International Journal of Advance Research, IJOAR .org
Volume 4, Issue 4, April 2016, Online: ISSN 2320-9194

# Query re-engineering with virtual database object

Sandhya Pandharinath Patkar, Prof. Sushila Ratre

*Author Details*
*Sandhya Pandharinath Patkar is currently pursuing masters degree in Computer Engineering in Mumbai University, India, PH- +91-89758 20758. E-mail: patkar.sandhya@gmail.com*
*Prof. sushila.ratre is currently working as a professor for Computer Engineering Department in Mumbai University, India, PH- +91-9657381250. E-mail:sushila.ratre@gmail.com*

## ABSTRACT

Data models of numerous business enterprises encompass recursive data structures in the form of hierarchies and networks. Processing such data stored in relational databases is not easy, since the relational algebra and calculus do not provide adequate facilities. Object-relational mapping in computer's software is a programming technique for converting data between incompatible type systems in object-oriented programming languages. In market there are many ORM techniques are implemented such as – Active Record, Hibernate, Active JDBC and many more. In this paper we present thorough evaluation of all these solutions using data of various sizes up to million records. Since all these methods require writing complex code if applied directly, we have described prototype and integration of different object- relational mapping systems.

## KeyWords

Active Record, ORM, Object Relational Mapping, re-engineering, removal of nested queries, removal of complex joins, virtual database object.

## 1. INTRODUCTION

[1]At the beginning of a creation of an information system in the analysis and design phase, an analytical model of the future information system is being created and designed. At this stage the main problems and needs of the customer for whom the future of IS is created are already specified, and main requirements to establish the information system are defined as well. Suitable solutions to write down these requirements are use case diagrams, which use scenarios of usage to describe how the user uses specific system functionality.

Use case diagrams are, using the use case realization process, transformed into an analytical model that focuses on understanding the functional requirements, creating an almost ideal picture of the future system, in the form of classes and subsystems. For the purposes of future implementation of the system the object model, which consists of classes, their properties and interrelations, is an important output of the analysis and design phase. Today, the most common approach when implementing an information system object-oriented programming is used. Therefore the object model is a suitable model for developers who are developing the future system. On the other hand an information system is very often connected to some kind of relational database that uses a relational model. However, there are quite a few inconsistencies and significant differences between object and relational models. These differences can be found in the implementation phase and manifest themselves as differences between object-oriented technologies and methods of storing data in the form of relations in relational databases. These differences and inconsistences are therefore known as object relational impedance mismatch.

Object-relational mapping (ORM, O/RM, and O/R mapping) in computer software is a programming technique for converting data between incompatible type systems in object-oriented programming languages. This creates, in effect, a "virtual object database" that can be used from within the programming language. There are both free and commercial packages available that perform object -relational mapping, although some programmers opt to create their own ORM tools.

In the paper a fuzzy expert system based on information from the object model will be proposed. This expert system shall propose appropriate data types in a relational database and generate a relational schema, in which objects are mapped.

## 2. Related Work

**RDBMS** - [3]Most of us use a relational database every day. Relational technology is a known quantity, and this alone is sufficient reason for many organizations to choose it. But to say only this is to pay less respect than is due. Relational databases are entrenched because they're an incredibly flexible and robust approach to data management. Due to the complete and consistent theoretical foundation of the relational data model, relational databases can effectively guarantee and protect the integrity of the data, among other desirable characteristics. Some people would even say that the last big invention in computing has been the relational concept for data management as first introduced by E.F. Codd (Codd, 1970) more than three decades ago.

[3]Relational database management systems aren't specific to Java, nor are a relational database specific to a particular application. This significant principle is known as data independence. In distinctive words, and we can't defend this admissible fact padding, data lives longer than any application does. Relational technologies that constitute parts of the uniform application (the transactional engine and the reporting engine, for example). Relational technology is a hack denominator of many diverse systems and technology platforms. Hence, the relational data model is constantly the hack enterprise-wide cross section of profession entities.

A relational database is a database that has a accumulation of tables of data items, all of which is formally described and organized according to the relational model. Data in a single table represents relation, from which the name of the database type comes. In typical solutions, tables may have beyond defined relationships mutually with each other.

In the relational model, each table schema must detect a column or aggregation of columns, called the primary key, to uniquely identify each row. A relationship can then be established between each row in the table and a row in another table by creating a foreign key, a column or aggregation of columns in one table that points to the primary key of another table. The relational model offers diverse levels of refinement of table organization and reorganization called database normalization. The database management system (DBMS) of a relational database is called an RDBMS, and is the software of a relational database.

**Problems with RDBMS in design of complex system**

[3]Data models of numerous business enterprises encompass recursive data structures in the form of hierarchies and networks. They five and dime store data on e.g. railway networks, bill of accessories and product categorization. Their certain storage format can be recommended from a superabundance of proposals. There are distinctive ways to query such data. Obviously, a steadfast 3GL client code can be written. Then, the data processing is done on the client side. In this situation a significant amount of complicated source code should be created, debugged and maintained. This constantly causes a notable increase of the economical and a bend in the delivery schedule. Therefore, a server side solution is called for. It was eventual as extensions to SQL, e.g. Oracle's CONNECT BY clause or recursive Common Table Expressions eventually adopted in SQL: 1999. Such extensions have been implemented in beyond number database systems.

Simultaneously the academia worked on optimization methods for such queries. However, there are likewise database managements systems that do not uphold recursion in queries, e.g. MySQL. Since they are widely adopted and used, applications programmers often face the question how to query their recursive data. As noted before, they can choose to hardcode acceptable

logic in the application. In spite of deceptive simplicity of these solutions, it causes merely troubles: lower efficiency, increased charge and difficulty, as well as drained maintainability.

**Unrolling -**

[6]There are database management systems that do not respond recursive queries by the whole of MySQL as the most famous example. The wide adoptions of the LAMP paradigm of web development make us convinced that numerous web enterprises have to write extra code that queries recursive data. Most of such projects have redolent data, e.g. nested categories of summer stock items, posts in argumentation forums or income sharing chains in multi-level marketing. In verdict to plug such applications, we have created a number of methods to run queries to such data even against database management systems that lack this feature.

**Example:-**

**Self-Join in SQL:** A self-join is a join in which a table is joined with itself (which is also called Unary relationships), especially when the table has a FOREIGN KEY which references its own PRIMARY KEY. To join a table itself means that each row of the table is combined with itself and with every other row of the table.

The self-join can be viewed as a join of two copies of the same table. The table is not actually copied, but SQL performs the command as though it were. The syntax of the command for joining a table to itself is almost same as that for joining two different tables. To distinguish the column names from one another, aliases for the actual the table name are used, since both the tables have same name. Table name aliases are defined in the FROM clause of the SELECT statement. See the syntax:

```
SELECT a.column_name, b.column_name...
 FROM table1 a, table1 b
WHERE a.common_filed = b.common_field;
```

Consider the following example for the above:

| Column Name | Data Type | Nullable | Default | Primary Key |
|---|---|---|---|---|
| EMP_ID | VARCHAR2(5) | No | - | 1 |
| EMP_NAME | VARCHAR2(20) | Yes | - | - |
| DT_OF_JOIN | DATE | Yes | - | - |
| EMP_SUPV | VARCHAR2(5) | Yes | - | - |

1 - 4

| Constraint | Type | Table |
|---|---|---|
| SYS_C004074 | C | EMPLOYEE |
| EMP_ID | P | EMPLOYEE |
| EMP_SUPV | R | EMPLOYEE |

Primary key

Foreign key
Referencing EMP_ID of this table

**Figure 1: Structure of the table**

In the EMPLOYEE table displayed above, emp_id is the primary key. emp_supv is the foreign key (this is the supervisor's employee id). If we want a list of employees and the names of their supervisors, we'll have to JOIN the EMPLOYEE table to itself to get this list.

**Unary relationship to employee**

How the employees are related to themselves:

An employee may report to another employee (supervisor). An employee may supervise himself (i.e. zero) to many employees (subordinates).

We have the following data into the table EMPLOYEE.

| EMP_ID | EMP_NAME | DT_OF_JOIN | EMP_SUPV |
|---|---|---|---|
| 20051 | Vijes Setthi | 15-JUN-09 | - |
| 20073 | Unnath Nayar | 09-AUG-10 | 20051 |
| 20064 | Rakesh Patel | 23-OCT-09 | 20073 |
| 20069 | Anant Kumar | 03-DEC-08 | 20051 |
| 20055 | Vinod Rathor | 27-NOV-09 | 20051 |
| 20075 | Mukesh Singh | 25-JAN-11 | 20073 |

**Figure 2: Data present in the EMPLOYEE Table**

The above data shows:
Unnath Nayar's supervisor is Vijes Setthi.
Anant Kumar and Vinod Rathor can also report to Vijes Setthi.
Rakesh Patel and Mukesh Singh are under supervison of Unnith Nayar.

**Example of SQL SELF JOIN:**

In the following example we will use the table EMPLOYEE twice and in order to do this we will use the alias of the table. To get the list of employees and their supervisor the following sql statement has used:

**SELECT a.emp_id AS "Emp_ID",a.emp_name AS "Employee Name",b.emp_id AS "Supervisor ID",b.emp_name AS "Supervisor Name"**
**FROM employee a, employee b**
**WHERE a.emp_supv = b.emp_id;**

**Output:**

| Emp_ID | Employee Name | Supervisor ID | Supervisor Name |
|---|---|---|---|
| 20055 | Vinod Rathor | 20051 | Vijes Setthi |
| 20069 | Anant Kumar | 20051 | Vijes Setthi |
| 20073 | Unnath Nayar | 20051 | Vijes Setthi |
| 20075 | Mukesh Singh | 20073 | Unnath Nayar |
| 20064 | Rakesh Patel | 20073 | Unnath Nayar |

**Figure 3: Output of the self-join query on employee table**

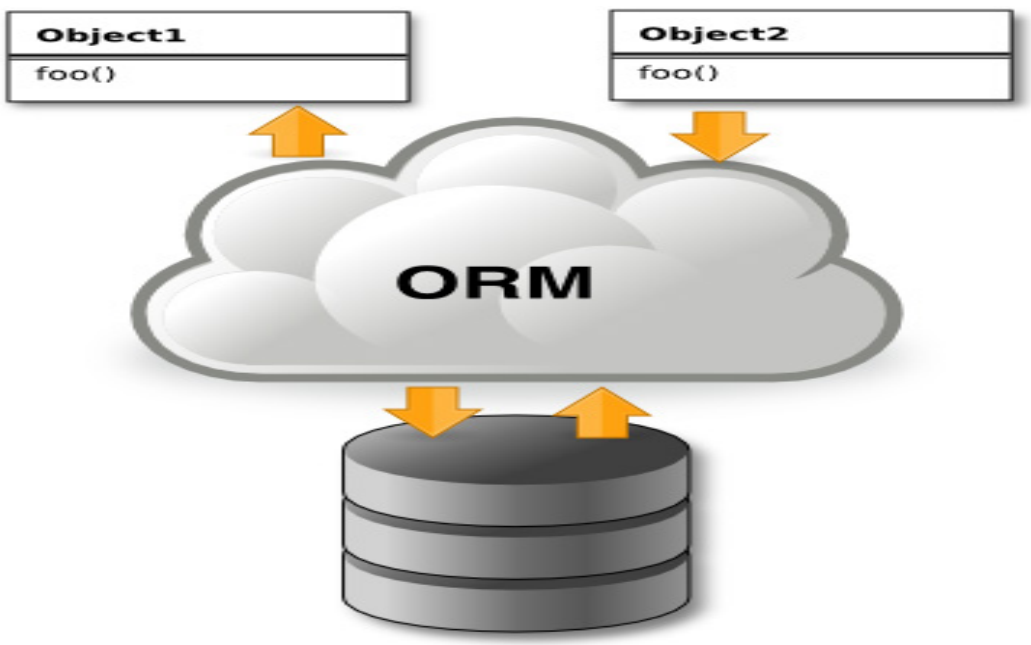



**Figure 4: ORM**

## 3. ORM

[1][2][5]Object-relational mapping (ORM, O/RM, and O/R mapping) in computer science is a programming stratagem for converting data between incompatible type systems in object-oriented programming languages. This creates, in ending, a "virtual object database" that can be handled from within the programming language. There are both free and commercial packages at hand that pound object-relational mapping, during the time some programmers opt to create their own ORM tools.

The heart of the problem is translating the logical representation of the objects into an atomized form that is capable of being stored in the database, while preserving the properties of the objects and their relationships in case they can be reloaded as objects when needed. If this computerized information and retrieval functionality is implemented, the objects are said to be persistent.

There are many ORM available in market some of them are –

- Active Record
- Hibernate
- ADO .NET Entity Framework

**A. Active Record**

[1][3]In software engineering, the active record format is an architectural pattern found in software that stores its data in relational databases. It was suggested by Martin Fowler in his 2003 book Patterns of Enterprise Application. The interface of an object conforming to this pattern would include functions such as Insert, Update, and Delete, with properties that correspond preferably or less promptly to the columns in the underlying database table.

The active record format is an concern to accessing data in a database. A database table or view is wrapped into a class. Thus, an object instance is tied to a single row in the table. After onset of an object, a trendy row is reproduced to the table upon save. Any object loaded gets its idea from the database. When an object is updated the corresponding row in the table is furthermore updated. The wrapper class implements access or methods or properties for each column in the table or view.

This pattern is commonly used by object persistence tools, and in object-relational mapping (ORM). Typically, foreign key relationships will be exposed as an object instance of the appropriate type via a property.

Implementations of the concept can be found in various frameworks for many programming environments. For example, if in a database there is a table parts with columns name (string type) and price (number type), and the Active Record pattern is implemented in the class Part, the pseudo-code –

```
part = new Part()
part.name = “Sample part”
part.price = 123.45
part.save()
```

will create a new row in the parts table with the given values, and is roughly equivalent to the SQL command
INSERT INTO parts (name, price) VALUES (‘Sample part’, 123.45);
Conversely, the class can be used to query the database:

```
b = Part.find_first (“name”, “gearbox”)
```

This will find a new Part object based on the first matching row from the parts table whose name column has the value "gearbox". The SQL command used might be similar to the following, depending on the SQL implementation details of the database:

**SELECT * FROM parts WHERE name = ‘gearbox’ LIMIT 1;** -- MySQL or PostgreSQL

| Model / Class | Table / Schema |
|---|---|
| Post | posts |
| LineItem | line_items |
| Deer | deers |
| Mouse | mice |
| Person | people |

**Figure 5: Architecture of Active Record**

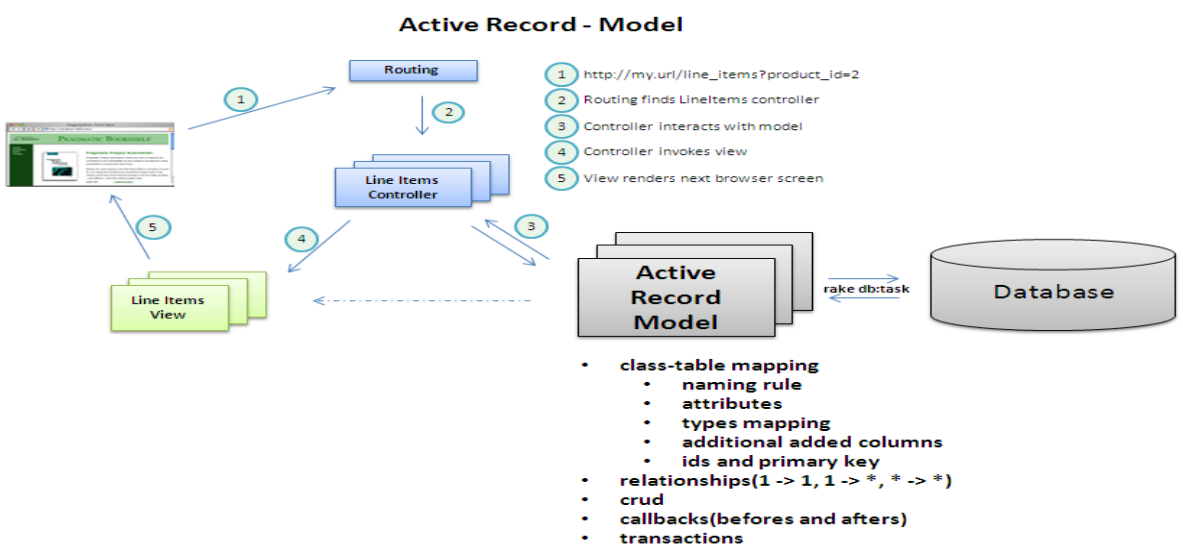
**Convention over Configuration in Active Record**

However, if you inherit the conventions adopted by Rails, you will require to write very little configuration (in some case no configuration at all) when creating Active Record models. The point is that if you configure your applications in the absolutely same way most of the time then this should be the default way. Thus, definitive configuration would be needed only in those cases where you can't follow the standard convention.

**Naming Conventions:-**

By default, Active Record uses several naming conventions to confront out how the mapping between models and database tables should be created. Rails will pluralize your class names to confront the respective database table.

By default, Active Record uses several naming conventions to confront out how the mapping between models and database tables should be created. Rails will pluralize your class names to confront the respective database table.




**Example:-**

Database Table - Plural with underscores separating words (e.g., book_clubs).

Model Class - Singular with the first letter of each word capitalized (e.g., BookClub).

**Figure 6: Naming Conventions in Active Record**

**Schema Conventions:-**

Active Record uses naming conventions for the columns in database tables, depending on the purpose of these columns.

**Foreign keys -** These fields should be named following the pattern singularized_table_name_id (e.g., item_id, order_id). These are the fields that Active Record will look for when you create associations between your models.

**Primary keys -** By default, Active Record will use an integer column named id as the table's primary key. When using Active Record Migrations to create your tables, this column will be automatically created.

**B. Hibernate**

[4][6]Hibernate ORM (Hibernate in short) is an object- relational mapping library for the Java interpretation, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions.

Applications using Hibernate are portable to supported SQL databases with little performance overhead.

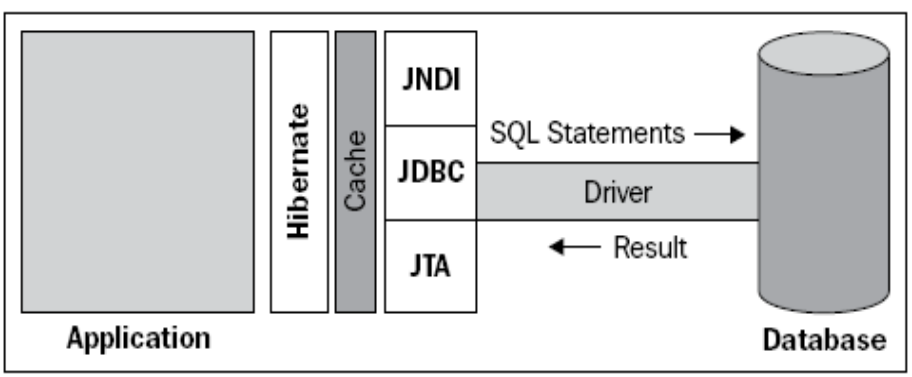



**Figure 7: Architecture of Hibernate**

**ORM**

Mapping Java classes to database tables is experienced through the configuration of an XML file or by using Java Annotations. When using an XML file, Hibernate can precipitate skeleton source code for the persistence classes. This is immaterial when annotations are used. Hibernate can handle the XML file or the annotations to uphold the database schema.

Facilities to exhibit one-to-many and many-to-many relationships between classes are provided. In summation to managing associations between objects, Hibernate can also reach reflexive associations where an object has a one-to-many relationship with other instances of its own type.

Hibernate supports the mapping of custom figure types. This makes the consequently scenarios possible: Overriding the regular SQL type that Hibernate chooses when mapping a column to a property. Mapping Java Enum to columns probably they were stable properties. Mapping a single property to numerous columns. Objects in a front-end application inherit OOP principles, meanwhile objects in the back-end follow database normalization principles, resulting in different representation requirements.

This issue is called "object-relational impedance mismatch". Mapping is a manner of resolving the impedance mismatch problem. Mapping tells the ORM tool which java class object an application is needed to be store in which table of database.

**Features:-**

Hibernate ORM enables developers to more plainly write applications whose data outlives the application process. As an Object/Relational Mapping (ORM) framework, Hibernate is worried with data persistence as it applies to relational databases (via JDBC).

**JPA Provider**

In summation to its own "native" API, Hibernate is besides an implementation of the Java Persistence API (JPA) specification. As such, it can be plainly used in any environment supporting JPA including Java SE applications, Java EE application servers, Enterprise OSGi containers, etc.

**Idiomatic persistence**

Hibernate enables you to transpire persistent classes following spontaneous Object-oriented idioms including inheritance, polymorphism, association, composition, and the Java collections framework. Hibernate requires no interfaces or base of operation classes for persistent classes and enables any class or data structure to be persistent.

**High Performance**

Hibernate consistently offers superior performance over straight JDBC code, both in terms of developer productivity and runtime performance.

**Scalability**

Hibernate scales well in any environment: Use it to drive your in-house Intranet that serves hundreds of users or for mission-critical applications that serve hundreds of thousands.

**Reliable**

Hibernate is well known for its excellent stability and quality, proven by the acceptance and use by tens of thousands of Java developers.

**Extensibility**

Hibernate is highly configurable and extensible.

**C. ADO.NET Entity Framework**

[1]As a part of .NET Framework, an open source object-relational mapping for ADO.NET is Entity Framework (EF). The development of data-oriented software applications are supported by Entity Framework, a set of technologies in ADO.NET. Architects and developers of data-oriented applications have substantially struggled by the whole of the require to achieve two very different objectives. For solving the business problems, developers must model the entities, relationships, and logic, and they must also work with the data engines used to store and retrieve the data. Each with its own protocols, the data may span multiple storage systems; even applications that function with a single storage system must insure the requirements of the storage system against the requirements of writing efficient and maintainable application code.

The Entity Framework enables developers to work with data in the form of domain-specific objects and properties, such as customers and client addresses, without having to consider themselves with the underlying database tables and columns where this data is stored. With the Entity Framework, developers can work at a higher level of abstraction when they deal by the whole of data, and can create and raise data-oriented applications with less code than in traditional applications. Because the Entity Framework is a element of the .NET Framework, Entity Framework applications can run on any computer on which the .NET Framework (starting with version 3.5 SP1) is setup.

The Entity data model (EDM) specifies the conceptual model (CSDL) of the data, using a modeling technique that is itself called Entity Data Model, an extended version of the Entity-Relationship model.[8] The data model primarily describes the Entities and the Associations they apportion in. The EDM schema (step by step diagram) is expressed in the Schema Definition Language (SDL), which is an application of XML. In addition, the mapping (MSL) of the fundamentals of the conceptual schema (CSDL) to the storage schema (SSDL) must also be specified. The mapping specification is further expressed in XML.

Visual Studio besides provides Entity Designer, for visual formation of the EDM and the mapping specification. The product of the tool is the XML file (*.edmx) specifying the schema and the mapping. Edmx document contains EF metadata artifacts (CSDL/MSL/SSDL content). These 3 files (csdl, msl, ssdl) can besides be created or edited by hand.

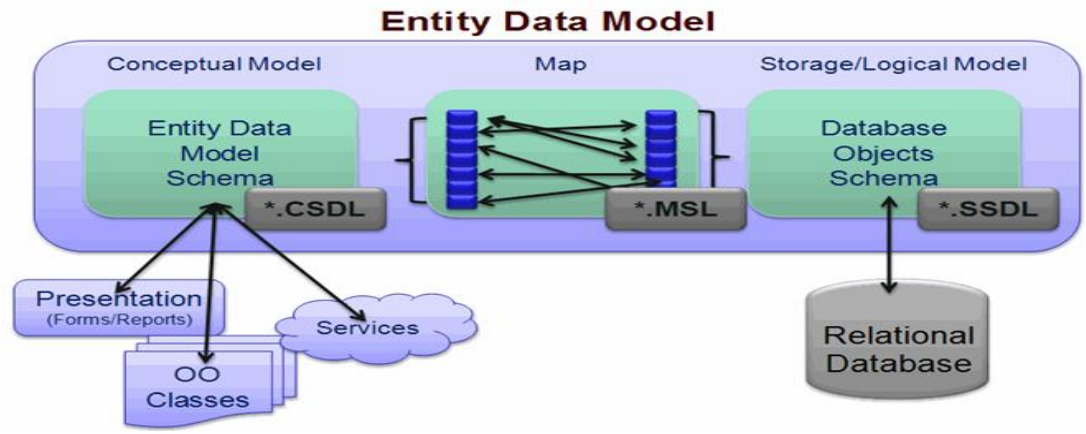



**Figure 8: Entity Data Model in ADO .NET Framework**

CSDL represents application data as a collection of entities and relationships in a conceptual model, and it is an implementation of Entity Data Model. A storage model is expressed in SSDL and represents the schema (step by step representation) of the data store. The mapping mid two models is expressed in MSL.

The CSDL, SSDL, and MSL contents for an application can be automatically generated by the ADO.NET Entity Data Model Tools. The Entity Data Model Wizard generates model and mapping impression as well as data classes from an existing database. The Entity Data Model Designer (Entity Designer) can formerly be handled to graphically modify model and mapping information. Or, you can graphically create a conceptual model with the Entity Designer, and then use the Generate Database Wizard to automatically precipitate the storage model, mappings, data classes, and a database that supports the conceptual model. In as a substitute scenario, you can handle the Update Model Wizard to apprise model and mapping information when the underlying database changes.

**4. Object Relational Database Mismatch**

[5][6]Today, for the transformation of the object model to relational model technique known as object relational mapping is used. Object relational mapping is an activity that is performed during the design of an information system and performs mapping of the objects from the object model into entities (tables) in relational database. The object relational impedance mismatch occurs during this activity. The basic differences between object-oriented technologies and storing of data in the form of relationships include:-

**Differences in data types:-**

The main difference lies in the discrepancy between the data types of the both models. Object-oriented programming languages allow the declaration of data types and attributes of objects whose size is usually limited only by the size of allocated memory. In relational databases, however, it is required to declare the data types with the exact determination of their size or length. For example: in Java language, a variable to store the string is declared as a STRING, in a relational database system MySQL, data type to store strings are declared as VARCHAR.

**Differences in structure and integrity:-**

In the object model, an object may be composed of other objects. It is possible to define different types of relationships between objects, such as generalization, aggregation, etc. In the relational model only one type of relationship - a relationship between entities - can be defined. In the terms of data integrity, well-defined relationships ensure that data consistency within the model is automatically controlled (or checked). In the object model the links between objects are more in the logical layer, so the consistency of data must be provided with supporting mechanisms.

**Differences in the ways how data are handled:-**

The relational model has a relatively small but well-defined set of primitive operations for querying data and for data manipulation. The object model provides general query and manipulates with data using low-level operations to enable access to objects on its physical level. Object-oriented programming languages tend to work with lists of objects or hash tables, so the set of operations for handling operations with objects is different compared to sets of operations used by the relational model.

**Differences in transactions:-**

In the relational model a transaction is defined as an indivisible set of activities. In the object model there is no analogy for set of operation that can manifest with the attributes of isolation and atomicity.

## Conclusion

In this document, the main differences between object and relational model were identified. Current approach to object relational mapping was stated .The proposed solution is appropriate for simple object models without complex compositions or many-to-many relationships. So in the future work we will focus on extending the proposed approach for complex object models. In the case of complex object model there are several options for creating a relational database schema. It is therefore appropriate to provide all options for creating a relational database schema based on a specific object model and to propose the most suitable solution.

## References

[1] Bogdan Walek, JiĜí Bartoš, and Cyril Klimeš, Fuzzy Tool for Reasoning in Object Relational Mapping of Information System, IEEE, 2012.

[2] Aleksandra Boniewicz , Piotr Wi niewskí And Krzyszt of Stencel, On Redundant Data for Faster Recursive Querying Via ORM Systems, Proceedings of the 2013 Federated Conference on Computer Science and Information Systems pp. 1451–1458 , IEEE 2013.

[3] Sergey Melnik , Atul Adya and Philip A. Bernstein, Compiling Mappings to Bridge Applications and Databases, SIGMOD'07, June 11–14, 2007, Beijing, China.

[4] CHRISTIAN BAUER AND GAVIN KING, Java Persistence with Hibernate.

[5] Wolfgang Keller, Mapping Objects to Tables A Pattern Language.

[6] Aneta Szumowska, Aleksandra Boniewicz, Marta Burzańska, Piotr Wiśniewski, Hibernate the Recursive Queries - Defining the Recursive Queries Using Hibernate ORM.