International Journal of Engineering, Economics and Management
ISSN: 2319-7927, Volume 2, Issue 1

Reengineering of Relational Databases to Object Oriented Database: Schema Mapping & Data Migration

R.S.Mawale **Prof.A.V.Deorankar** **Prof. R.L. Pardhi**

***Abstract* --- Many information systems use relational database systems for efficient sharing, storage, and retrieval of large quantities of data. On the other hand, object-oriented programming has been gaining wide acceptance in the programming community as a paradigm for developing complex applications that are easy to extend and maintain. This paper discusses development of an integrated environment which maps a relational schema to an object-oriented schema without the need to modify the existing relational schema and providing a platform for migrating data from relational database to object oriented database. The performance analysis of this approach is done by providing NEONS grid relational schema to the proposed system.**

***Keywords* ---- Schema Mapping, RDBMS, OODBMS, Reengineering**

I. INTRODUCTION

Relational database management systems (RDBMS) provide a variety of tools and services for data management. There are many tools that interface with RDBMSs to enable end-users to carry out reporting, querying, and other data analysis activities easily. On the other hand, object-oriented programming has been gaining wide acceptance in the programming community as a paradigm for developing complex applications that are easy to extend and maintain. Developers typically implement object-oriented applications using object-oriented programming languages such as C# and Java. These applications use a schema made up of object classes and relationships between those object classes. Each object has a set of attributes. The value of an attribute could be another object itself, thus giving rise to complex objects. The main problem arises when the data corresponding to such objects are persistent in a relational database. The problem is due to incompatibility between relations and objects. At first, a solution for this problem may seem to be to use an ODBMS instead of RDBMS. If an ODBMS is managing the persistent data, the objects do not lose their structure after the application stores them in the object database. Moving to an ODBMS might mean throwing away all of the old ("legacy") data and applications. Most users of databases will not accept such a solution. They wish to be able to run their existing applications on existing databases and have access to the same data from object-oriented programs, too. Therefore, we need special techniques to convert the data that is residing in a relational database to a format that is suitable for access and manipulation by object-oriented applications. The Performance analysis will be carried out in order to determine consistency and completeness of the approach

II. LITERATURE REVIEW

The problem of migrating data is present in almost every application development process, such as data warehousing and application integration. The algorithm implements process of migrating data involves firstly the mapping between the structures of the source and target databases and secondly the migration of the data from the source to the target [3].

The problem is how to effectively migrate existing RDBs, as a source, into OODB, as targets, and what is the best way to enrich and maintain RDBs' semantics and constraints in order to meet the characteristics of the targets? Existing work does not appear to provide a complete solution for more than one target database. We tackle this question by proposing a solution for migrating an RDB into the three targets based on available standards [4].

One general approach to migrate to object technology is to divide the process into two phases, where the first one transforms the relational into an object-oriented schema and the second one migrates the data into the object-oriented database system discussed in [5].

The goal of re-engineering is to mechanically reuse past development efforts particular relational databases (RDBs).in order to reduce maintenance expense and improve software flexibility [4]. The object behavior of eight Java programs including four real-world Java object oriented database management systems and a counterpart of four real-world Java programs are analyzed in [6]. A technique for transferring query optimization techniques, developed for relational databases, into object Databases. This technique for ODMG database schemas defined in ODL and object queries expressed in OQL. The object schema is represented using a logical representation (Datalog) [7].

III. ANALYSIS OF PROBLEM

Moving to an ODBMS might mean throwing away all of the old ("legacy") data and applications. Most users of databases will not accept such a solution. They wish to be able to run their existing databases and have access to the same data from object-oriented program.

Thus we need to implement a system that builds an understanding of a given conventional database by taking these characteristics as input and produces the corresponding object-oriented database as output.

Finally, we handle the migration of data from the conventional database to the constructed object-oriented database.

The Primary objectives of proposed work are as follows.

1. Study the automatability of the relational-to-OO schema mapping process.
2. Define an interactive process for mapping an existing relational schema to an object-oriented schema.
3. Develop an interactive system to validate proposed work

IV. SYSTEM ARCHITECTURE

The architecture contains two major components needed for fulfilling our aim. The first component deals with mapping the relational schema to an object-oriented schema. The second component deals with the mapping between the relational data and objects.

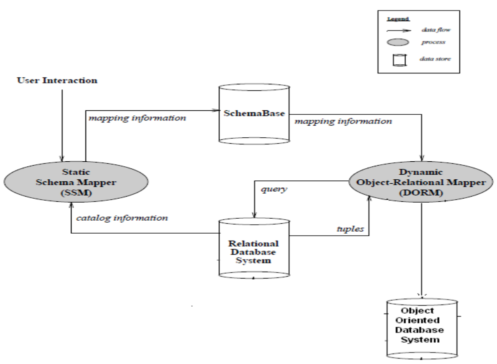

Figure 1. System Architecture

Schema Mapping

The static schema mapping process is a two-phase process. In the first phase, the relational schema is adjusted and transformed into another virtual relational schema that has some specific properties. In the second phase, object-oriented structures are extracted from the virtual relational schema.

Phase I: Adjusting the Relational Schema

There are four specific aspects that are addressed during phase one. They are as follows:

Step 1. Eliminate 2NF relations and replace them with new 3NF virtual relations.

Step 2. Create virtual subclass relations for widow super class relations

Step 3. Create virtual superclass relations for orphan subclass relations.

Step 4. Eliminate multi-valued attributes and replace them with new 3NF virtual relations.

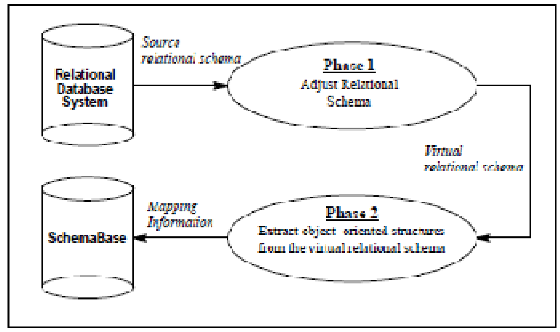

Figure 2.Two Phases of Static Schema Mapping

Phase II: Generation of the Object Schema

At the end of phase one of the schema mapping process, the relational schema has been adjusted to a form in which schema mapping rules can be applied uniformly.

Step1. Identifying Object Classes:

Those relations that correspond to object- classes must be identified.

Step2. Identifying Relationship:

There are three types of relationships that can be represented in an object model. They are associations, generalizations specializations, and aggregations. Identifying each of these constructs constitutes a step in the mapping process.

Identifying Associations. Since the object model allows associations to be modeled as classes, we must either establish a simple association between two object classes or identify relationships where the associations are modeled as classes.

Identifying Inheritance. Inheritance structures capture the generalization and specialization relationships between object classes that have been identified so far.

Identifying Aggregation. The aggregation relationship models the composition of one object with other objects. such complex object must be identified. The difference between aggregation and association is that the former involves existence dependence of the sub-object on the whole object. For example, a door object, which is a part-of" of a car object, cannot exist if the car object does not exist. On the other hand, the enrollment of a student in a course is an association rather than an aggregation because the student and course objects can exist independent of each other.

Step 3. Establishing Cardinalities : Establishing the cardinalities of associations is important in order to facilitate the implementation of the object schema in a given programming language (e.g., C#). The different possible cardinalities are one-one, one-many, and many-many.

V. PERFORMANCE ANALYSIS

Once system is developed then we will got the result.

From the result, we can analyse that the requirements gets fulfilled or not. the implementation of propose approach following questions can be asked ;

1. Elimination of 2NF relations are carried out or not ?
2. Elimination of widow relations are carried out or not ?
3. Elimination of orphan relations are carried out or not ?
4. Elimination of BLOB attributes are carried out or not ?
5. Identification of classes are carried out or not ?
6. Identification of associations are carried out or not ?
7. Identification of inheritance are carried out or not ?
8. Identification of aggregation are carried out or not ?
9. Data Mapping is carried out or not ?

For testing the computational models with our experimental model, we are going to use Grid Data Relational Schema. The Naval Environmental Oceanographic Now-casting System (NEONS) is a geophysical relational database designed by the Naval Oceanographic and Atmospheric Research Laboratory. NEONS is a large database containing a variety of data such as grid model output, satellite images, and so on. The portion of the database to be considered here corresponds to grid data. Grid data contains atmospheric and oceanographic numerical model out- puts, user-defined parameters, and gridded climatology data. The collection of data values corresponding to the grid points in a rectangular grid comprises a grid_eld. Figure 3 contains the relational schema from the NEONS database corresponding to grid data. A few attributes have been left out of the schema for the sake of brevity. The relational schema does not contain any widow or orphan relations. One notice- able relation in the schema is otisP1, which includes an attribute named bit stream of BLOB data type. The bit stream is the attribute that contains all the measurements of a given parameter at all the specified grid points. Since the number of grid points usually exceeds 10,000, using a bit stream is an efficient way of improving the throughput of the database queries. The object schema that results from the application of the mapping algorithms is given in Figure 4. The major adjustment the source relational schema goes through is the creation of an aggregation relationship corresponding to the BLOB attribute of the otisP1 relation. A new virtual relation and a corresponding object class named bit stream are created based on the description given by the user. Based on the available description of the data, the object schema is intuitive with respect
to the domain semantics.

as grid	grid id, model type, vrsn name, geom id, min base etm,max base etm PK = fgrid idg FK = fgeom id (grid geom)g, fmodel type (grid model)
grid model	model id, model type, fcst dsc, remark, pack null PK = fmodel idg, fmodel typeg FK = None
grid geom	grid geom geom id, geom name, geom dsc, bgn etm, stor dsc
grid reg geom	PK = fgeom idg, fgeom nameg FK = None geom id, prjn name, longitude, latitude, geom parm 1, geom parm 2, geom parm 3
grid spct geom	PK = fgeom idg FK = fgeom id (grid geom)g, fprjn name (projection) grid spct geom geom id, max lat wav num, max lon wav num 1
projection	prjn name, row int dsc, col int dsc, geom parm dsc 1, geom parm dsc 2, geom parm dsc 3, remark PK = fprjn nameg
grid parm	parm id, parm name, parm dsc, bit cnt, unit name
grid lvl	PK ,fparm idg, fparm nameg FK = None grid lvl lvl id, lvl type, lvl name 1, lvl name 2, unit name, lvl dsc PK = flvl idg, flvl typeg FK = None
otis p1	grid id, parm id, lvl id, lvl 1, lvl 2, base etm, fcst per, bitstream PK = fgrid id, parm id, base etm, lvl id, lvl 1, lvl 2g FK = flvl id (grid lvl)g, fparm id (grid parm)g, fgrid id (as grid)g

Figure 3. NEONS Grid Relational Data Schema

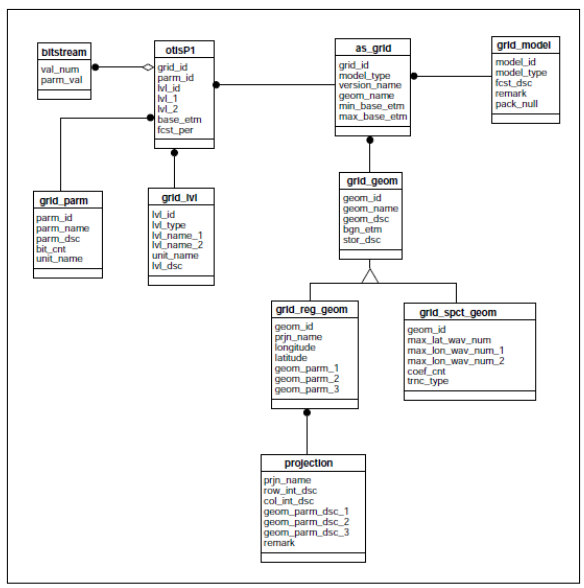

Figure 4. Object Schema for NEONS Database

My proposed model is an experimental model which I have described in section 3.1. Here I will give the different relational database schemas as input to the system. Since all the special features that the schema mapping process can handle may not be present in a single relational schema, Four different perspectives of a relational database schema were considered to illustrate the overall capabilities of the mapping. One real-world geophysical relational database and three different relational schemas will be used for analysis.

Operations	Experimental Model
Elimination of 2NF relations	√
Elimination of widow relations	√
Elimination of orphan relations	√
Elimination of BLOB attributes	√
Identifying classes	√
Identifying associations	√
Identifying inheritance	√
Identifying aggregation	√
Data Mapping	√

Figure 5. Analysis of Proposed Approach using NEONS Schema

CONCLUSION

We will implement the architecture having two phases in the data mapping process; one mapping procedure specifies the data mapping between the original relational schema and the adjusted relational schema. The second mapping procedure specifies the data mapping between the adjusted relational schema and the object schema. The data mapping procedures have been specified using relational algebra for each new virtual relation that is created during phase one of the static schema mapping. Again we have shown the performance analysis using NEONS database schema. The proposed system is able to eliminate 2NF relations, widow relation, orphan relation, BLOB attributes and can identify class, association, inheritance, aggregation. Also the proposed system is able to migrate data from relational database to object oriented database.

REFERENCES

[1] Ramanathan, C. Providing object-oriented access to a relational database. In Proceedings of the 32nd ACM annual southeast conference held in Tuscaloosa, Alabama, March, 1994.

[2] Andersson M. Extracting an entity-relationship schema from a relational database through reverse engineering. In Proceedings of the 13th international conference on entity relationship approach held in Manchester, UK, December, 1991

[3] Elton Manoku, Guido Bakema, A fact approach on Data Migration published in Academy of Communication and Information Technology HAN University of Applied Sciences, The Netherlands In June 2006

[4] Abdelsalam Maatuk, Akhtar Ali, and Nick Rossiter , A Framework for Relational Database Migration published in School of Computing, Engineering & Information Sciences Northumbria University, Newcastle upon Tyne, UK in 2001

[5] Andreas Behm, Andreas Geppert, Klaus R. Dittrich2 ,On the migration of relational schemas and data to object-oriented database published in Proceedings of the 5th International Conference on Re-Technologies in Information Systems, Klagenfurt, Austria, December 1997

[6] William J Premerlani, Michael R Blaha, An Approach for Reverse Engineering of Relational Databases published in May 1994 Communications of the ACM

[7] Chia-Tien Dan Lo, Morris Chang, Ophir Frieder and David Grossman, The Object Behavior of Java Object-Oriented Database Management Systems published in the Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC.02) IEEE 2002

[8] John Grant, Jarek Gryz, Jack Minker, and Louiqa Raschid, Logic-Based Query Optimization for Object Databases published in IEEE Transactions on knowledge and data engineering, vol. 12, no. 4, July/August 2000.