International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:6, No:5, 2012

A methodology for Data Migration between Different Database Management Systems

Bogdan Walek, Cyril Klimes

***Abstract*—**In present days the area of data migration is very topical. Current tools for data migration in the area of relational database have several disadvantages that are presented in this paper. We propose a methodology for data migration of the database tables and their data between various types of relational database systems (RDBMS). The proposed methodology contains an expert system. The expert system contains a knowledge base that is composed of IF-THEN rules and based on the input data suggests appropriate data types of columns of database tables. The proposed tool, which contains an expert system, also includes the possibility of optimizing the data types in the target RDBMS database tables based on processed data of the source RDBMS database tables. The proposed expert system is shown on data migration of selected database of the source RDBMS to the target RDBMS.

***Keywords*—**expert system, fuzzy, data migration, database, relational database, data type, relational database management system

I. Introduction

IN present days the area of data migration is very topical. Various information systems and their databases are often transferred between different types of hardware or software equipment. This activity requires the proper coordination and management, because it is necessary transfer data and their structure to the new system correctly [5]. In the area of databases we had to correctly migrate logical structure of the database and their data stored in database tables along with appropriate data types. We also need to remember that various types of relational database systems (further in text referred as RDBMS), which contain and manage databases, have different characterics and properties. All RDBMS are based on the relational model [1], [2], but the specific properties and activities provided in the database can be implemented differently. Such as naming and support of data types, SQL commands for creating and editing database tables, as well as the specific features of the RDBMS, which may not be supported in another RDBMS.

Now we would like to define the process that is used to migrate the database data.

The general process of data migration, which is used for database migration and data between different types of RDBMS is called *ETL* (Extract, transform and load) and consists of the following 3 steps:

Bogdan Walek is with the Department of Informatics and Computers, University of Ostrava, 30. dubna 22, 701 03 Ostrava, Czech Republic (e-mail:bogdan.walek@osu.cz).

Cyril Klimes is with the Department of Informatics and Computers, University of Ostrava, 30. dubna 22, 701 03 Ostrava, Czech Republic (e-mail:cyril.klimes@osu.cz).

1. Extracting data from the source database.
2. Transforming data into usable form for migration to the target database.
3. Migration of data to the target database.

The ETL process uses the terms source and target databases that are refined here:

Source database – database of source RDBMS, from which data are migrated

Target database – database of target RDBMS, into which data are migrated

There are a lot of reasons for data migration between various types of RDBMS, we would like to state here only few of them:

1. *Upgrading to new version of the same RDBMS* – in the case of regular upgrading of sotware equipment in the company, including new version of RDBMS, we need to migrate data to new RDBMS
2. *The existing RDBMS is insufficient* – in the case of a large increase of stored data, insufficient speed or capacity of RDBMS
3. *Change of company policy* – in the case of changing security or another type of policy in company, we need to upgrade to better, more sophisticated RDBMS
4. *Economical problems in company* – this can lead to uninstalling commercial and expensive RDBMS and installing open-source and cheaper RDBMS instead
5. *A lot of various applications and RDBMS* – we need to unify various database and various RDBMS to one specific RDBMS to ensure better efficiency and consistency of data

As mentioned above, one of the main differences between various types of RDBMS is the naming and definition of data types that are supported. For illustration there is a definition of several data types according to SQL3 standard [3] and their equivalents in the selected RDBMS in the table below:

TABLE I
Differences in Data Types

SQL3	MySQL	Oracle	MSSQL
INTEGER	TINYINT SMALLINT INTEGER MEDIUMINT BIGINT	SMALLINT INTEGER	TINYINT SMALLINT INTEGER BIGINT
CHARACTER	CHAR	CHAR NCHAR	CHAR NCHAR
CHARACTER VARYING	VARCHAR TINYTEXT	VARCHAR2 NVARCHAR2	VARCHAR NVARCHAR
CHARACTER LARGE OBJECT	MEDIUM-TEXT TEXT LONGTEXT BLOB	LONG BLOB CLOB NCLOB LONGBLOB	TEXT NTEXT LONG-VARCHAR

From the table we can conclude that the naming of data types and their support in various RDBMS are very different.

Currently there are also a lot of tools for data migration between databases of various types of RDBMS. We provide here only a few of them, because it is not possible to quantify and describe all of these tools:

1. SwisSQL Data Migration Tool
2. ESF Database Migration Toolkit
3. DatabaseBridge
4. Cross-Database Studio
5. Data Management Center

Now allow us to summarize the most important features of these tools:

a) Tools can enable migration of database tables and their data
b) Some tools have problems with migration of foreign keys that are a part of the relations between database tables
c) Some tools enable migration of procedures, functions and triggers stored in a database
d) Some tools are capable of comparing selected tables of source and target database and compare them
e) Tools can´t be modified or extended by user
f) There is no possibility to change parameters or features of target database tables and their attributes (such as data types, etc.)

II. Problem Formulation

From the summary of the tools for data migration between various types of RDBMS we can conclude that tools enable migration of database tables and their data. The disadvantage of some tools is the inability to migrate foreign keys that are a part of relations between database tables. Another disadvantage is the impossibility of modifying or extending of existing tools. It is also impossible to change parameters and features of target database tables and their attibutes. For example database specialist can´t change data types of the target database tables attributes.

So the main goal of this article is to propose a methodology for data migration between of RDBMS, which should help to reduce or eliminate disadvantages of the existing tools. The methodology will also propose an expert system for proposal of the suitable data types for attributes of target database tables. Based on the methodology we are also proposing a tool for migrating database tables and their data that can be modified and extended by a database specialist.

III. Problem Solution

The main aim of this paper is to propose the methodology for migrating database tables and their data between various types of RDBMS.

Here are the main steps of the methodology that are visually displayed in the following figure:

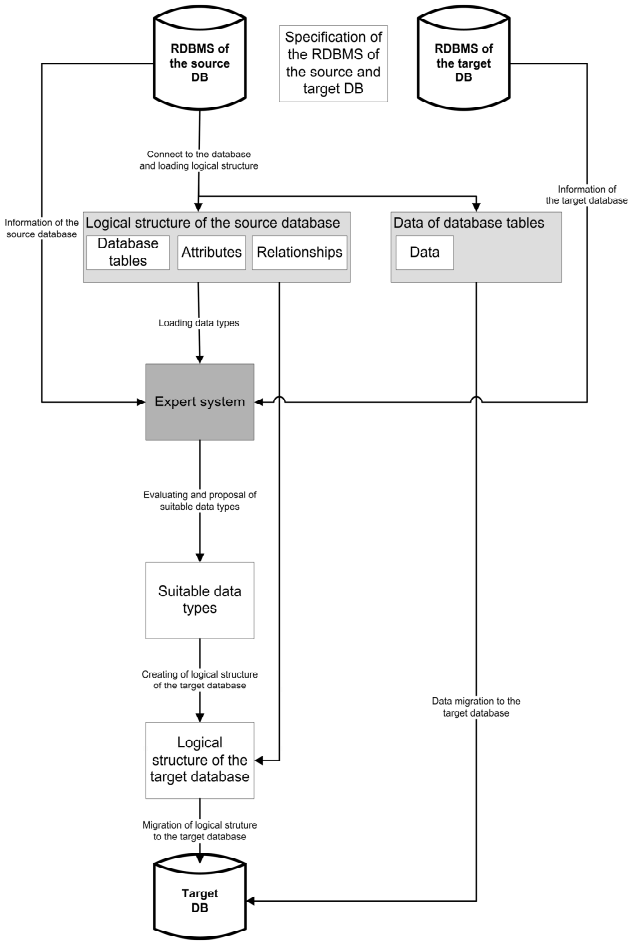

Fig. 1 A methodology for data migration

The main steps of the methodology will be described in the next part of this paper.

A. Specification of the source and target RDBMS

At the beginning it is necessary to specify the RDBMS of the source database. With the type of RDBMS we need to specify the version of the selected RDBMS, because various versions of the same type of RDBMS can be different. The best way to obtain information about type and version of RDBMS is to connect with source database and load the metadata of the database that contains the required information.

We also need to specify the RDBMS of the target database. With the type of RDBMS we need to specify the version of this RDBMS. This information will be used in the next steps of the methodology.

There are examples of several types of RDBMS along with the list of their versions:

- MySQL – versions: 3.2.3, 4.0, 4.1, 5.0
- Oracle – versions: 8i, 9i, 10g, 11g
- PostgreSQL – versions: 8.3, 8.4, 9.0, 9.1
- MSSQL – versions: 2005, 2008, 2008 R2, 2012

B. Loading the logical structure of the source database

In the next step it is necessary to load the logical structure

of the source database. Logical structure of the database contains the database tables, their attributes and the relations between database tables. We devide the process of loading the logical structure of the source database into two parts:

a) Loading the database tables
b) Finding and identifying all relationships between database tables

First part contains loading these properties for each database table:

1. Database table name
2. A list of attributes
3. Data types of attributes
4. Size of the data types

Visually the first part is shown in the following figure:

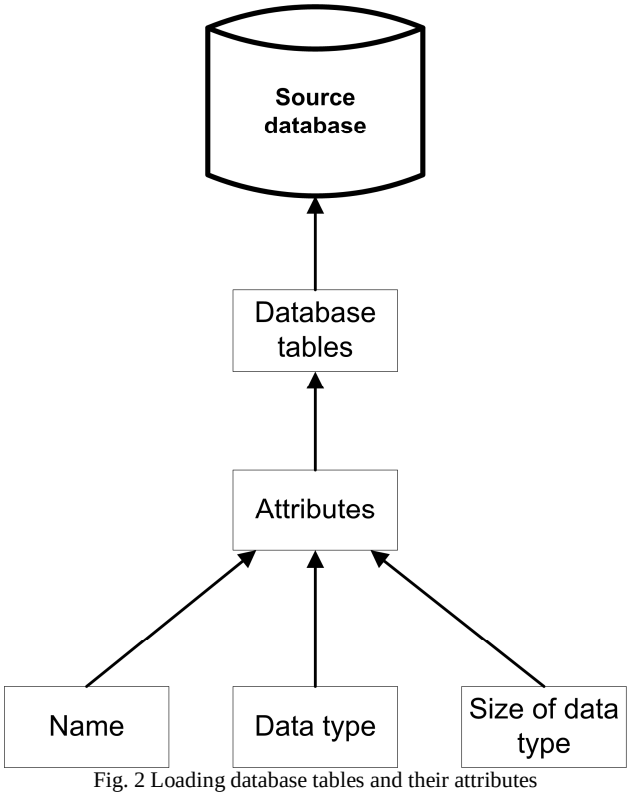

Fig. 2 Loading database tables and their attributes

Loading database tables and their attributes can be implemented in different ways, one of them is to use JDBC driver and load metadata of the database using predefined methods for loading table names and a list of their attributes.

In the second part we need to find and identify of all relationships between database tables in the source database. For each relation we had to define this set:

R = (PK_TABLE, PK_COL, FK_TABLE, FK_COL), where:

PK_TABLE – name of referenced database table

PK_COLUMN – column that contains the primary key of the referenced tables

FK_TABLE – name of the referencing database table

FK_COLUMN – column that contains the foreign key of the referencing database table

The process of finding relationships in database is shown in the following figure:

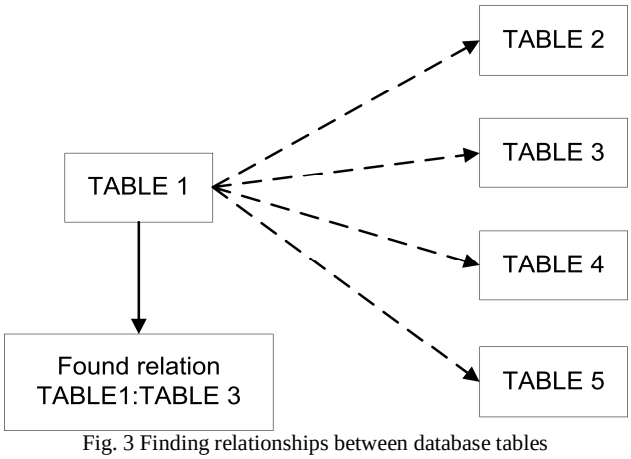

Fig. 3 Finding relationships between database tables

In this figure is shown the process of finding relationships only for TABLE I. We had to repeat this process for combination of all pairs of database tables and then the result of this process will be a set of all the relations in the source database.

C. Proposal of suitable data types by expert system

In this step we propose the expert system which will propose a set of suitable data types of the attributes in the target database tables. The proposal of suitable data types will be based on the type of the RDBMS of the source database, the type of the RDBMS of the target database and the specific attribute found in the source database table.

The expert system contains knowledge base that is composed of the suitable IF-THEN rules.

Input data for expert system are:

1. The type of RDBMS of the source database (SOURCE_DB)
2. The name of the attribute in the source database table (SOURCE_DATA_TYPE)
3. The type of RDBMS of the target database (TARGET_DB)

Output of the expert system is:

1. Evaluated attribute in the target database table

An example of the IF-THEN rules for finding suitable data types for data type VARCHAR, when RDBMS of the source database is MySQL and RDBMS of the target database is ORACLE is shown below:

```
IF SOURCE_DB IS MySQL AND
SOURCE_DATA_TYPE IS VARCHAR AND
TARGET_DB IS ORACLE THEN
SUITABILITY OF VARCHAR2 IS VERY BIG

IF SOURCE_DB IS MySQL AND
SOURCE_DATA_TYPE IS VARCHAR AND
TARGET_DB IS ORACLE THEN
SUITABILITY OF NVARCHAR2 IS VERY BIG
```

```
IF SOURCE_DB IS MySQL AND
SOURCE_DATA_TYPE IS VARCHAR AND
TARGET_DB IS ORACLE THEN
SUITABILITY OF BLOB IS VERY SMALL
```

Evaluation of suitable data types for source data type VARCHAR, if source RDBMS is MySQL and target RDBMS is Oracle is presented in the following table:

TABLE II
EVALUATING SUITABILITY OF THE PROPOSED DATA TYPES

Data type name	Suitability of the data type
VARCHAR2	0.99
NVARCHAR2	0.99
CHAR	0.99
LONG	0.01
BLOB	0.01

Evaluation of data types results in a number from <0,1>, that indicates the degree of suitability of the proposed data type. 0 indicates that the data type is completely unsuitable, 1 indicates that the data type is very suitable.

From the table above we can conclude that very suitable data types are VARCHAR2, NVARCHAR2 and CHAR, but data types LONG and BLOB are very unsuitable, because they are designed for storing large strings.

D. Selection of suitable data types

In this step the user (database specialist) can select one of the evaluated suitable data types for each attribute in the target database table. Or he can use default data type that was predefined by the expert system.

E. Generating SQL dump file for creating the target database in the RDBMS

At the end we propose generating SQL dump file that contains SQL command for creating database tables, their attributes and relationships in the RDBMS of the target database tables.

SQL dump file contains these parts:

a) SQL CREATE commands for creating database tables and their attributes
b) SQL ALTER commands for creating primary and foreign keys in the database tables
c) SQL INSERT commands for inserting data to the database tables

After that, the generated SQL dump file will be imported to the RDBMS of the target database and the target database will be created.

IV. RESULTS

The methodology will be verified by means of the tool that is created for data migration between various types of RDBMS. RDBMS of the source database is MySQL. RDBMS of the target database is Oracle. Selected database is the university database.

Logical structure of this database is shown in the following figure:

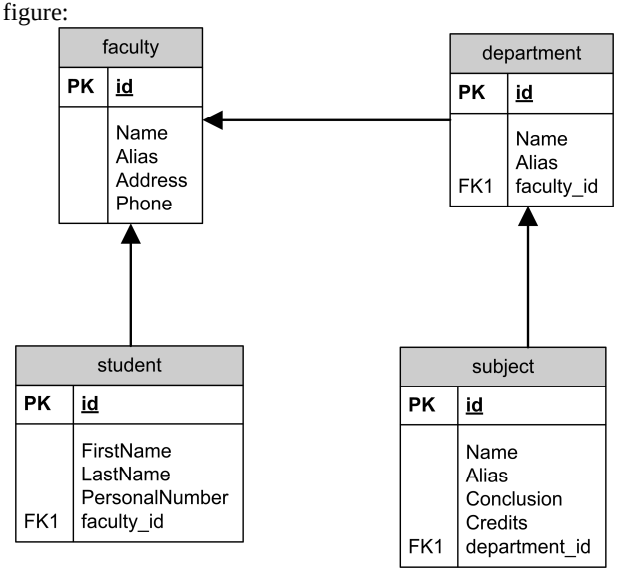

Fig. 4 Selected database logical structure

A. Specification of the source and target RDBMS

As the source RDBMS was selected RDBMS MySQL 5.0. As the target RDBMS was selected RDBMS Oracle 11g.

B. Loading the logical structure of the source database

Logical structure of the database was loaded by methods predefined in java.sql and JDBC driver.

```
department
  id - int(10)
  faculty_id - int(10)
  Name - varchar(100)
  Alias - varchar(10)
faculty
  id - int(10)
  Name - varchar(100)
  Alias - varchar(10)
  Address - varchar(50)
  Phone - varchar(20)
student
  id - int(10)
  faculty_id - int(10)
  FirstName - varchar(50)
  LastName - varchar(50)
  PersonalNumber - varchar(10)
subject
  id - int(10)
  department_id - int(10)
  Name - varchar(50)
  Alias - varchar(10)
  Conclusion - varchar(10)
  Credits - int(10)
```

Fig. 5 Loading the logical structure of the source database

Then we found these relationships between the database tables:

```
department
id
subject
department_id

faculty
id
department
faculty_id

faculty
id
student
faculty_id
```

Fig. 5 Finding relationships between the database tables

C. Proposal of suitable data types by expert system

In this step we propose data types by expert system that is implemented in LFLC tool [4].

Firstly we define linguistic variables and fuzzy sets for each value of specified linguistic variable. The fuzzy set for linguistic variable SOURCE_DATA_TYPE represented by triangular functions is shown in the following figure:

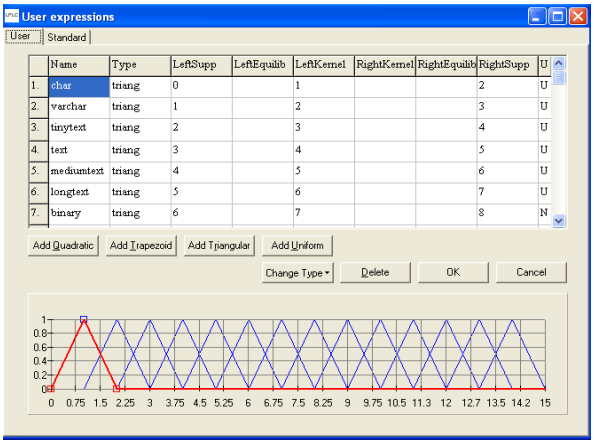

Fig. 6 Identify possible data type in LFLC tool

Then we create a knowledge base for evaluating data type – as for example the knowledge base of data type VARCHAR2:

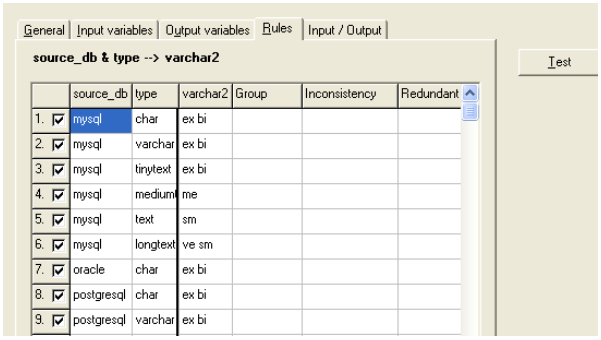

Fig. 7a A knowledge base in LFLC tool

Evaluation of data types for source data type VARCHAR is shown below:

```
VARCHAR2 : 0,99
NVARCHAR2 : 0,99
CHAR : 0,99
LONG : 0,01
BLOB : 0,01
```

Fig. 7b Evaluation of data types

D.Selection of suitable data types

In this step the user can select one of the proposed data types – we select the first data type VARCHAR2.

E. Generating SQL dump file for creating the target database in the RDBMS

In this step the tool generates SQL dump file that can be included in Oracle, part of this SQL dump is shown below:

```
CREATE TABLE department(
id NUMBER NOT NULL,
faculty_id NUMBER NOT NULL,
Name VARCHAR2(100) NULL,
Alias VARCHAR2(10) NULL
);
```

Fig. 8 SQL dump for RDBMS Oracle

Then we can import SQL dump into the database of RDBMS Oracle:

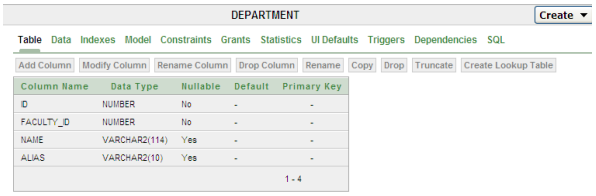

Fig. 9 The department table in the target database

Data of the department table are shown below:

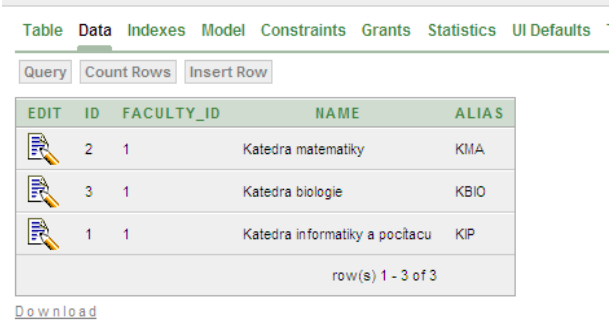

Fig. 10 Data of the department table

V. Conclusion

In this paper we analyzed the current state in data migration between various types of the RDBMS. Then we evaluate features and disadvantages of current tools for data migration. Next, we propose the methodology for data migration between various types of RDBMS that contains the expert system for

decision support during the process of data migration. The expert system proposes suitable data types for attributes of target database tables and can be modifiied or extended, if it´s necessary. In conclusion we propose a tool that is based on the proposed methodology. Results are shown on the data migration of the selected database.

ACKNOWLEDGMENT

The presented topic is also a part of the internal grant SGS10/PřF/2012, called Fuzzy modeling tools for analysis and design of information systems, at the Department of Infomatics and Computers, University of Ostrava.

REFERENCES

[1] P. Atzeni, V. De Antonellis, *Relational Database Theory*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, 1993, ch. 1.
[2] L. Jan Harrington, *Relational database design and implementation*. Elsevier Inc., Burlington, 2009, ch. 4,5.
[3] *ISO, ANSI: Framework (SQL/Framework)*. International Organization for Standartization, 2003.
[4] H. Habiballa, V. Novák, A. Dvořák, V. Pavliska, "Using software package LFLC 2000", 2nd International Conference Aplimat 2003, Bratislava, 2003, pp. 355-358.
[5] J. Morris, *Practical data migration*. The British Computer Society, Chippenham, 2009, ch.1.