World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:6, No:9, 2012

Data Migration between Document-Oriented and Relational Databases

Bogdan Walek, Cyril Klimes

***Abstract*—**Current tools for data migration between document-oriented and relational databases have several disadvantages. We propose a new approach for data migration between document-oriented and relational databases. During data migration the relational schema of the target (relational database) is automatically created from collection of XML documents. Proposed approach is verified on data migration between document-oriented database IBM Lotus/Notes Domino and relational database implemented in relational database management system (RDBMS) MySQL.

***Keywords*—**data migration, database, document-oriented database, XML, relational schema

I. INTRODUCTION

IN present days there are many types of databases and huge amount of data stored in databases. Sometimes it is necessary to migrate data from one type of database to another type of database. Or we need to create new database implemented in another type of database and move data from old database to new database. In these cases the process of data migration will be initiated. Process of data migration consists of three steps and is called *ETL* (Extract, transform and load) [8]:

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 from which data are migrated
Target database – database into which data are migrated

In this article two types of databases will be used for data migration. Source database is a document-oriented database. Target database is one of the most used type of database: relational database [1], [2].

A. Document-oriented database

A document-oriented database is designed for document-oriented applications like document management system (DMS) or system for storing important documents or contracts of a company. The basic entity of a document-oriented database is a document. The document contains fields and each field has a value. Relationships are represented by two fields depending on documents which have the same value.

A big advantage of this database type is easy usage and programming, so untrained business users can create applications and design their own databases.

There are a lot of implementations of document-oriented database: IBM Lotus/Notes Domino, eXist, Apstrata, MongoDB, etc.

B. Relational database

A relational database is a database based on a relational model. The basic entity of a relational database is a relation. A relation is usually described as a table which is organized into rows and columns. Rows are understood as records and columns represent attributes or properties of the table [1], [2].

This paper focuses of a new approach for data migration between document-oriented databases and relational databases using XML documents for storing data of document-oriented database. During data migration the relational schema of the target (relational database) will be automatically created from collection of XML documents generated from document-oriented database. Proposed approach will be verified on data migration between document-oriented database IBM Lotus/Notes Domino [3] and relational database implemented in relational database management system (RDBMS) MySQL.

II. PROBLEM FORMULATION

Currently, there are a few tools for data migration between document-oriented and relational databases. Here are few of them which are used for data migration between document-oriented database IBMLotus/Notes Domino and various types of relational database management systems.

A. IBM Tivoli Directory Integrator

Software from IBM can transform, move and migrate data between heterogeneous directories, databases, files, collaborative systems and applications, including migration from the IBM Lotus Domino database to a relational database. The tool helps enhance robust integration of important data with a possibility of transformation to other file formats and synchronization between two or more systems, but IBM Lotus Notes has to be the source or target system.

Migration from the IBM Lotus Notes database to a relational database management system (RDBMS), such as Microsoft SQL Server, Oracle or MySQL can be provided by connection via JDBC or ODBC driver. After successful connection to RDBMS, we can choose source data from the Lotus database and migrate it to a table of RDBMS. IBM Tivoli Directory Integrator is offered as a free tool for the IBM Lotus Notes platform [4].

B. Notes2DB

An open source tool for data migration from the Lotus Notes database to a relational database, for example: IBM DB2 7, Oracle 9I, Microsoft SQL Server 2000, MySQL 3.23.

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).

In Notes2DB we can easily configure the ODBC data source of the RDBMS to which we want to export the data. Then we can enter the target RDBMS table name to which we want to export data of Lotus Notes documents. Next, we can choose fields from Lotus Notes to export and map them to their equivalent columns in the RDBMS table. Finally, we can migrate data from Lotus Notes documents to the table of the RDBMS. Notes2DB supports data migration from the Lotus Notes database to a relational database with creating one-to-many relationships of a relational database and is offered as an open source tool [5].

C. CSVExporter for Lotus Notes

CSVExporter is a simple tool for data export from the Lotus Notes database to CSV files, which can be imported to a relational database. CSVExporter for Lotus Notes doesn´t support data migration to a relational database, but data can be exported to a CSV file and then we can import it to a relational database. Firstly, we select Lotus Notes database and specific view or folder of this database. Then we can choose fields from the Lotus Notes view of folder to export. Finally, we can migrate data to a CSV file. CSVExporter for Lotus Notes is offered as trial software [6].

D. DetachIt for Lotus Notes

DetachIt is another simple tool for data export from the Lotus Notes database to CSV files, which can be imported to a relational database, text files, HTML, PDF, etc. DetachIt for Lotus Notes is similar to CSVExporter for Lotus Notes, but it allows exporting attachments and rich text fields of Lotus Notes documents. Working with this tool is very similar to working with CSVExporter for Lotus Notes tool. After selection of the Lotus Notes database, view or folder and fields to export we can choose the type of the target file and migrate data to the selected file type. DetachIt for Lotus Notes is offered as trial software in two editions – Standard and Premium. Premium Edition allows exporting multiple databases in batch mode [7]. From this simple review of tools that allow data migration from the Lotus Notes database to a relational database, we can conclude that the most robust tool is IBM Tivoli Directory. 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 main disadvantage of all of the tools is that they are not able to automatically create database schema of the target (relational) database based on migrated data from document-oriented database.

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.

III. PROBLEM SOLUTION

The main aim of the paper is to propose a new approach for data migration between document-oriented databases and relational databases using XML documents for storing data of document-oriented database.

During data migration the relational schema of the target (relational database) will be automatically created from collection of XML documents.

Proposed approach is shown in the following figure:

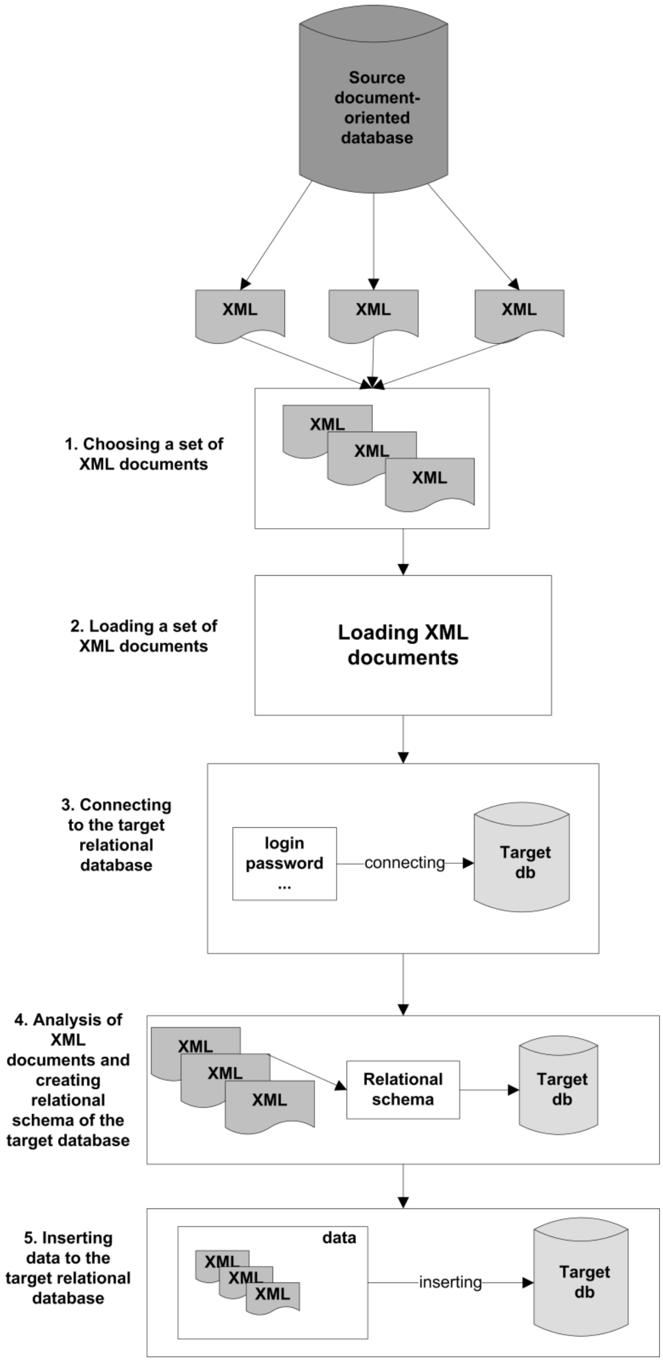

Source document-oriented database
XML
XML
XML
1. Choosing a set of XML documents
XML
XML
XML
2. Loading a set of XML documents
Loading XML documents
3. Connecting to the target relational database
login
password
...
connecting
Target db
4. Analysis of XML documents and creating relational schema of the target database
XML
XML
XML
Relational schema
Target db
5. Inserting data to the target relational database
data
XML
XML
XML
inserting
Target db

Fig. 1 Proposed approach for data migration

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

A. Choosing a set of XML documents

At the beginning it is necessary to choose appropriate set of XML documents. XML documents are generated from the source document-oriented database and store data of document-oriented database which will be migrated. For example, if we need to migrate information about students and their subjects, we choose XML documents Student.xml and Subject.xml. Each XML document stores information about one or more documents that contain information about the same type of document of the selected document-oriented database.

We also need to identify dependencies and links between XML documents. For this activity is suitable to create conceptual model of the selected set of XML documents. Each XML document represents one entity of the resulting conceptual model. Attributes of entity are represented by field of the specified type of document which is stored in XML document. Relationships between entities are represented by couples of fields of documents which have the same value and are linked together (for example field *Faculty* in document type *Department* and field *Name* in document type *Faculty*).

Then we need to specify superior entities which are not depended on other entities. These entities will be migrated in the first phase and from them the main database tables of relational schema of the target relational database will be created. Then the subordinate entities will be specified and they will represent database tables which are depended on main tables.

For example XML document *department-faculty.xml* represents subordinate entity. It stores information about departments and each document type *Department* has a link to document type *Faculty* which represents superior entity.

XML document department-faculty is shown in the following figure:

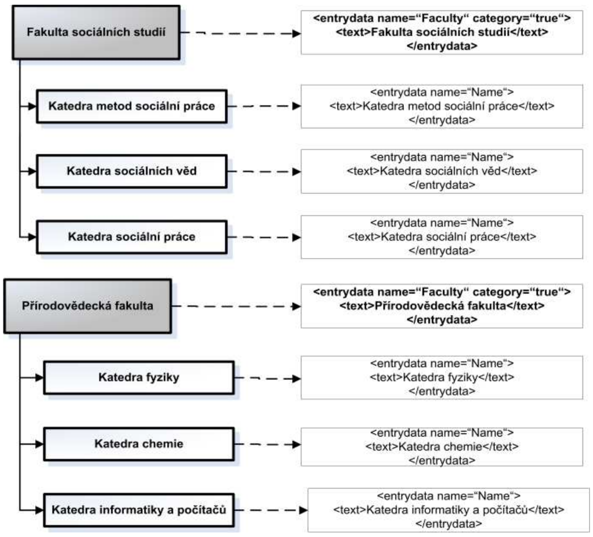

Fakulta sociálních studií
<entrydata name="Faculty" category="true">
<text>Fakulta sociálních studií</text>
</entrydata>
Katedra metod sociální práce
<entrydata name="Name">
<text>Katedra metod sociální práce</text>
</entrydata>
Katedra sociálních věd
<entrydata name="Name">
<text>Katedra sociálních věd</text>
</entrydata>
Katedra sociální práce
<entrydata name="Name">
<text>Katedra sociální práce</text>
</entrydata>
Přírodovědecká fakulta
<entrydata name="Faculty" category="true">
<text>Přírodovědecká fakulta</text>
</entrydata>
Katedra fyziky
<entrydata name="Name">
<text>Katedra fyziky</text>
</entrydata>
Katedra chemie
<entrydata name="Name">
<text>Katedra chemie</text>
</entrydata>
Katedra informatiky a počítačů
<entrydata name="Name">
<text>Katedra informatiky a počítačů</text>
</entrydata>

Fig. 2 XML document department-faculty.xml

Finally, we need to specify all relationships between documents in document-oriented database and create map file which represents all relationships between subordinate and superior entities. We need to identify couple of document fields which represents relationships between subordinate document and superior document of source database. In document-oriented database are not relations (which are known from relational database) so relationships between documents are represented by the couple of fields which have the same value.

Finally, the map file will be constructed from quaternions {T, TC, PT, PTC}:

1. T – name of type of document which represents subordinate entity
2. TC – name of field which contains value of field which represents the relationship
3. PT – name of type of document which represents superior entity
4. PTC – name of field which contain value of field which represents the relationship

Process of creating map file is shown in the following figure:

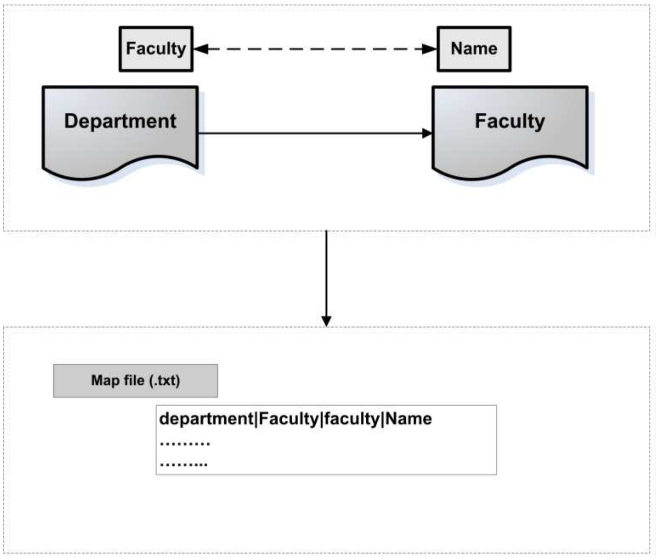


Fig. 3 Example of map file

B. Loading a set of XML documents

In this step we need to load all selected XML documents which will be migrated to target relational database.

Firstly, it is necessary to specify appropriate names of the XML documents which represent superior entities. The best option is to specify the name of XML document based on type of document which is stored in the XML document. For example for document type *Faculty* the name of XML document will be specified as *Faculty.xml*. These XML files will be migrated at first and the name of main database tables in target relational database will be determined from the names of XML documents.

Next, we need to specify appropriate names of XML documents which represent subordinate entities. These names will consist of the name of subordinate entity and the name of superior entity. For example for XML document which stores information about document type *Department* and document type *Faculty*, the name will be specified as *Department-Faculty.xml*. Based on this information the dependent database table with appropriate foreign key will be created.

C. Connecting to the target relational database

In this step it is necessary to connect to the target relational database. Connection can be provided by using JDBC driver and appropriate login data. The target relational database is empty, so the relational schema of the database will be created later.

Process of connecting to the target relational database is shown in the following figure:

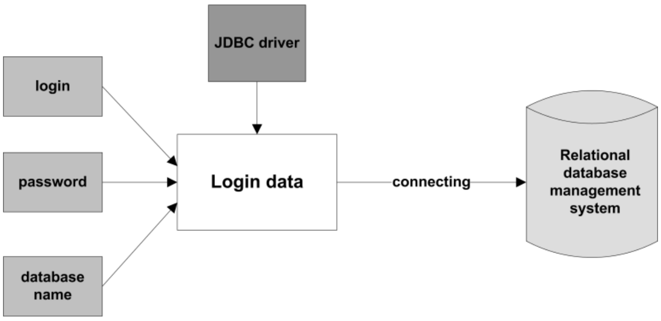


Fig. 4 Connecting to the target relational database

D. Analysis of XML documents and creating relational schema of the target database

In this step is necessary to analyze XML documents and based of information and of the XML documents to automatically create relational schema of the target relational database.

In the first part we need to analyze XML documents that represent superior entities:

1. Loading elements viewentry of XML document.
2. For all elements viewentry – loading all child elements entrydata and other elements with detection of data types.
3. Based on information from step 2 the objects that represent fields of documents from source document-oriented database are created.
4. Based on information from step 3 the database table in the target database is created. SQL INSERT queries for inserting data to the created database table are prepared.

Then XML documents which represent subordinate entities are loaded:

1. Loading elements viewentry of XML document.
2. For all elements viewentry – loading all child elements entrydata and other elements with detection of data types.
3. Based on information from step 2 the objects that represent fields of documents from source document-oriented database are created.
4. Creating object which contains information about foreign key of database table based on data stored in specified map file.
5. Based on information from step 3 and 4 the dependent database table with foreign key in the target database is created. SQL INSERT queries for inserting data to the created database table are prepared.

The output of this phase is created relational schema of the target database.

E. Inserting data to the target relational database

In the last step the data of the XML documents are inserted to the created relational database. Inserting data consists of two steps:

1. Inserting data to the main database tables
2. Inserting data to the dependent database tables

Finally, the process of migration is finished and results of data migration are shown to the user.

IV. RESULTS

The proposed approach will be verified on migrating data of the document-oriented database which contains information about faculties, departments, students and subjects.

A. Choosing a set of XML documents

Information about types of documents and dependencies in the source document-oriented database are shown in the following figure:

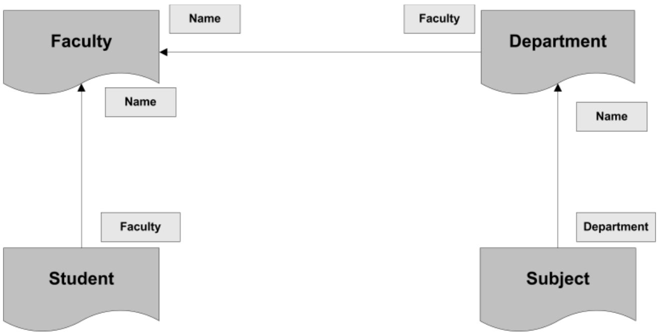


Fig. 5 Source document-oriented database

Here are XML documents generated from the source database:

1. Faculty.xml
2. Department-Faculty.xml
3. Subject-Department.xml
4. Student-Faculty.xml

Mapping file *mapping.txt* contains these rows:

```
Department|Faculty|faculty|Name
Subject|Department|department|Name
Student|Faculty|faculty|Name
```

B. Loading a set of XML documents

In this step XML documents will be loaded.

C. Connecting to the target relational database

Next, the database connection to the target relational database will be provided. Target relational database will be implemented in RDBMS MySQL. The name of target relational database is *university*.

D. Analysis of XML documents and creating relational schema of the target database

In this step XML documents will be analyzed and relational schema will be created.

Relational schema created from XML documents is shown in the following figure:

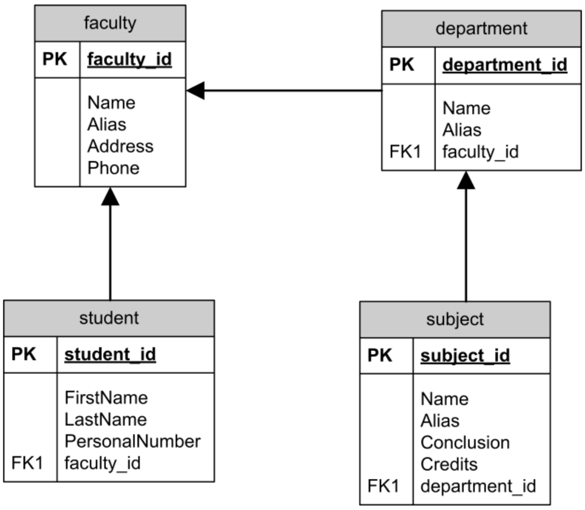
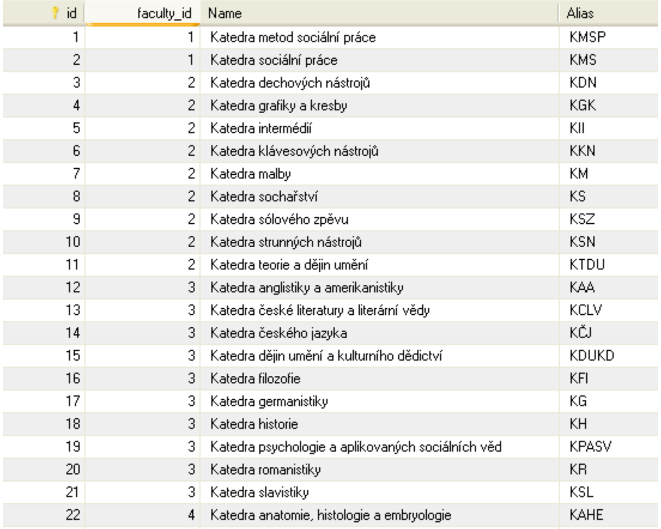


Fig. 6 Created relational schema of the target relational database

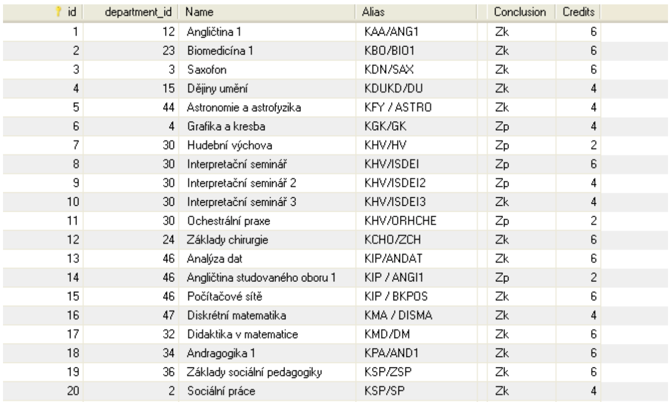
E. Inserting data to the target relational database

Finally, data to the created relational database are inserted. Examples of inserted data are shown in the following figures:



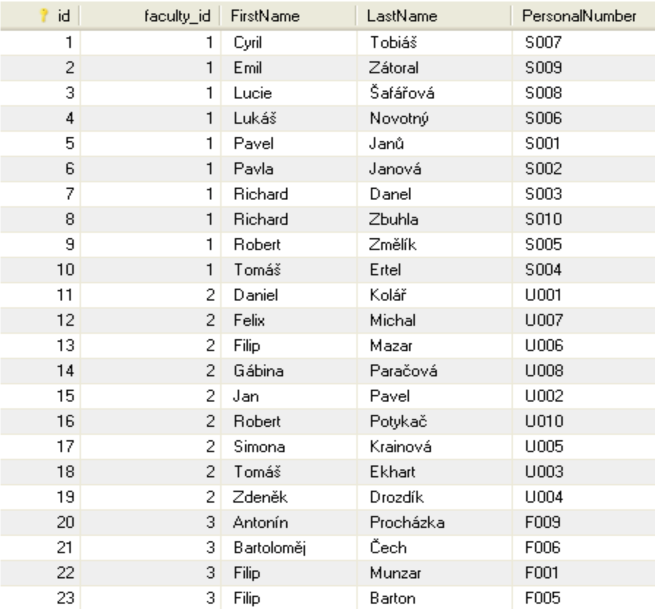
id	faculty_id	Name	Alias
1	1	Katedra metod sociální práce	KMSP
2	1	Katedra sociální práce	KMS
3	2	Katedra dechových nástrojů	KDN
4	2	Katedra grafiky a kresby	KGK
5	2	Katedra intermédií	KII
6	2	Katedra klávesových nástrojů	KKN
7	2	Katedra malby	KM
8	2	Katedra sochařství	KS
9	2	Katedra sólového zpěvu	KSZ
10	2	Katedra strunných nástrojů	KSN
11	2	Katedra teorie a dějin umění	KTDU
12	3	Katedra anglistiky a amerikanistiky	KAA
13	3	Katedra české literatury a literární vědy	KCLV
14	3	Katedra českého jazyka	KČJ
15	3	Katedra dějin umění a kulturního dědictví	KDUKD
16	3	Katedra filozofie	KFI
17	3	Katedra germanistiky	KG
18	3	Katedra historie	KH
19	3	Katedra psychologie a aplikovaných sociálních věd	KPASV
20	3	Katedra romanistiky	KR
21	3	Katedra slavistiky	KSL
22	4	Katedra anatomie, histologie a embryologie	KAHE

Fig. 7 Data of the database table department



id	department_id	Name	Alias	Conclusion	Credits
1	12	Angličtina 1	KAA/ANG1	Zk	6
2	23	Biomedicína 1	KBO/BIO1	Zk	6
3	3	Saxofon	KDN/SAX	Zk	6
4	15	Dějiny umění	KDUKD/DU	Zk	4
5	44	Astronomie a astrofyzika	KFY / ASTRO	Zk	4
6	4	Grafika a kresba	KGK/GK	Zp	4
7	30	Hudební výchova	KHV/HV	Zp	2
8	30	Interpretační seminář	KHV/ISDEI	Zp	6
9	30	Interpretační seminář 2	KHV/ISDEI2	Zp	4
10	30	Interpretační seminář 3	KHV/ISDEI3	Zk	4
11	30	Ochestrální praxe	KHV/ORHCHE	Zp	2
12	24	Základy chirurgie	KCHO/ZCH	Zk	6
13	46	Analýza dat	KIP/ANDAT	Zk	6
14	46	Angličtina studovaného oboru 1	KIP / ANGI1	Zp	2
15	46	Počítačové sítě	KIP / BKPOS	Zk	6
16	47	Diskrétní matematika	KMA / DISMA	Zk	4
17	32	Didaktika v matematice	KMD/DM	Zk	6
18	34	Andragogika 1	KPA/AND1	Zk	6
19	36	Základy sociální pedagogiky	KSP/ZSP	Zk	6
20	2	Sociální práce	KSP/SP	Zk	4

Fig. 8 Data of the database table subject



id	faculty_id	FirstName	LastName	PersonalNumber
1	1	Cyril	Tobiáš	S007
2	1	Emil	Zátoral	S009
3	1	Lucie	Šafářová	S008
4	1	Lukáš	Novotný	S006
5	1	Pavel	Janů	S001
6	1	Pavla	Janová	S002
7	1	Richard	Danel	S003
8	1	Richard	Zbuhla	S010
9	1	Robert	Změlík	S005
10	1	Tomáš	Ertel	S004
11	2	Daniel	Kolář	U001
12	2	Felix	Michal	U007
13	2	Filip	Mazar	U006
14	2	Gábina	Paračová	U008
15	2	Jan	Pavel	U002
16	2	Robert	Potykač	U010
17	2	Simona	Krainová	U005
18	2	Tomáš	Ekhart	U003
19	2	Zdeněk	Drozdík	U004
20	3	Antonín	Procházka	F009
21	3	Bartoloměj	Čech	F006
22	3	Filip	Munzar	F001
23	3	Filip	Barton	F005

Fig. 9 Data of the database table student

V. CONCLUSION

In this paper we analyzed the current state in data migration between document-oriented database and relational database. Then we evaluate features and disadvantages of current tools for data migration. Next, we propose new approach for data migration between document-oriented databases and relational databases using XML documents for storing data of document-oriented database. During data migration the relational schema of the target (relational database) was automatically created from collection of XML documents. Proposed approach was verified on data migration between selected document-oriented database IBM Lotus/Notes Domino and relational database implemented in relational RDBMS MySQL.

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 Informatics 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] *IBM Lotus Notes*. IBM, 2011. http://www-01.ibm.com/software/lotus/products/notes/.

[4] IBM – Software to synchonize across multiple repositiories – Tivoli Directory Integrator. IBM, 2011. http://www-01.ibm.com/software/tivoli/products/directory-integrator/**.**

[5] Notes2DB: Migrate Data from IBM Lotus Notes Domino to a RDBMS. 2011. http://my.advisor.com/doc/12332.

[6] *DetachIt Attachment Extraction and field Export Tool for Lotus Notes*. Kim Beros Consulting, 2011. http://www.lotus-notes-export.com/DetachIt.asp?MoreInfo=1.

[7] *Lotus Notes to SQL. Convert Lotus Notes to SQL. Lotus Notes Export to SQL*. Kim Beros Consulting 2011. http://www.lotus-notes-export.com/Lotus-Notes-To-SQL.asp.

[8] J. Morris, *Practical data migration.* The British Computer Society, Chippenham, 2009, ch.1.