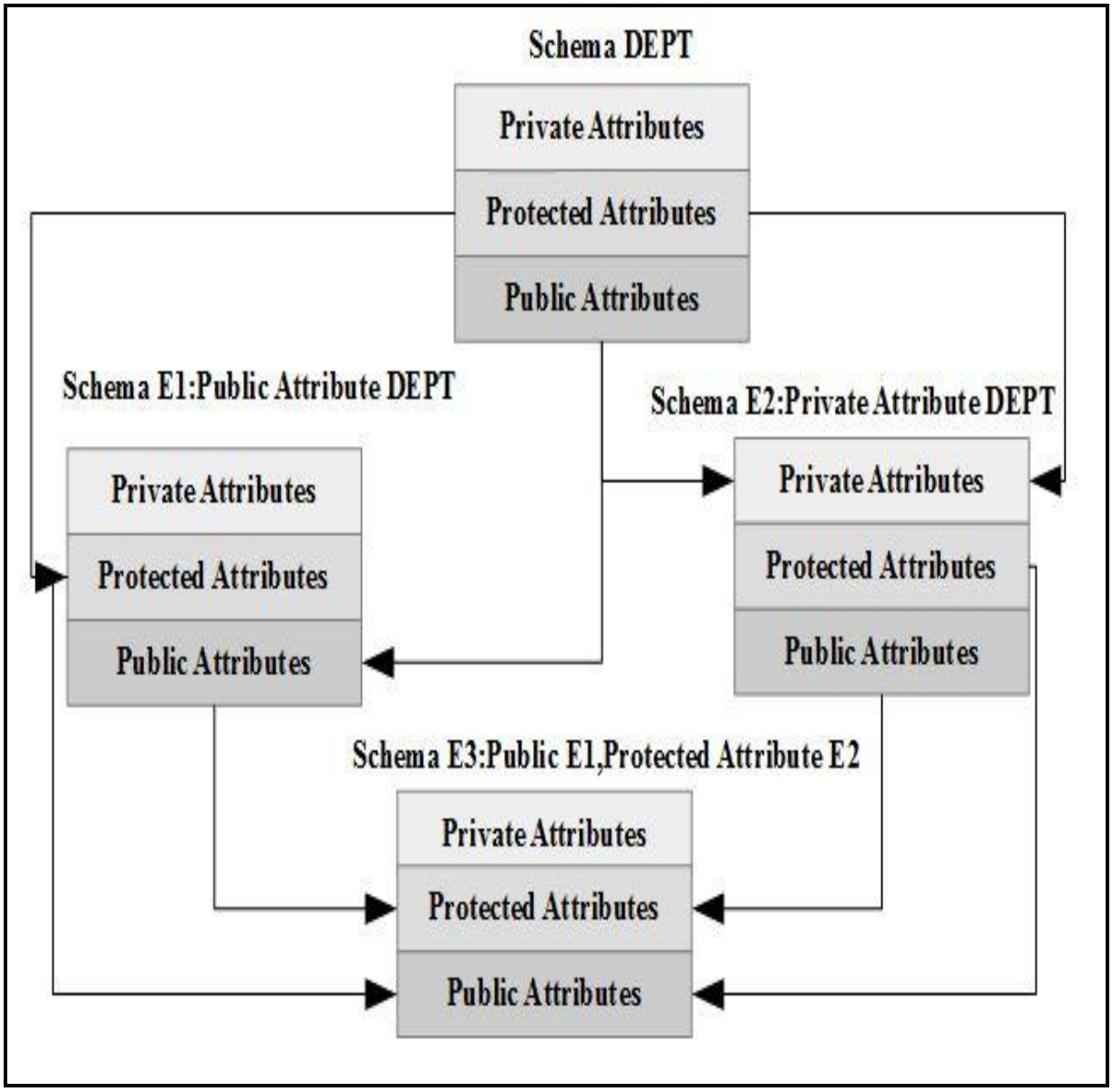

Figure 8: Data hiding Model in Object Oriented Database

Chapter 3
PRESENT WORK

3.1 Problem Formulation

After studying the different models for the security for object oriented database systems found that the Access Specifiers Model for the security of object oriented database systems is different from models previously present in OODB's because it is based on the features of object oriented programming. It is based concept of Access Specifiers used in object oriented programming paradigm. Access Spcecifiers are used to set the accessibility of a class, variable and methods. Access Specifier's also used for encapsulation of components. Access Specifiers model uses only three access specifiers public, private and protected to secure object oriented databases. Another big limitation found in this model is that code injection attack by inheritance is easily possible when security is provided by this model.

3.2 Objectives

Code injection by inheritance: Code injection is a computer bug that comes into existence when some invalid data is processed. The main motive behind code injection is to changing the normal course of execution. Due to change in normal execution process, many security problems may arise. Code injection attack by inheritance can be used by an attacker or hacker to access data inside the database in object oriented database systems. In Object Databases, data inside database can only be accessed by methods because due to concept of Data Hiding in Object Oriented Paradigm. If methods are available to access data, then only data can accessed in Object Oriented databases, otherwise not. Concept of inheritance says that methods and data of the base class can be got to derived class depends upon the inheritance mode. This means that methods that are used to access the data of base class also present in derived class easily and derived class method can access the data inside base class. In Object databases, Class is a schema. A class can contain very confidential data depends upon the context in which it is used. If this class is inheritable, a class no longer is confidential. Because a derived class (other schema) have permissions to access the data of this confidential class. So, loss of confidential information or data can be disastrous. It attack can be used in various ways depends upon context of attacker or code injector. For example, scenario of University management System can be taken as below to show code injection attack by inheritance:

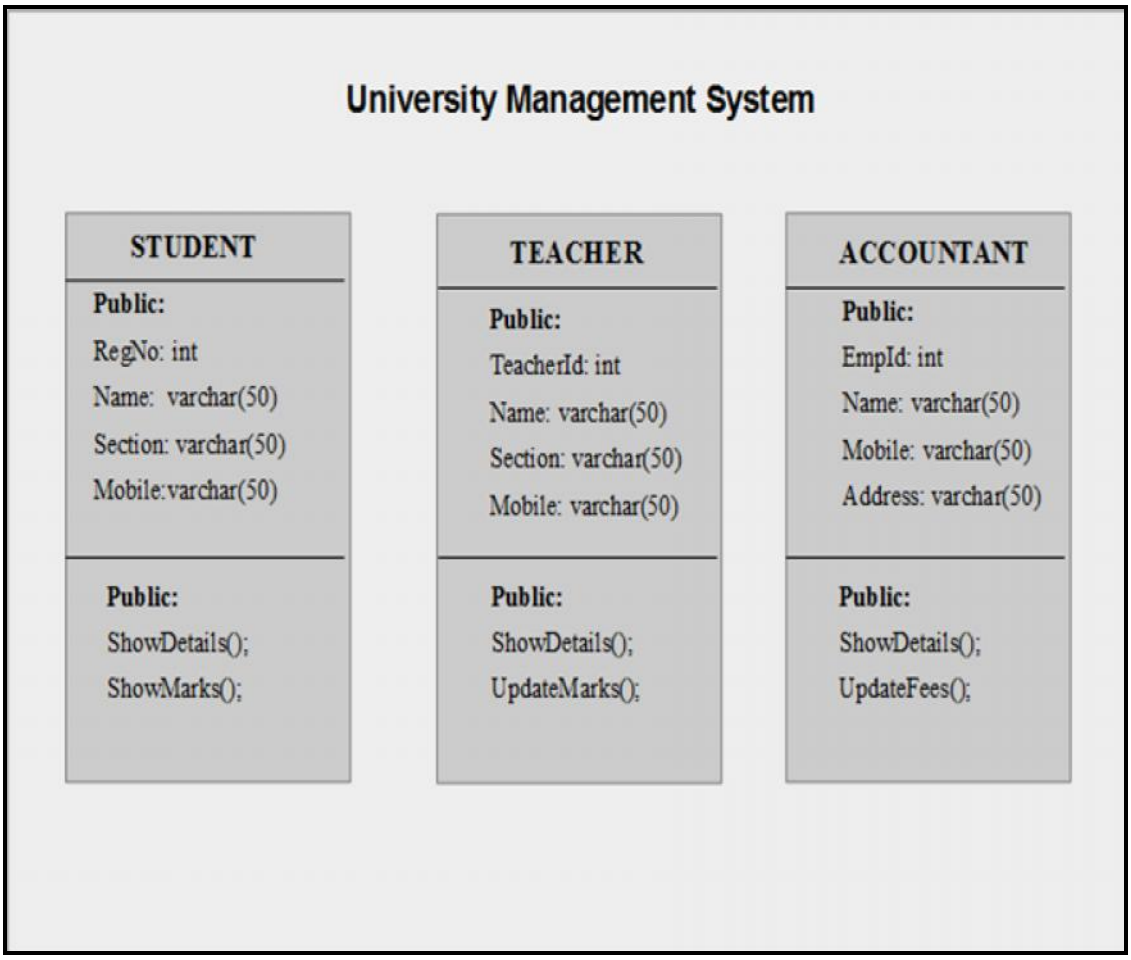

Figure 9: Class schema in Object Oriented Database

Here in a University Management System. We have three entities Student, Teacher and Accountant. All entities having their own roles in university. These Entities are defined inside three different classes like STUDENT; TEACHER AND ACCOUNTANT.A Teacher has ability to upload marks obtained by student in each subject and update if any changes are happen there. No other entity having this privilege. Remaining two entities having their own functions to complete their role in university. Now, suppose privilege to update marks also gained by student by unfair means then but happens. This can cause very serious problems in University Management System and whole system having different results and different consequences due to which objectives of UMS violated. This situation can easily happen in UMS when security is provided by Access Specifiers Model. Because a TEACHER class can be inherited by another class for example say CODE INJECTION due to which this class also acquire a function to update marks. This

means that another entity or object outside the UMS can also update the marks. This scenario shows in given below figure:

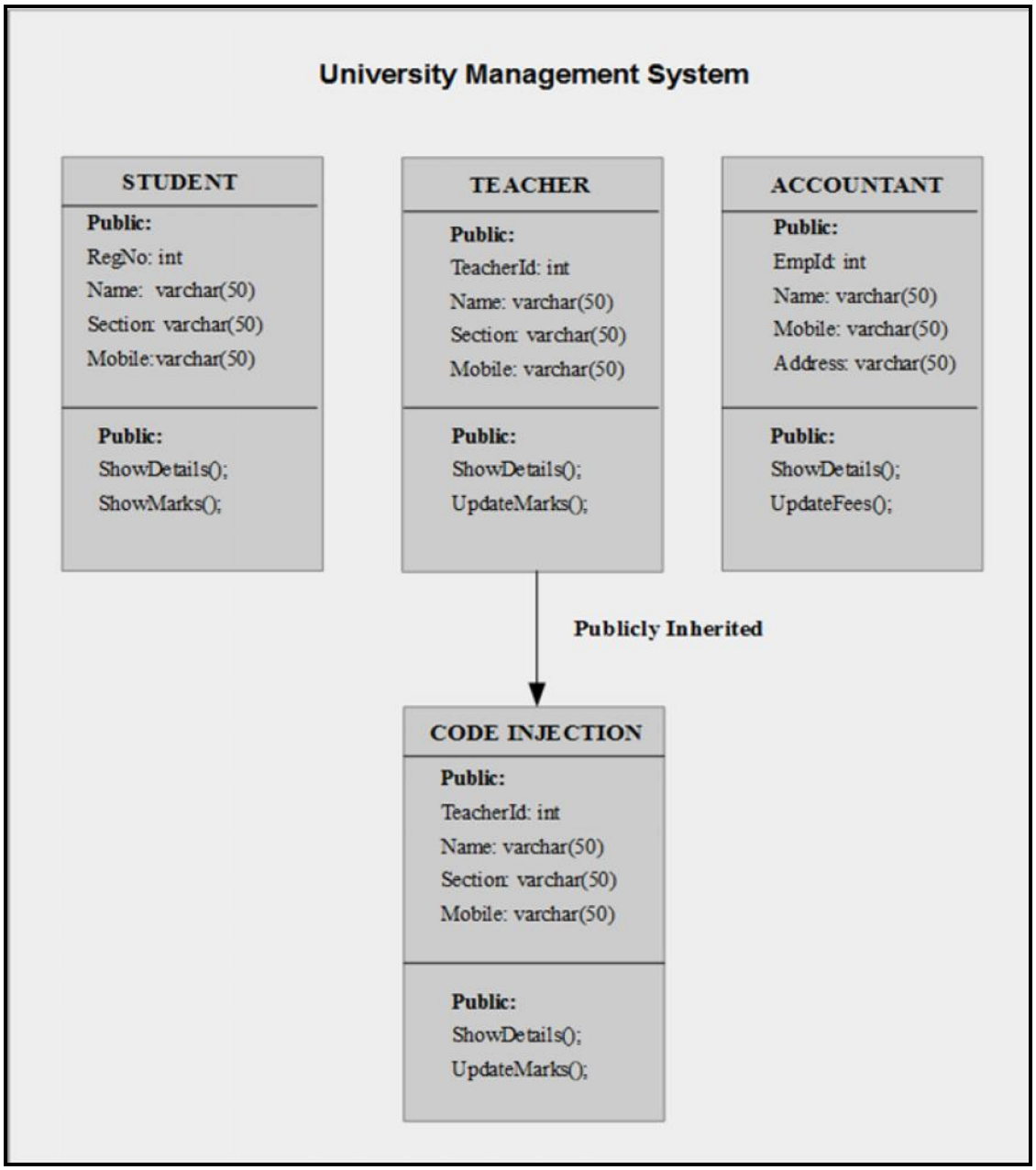

Figure 10: Inheriting one class from other in Object Oriented Databases

The main objective of our study is to remove the limitation of Access Specifiers model by making some changes to this model so that this model can be saved from code injection of inheritance which can cause serious problems in any system.

3.3 Methodology

3.3.1 Formulation of Hypothesis

After studying about the Object Oriented Database System and their security models found that are not much popular due to the lack of standards. Different Object oriented Database systems uses different approaches or models to secure the data inside the object oriented databases. Discretionary Access Control and Mandatory Access Control are main two approaches are used to provide security to object oriented databases. Third approach is based on characteristics of Object Oriented Programming Paradigm like encapsulation, inheritance etc. Latest security model founded in Object Oriented Databases is Access Specifiers Model is based on third approach to secure Object Databases. Limitation of this model is discussed in the objective of study.

A hypothesis is formulated that if some changes are made to Access Specifiers Model to remove its limitation of code injection by inheritance, then this model can made more secure than previously discussed in research work. As code injection attack is caused by inheritance, so efforts have to be made to restrict inheritance, then only code injection attack by inheritance can be stopped.

3.3.2 Research Design

As discussed in formulation of hypothesis, some changes should be made to Access Specifiers Model to make it more secure and save from code injection attack by inheritance. In Object Oriented Programming paradigm study that data hiding and encapsulation can be done with help of Access Specifiers and more access specifiers can be used in Access Specifers Model to make it more secure. As discussed in objective of study, to provide more security to data in Access Specifiers Model, restrictions should be applied on inheritance. This restriction can be provided by using another access specifier ‘sealed’ in Access Specifers Model. Due to this, any class cannot be further inherited and restriction is applied on inheritance and there is no chance to attack by code injection by inheritance. Research design can be draw by taking the scenario of University Management System in objective of study:

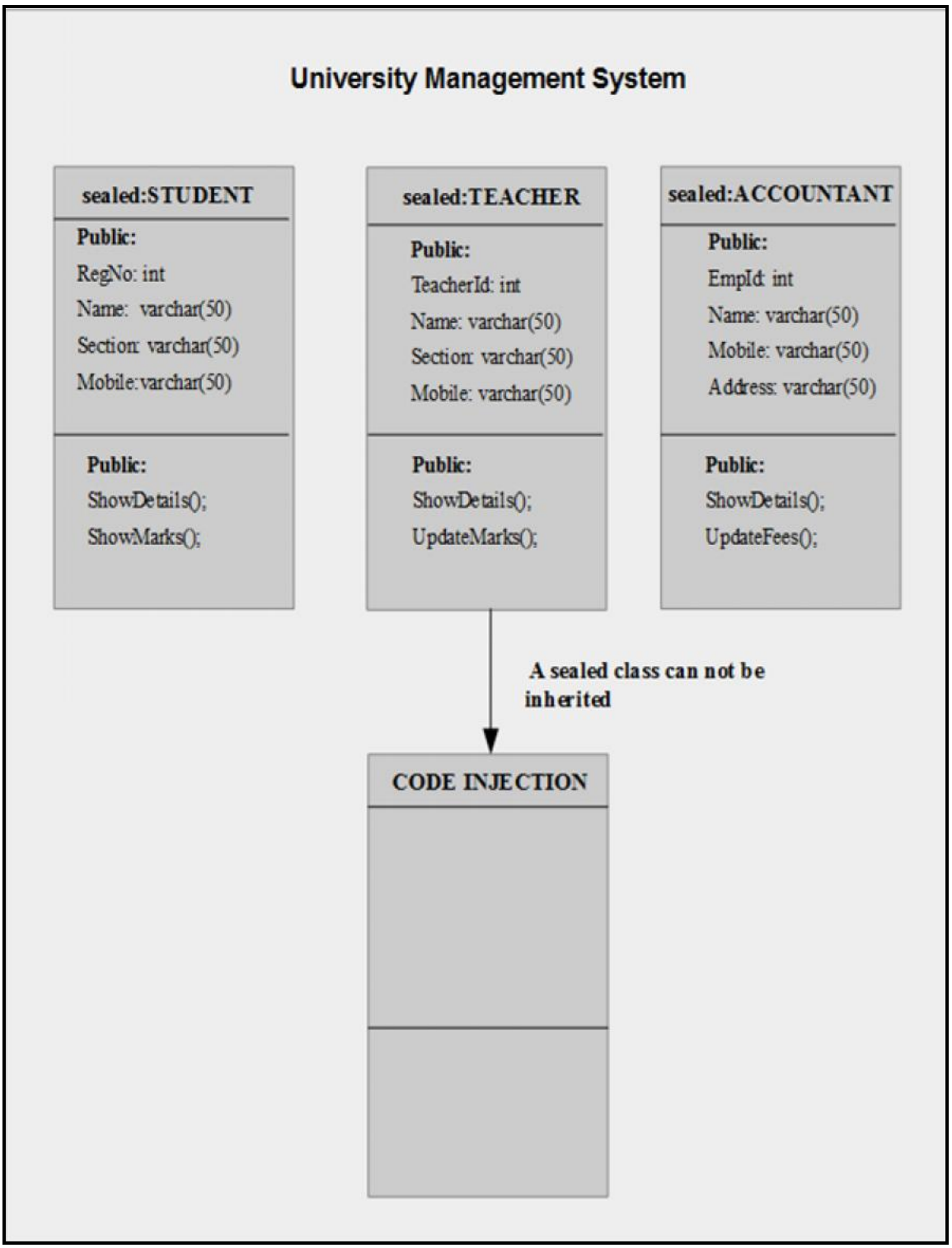

Figure 11: Prevention of inheritance using “sealed” modifier

Now there is no chance of attack of code injection by inheritance because TEACHER class cannot be inherited because restrictions are applied by sealed access specifier.

3.3.3 Software Requirement Specifications

Following are the specific software and hardware requirements for the project to work effectively.

The system must have following software requirements:

a) Client OS with .NET framework 4.0

b) DB4O native .NET mono open source object database

Chapter 6
RESULTS AND DISCUSSIONS

This research work is compared to older policies like DAC and MAC to provide security to Object Oriented Databases. In this comparison, it is found that new research work is better than older research work in providing security to object databases.

Table 4: Comparison of Previous and New Research Work

Parameter	Previous Work	New Research Work
Security Level	Security may be at External level, Physical Level or at Conceptual Level	Here security is provided at conceptual level which means security is stronger.
Security Policies	DAC or MAC policies are used secure object oriented databases.	Here security is provided by the specific properties of object oriented databases due to which are at fine grained level and very strong.
Security Models	DAC or MAC security models are used to provide security.	Here latest model known as "Access Specifiers Model "is used to provide security which based on concept of public, private and protected access specifiers to secure the object database. This models further extended in our research work to enhance more security.
Concept used for security	Concept of access matrix model, Positive or Negative and Internal or External methods is used.	Here concept of data hiding is used to secure object oriented which means security is provided by definition and it will be stronger.

Then comparison of Access Specifiers Model is done with Extended Access Specifiers Model to found that which is better to provide security to object databases.

Table 5: Comparison of Access Specifiers Model with Extended Access Specifiers Model

Parameter	**Access Specifiers Model**	**Extended Access Specifiers Model**
Code Injection Attack by Inheritance	In "Access Specifiers Model for secure Object database" only three access specifiers public, private and protected are used due to which code injection attack by inheritance is possible	Here we have extend the "Access Specifiers Model for secure Object database" to use four access specifiers public, private ,protected and sealed are used due to which code injection attack by inheritance is prevented using sealed access specfier.
Access Specifiers	Less secure due the use of less access specifiers public, private and protected.	More secure due the use of more access specifiers public, private, protected and sealed.

Future Scope	Can be extended using more access modifiers like internal and protected internal to provide better security according to domain of the user	Can be extended using more access modifiers like internal and protected internal to provide better security according to domain of the user

After comparison, it is found that Extended Access Specifiers Model provides better security than Access Specifiers Model.

Chapter 7
CONCLUSION AND FUTURE SCOPE

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 oriented database is the popularity of object oriented programming. 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. 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. Security is one of the limitations founded in object oriented systems.

Different OODBMS's having different security models to secure the data depends upon their implementation approaches. There are main two security policies are found in OODB's are Discretionary Access Control (DAC) and Mandatory Access Control (MAC) to secure the data. Third approach is based on concepts of Object Oriented Programming paradigm like data hiding, encapsulation etc. In this research work, third approach security model Access Specifier Model is used to provide security to data inside object database. This model uses access specifier's like public, private and protected to provide security. One limitation is found in this model is code injection by inheritance. To remove this limitation, another access specified 'sealed' is used to restrict inheritance and to stop code injection attack by inheritance.

In future, security can be enhanced by using these policies (DAC and MAC) with object oriented programming concepts. DAC and MAC policies can be used along with access specifier model to secure the data in object oriented databases. To enhance security of data integrates the DAC and MAC policies with access specifier's model to provide security at conceptual level by developing an algorithm at conceptual level with the help of access specifier model. By this scheme security of OODB's can be enhanced.

Chapter 8
REFERENCES

I. Books

[1] C.S.R Prabhu **"Object Oriented Database Systems: Approaches and Architecture"** New Delhi :PHI Learning;2011

[2] T. Connoly and C. Begg. **"Database System: A Practical Approach to Design, Implementation and Management"** Pearson;2009

[3] Jim Paterson, Stefan Edlich, Henrik horning and Reidar horning **"A Definitive Guide to DB4O"** Apress; 2006

II.Research Papers

[1] Dr. James Cannady in **"Security Models for Object Oriented Database Systems"** journal of Data Security Management, 11, (10), 38-44

[2] Dr. Elisa Bertino in **"Data hiding and Security in Object Oriented Database Systems"** Proceedings of Eighth International Conference on Data Engineering, 1992 IEEE

[3] Thomas F. Keefe, Bhavani M. Thuraisingham, Wei-Tek Tsai in **"Secure Object Oriented Database (SODA) Model"** Secure Query-Processing Strategies. IEEE Computer 22(3): 63-70 (1989)

[4] Premchand B. Ambhore, B.B. Meshram, V.B.Waghmare **"A Implementation of Object Oriented Database Security"** Proceedings of Fifth International Conference on Software Engg. Research, Management and Applications, 2007 IEEE

[5] Roshan K. Thomas and Ravi S. Sandhu **"Discretionary Access Control in Object Oriented Databases"** Proc. of 16th NIST-NCSC national Conference on Computer Security, Baltimore, MD, Sept 1993

[6] M.B. Thurisingham **"Mandatory Security in Object Oriented Database Systems"** Proceedings of OOPSLA,ACM,October 1989

[7] Mansaf Alam **"Access Specifiers Model for Data Security in Object Oriented databases"** Proceedings of International Conference on information technology, New Generations, 2011 IEEE

Chapter 9
APPENDIX

a) **RDBMS-** Relational Database Management Systems

b) **OODBMS-** Object Oriented Database Management Systems

c) **DAC-** Discretionary Access Control

d) **MAC-** Mandatory Access Control