DMD Project

Design and implementation of a Publication Management System

Bulat Mukhutdinov*
Innopolis University
1 Universitetskaya st
Innopolis, Russia
bulatmm@yandex.ru

Timur Shakirov†
Innopolis University
1 Universitetskaya st
Innopolis, Russia
shakirovtr23@gmail.com

Nikolay Yushkevich‡
Innopolis University
1 Universitetskaya st
Innopolis, Russia
n.yushkevich@hotmail.com

ABSTRACT

In this paper we describe the development of the system that is aimed at the management of the publication records. This project is divided into three phases. The first one is designing and implementation of the relational model using an existing DBMS. The second phase is the development of a web based user interface that provides the interaction between user and the database designed in the first phase. The third phase is the development of a new DBMS and the replacement with DBMS used in the first phase.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Algorithms

Keywords

DBMS, dblp

1. INTRODUCTION

The project's aim is to design and implement a publication records management system. In this project we use **dblp [1]** - Digital Bibliography & Library Project. It is a computer science bibliography. This source provides an open bibliographic information on major computer science journals and proceedings. It currently contains more than 2.6 million publications with over 1.4 million authors. Moreover, dblp indexes about 25 000 journal volumes, more than 24 000 conferences or workshops, and more than 17 000 monographs.

PostgreSQL [2] is used as a DBMS in order to implement relational model. It is an object-relational database system that can be run on the majority of the operating systems, such as Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), and Windows. Moreover, it has a native programming interface for Java.

DbSchema [3] is used to design the relational model of the project's database and to create database's ER model.

Java [4] programming language is used for the development of the project.

2. PHASE 1. DESIGN AND IMPLEMENTATION OF THE RELATIONAL MODEL

The aim of this phase is to use the database of different publications for real life problems. As it was mentioned before, dblp bibliography is used to crawl publications data. Therefore, the relational model is based on the structure of dblp. There are 16 entities in it: Article, Inproceedings, Proceedings, Book, Incollection, Phdthesis, Mastersthesis, www, Article_author, Inproceedings_author, Proceedings_author, Book_author, Incollection_author, Phdthesis_author, Mastersthesis_author, www_author. ER diagram is shown on Figure 1. You can see the full size image at https://github.com/BulatMukhutdinov/DMD_Java4Life/blob/master/Report/ER.jpg.

All publications are connected with their authors as 'one to many'. For example, one Article can have many Article_authors, one Book can have many Book_authors and so on. The ER model is based on dblp.dtd and dblp.xml files, that are crawled in real time from http://dblp.uni-trier.de.

The Relations are:

1. Article(key, mdate, editor, title, pages, year, journal, volume, number, month, url, ee, cdrom, cite, publisher, note, crossref)
2. Article_author(key, author)
3. Book(key, mdate, editor, title, pages, year, volume, month, url, ee, cdrom, cite, publisher, note, isbn, series, school, chapter)
4. Book_author(key, author)

*Responsible for creation SQL functions, database connection, logger, code optimization

†Responsible for creation Web-client, project structure, code optimization, works with the documentation

‡Responsible for creation XML parser, database structure, works on the data

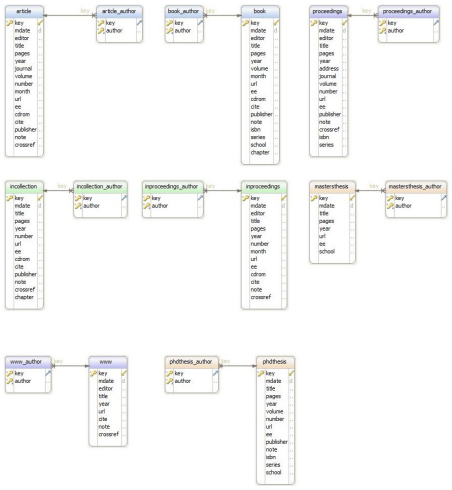

Figure 1: ER diagram

5. Proceedings(key, mdate, editor, title, pages, year, address, journal, volume, number, url, ee, publisher, note, crossref, isbn, series)
6. Proceedings_author(key, author)
7. Incollection(key, mdate, title, pages, year, number, url, ee, cdrom, cite, publisher, note, crossref, chapter)
8. Incollection_author(key, author)
9. Inproceedings(key, mdate, editor, title, pages, year, number, month, url, ee, cdrom, cite, note, crossref)
10. Inproceedings_author(key, author)
11. Masterthesis(key, mdate, title, pages, year, url, ee, school)
12. Masterthesis_author(key, author)
13. Phdthesis(key, mdate, title, pages, year, volume, number, url, ee, publisher, note, isbn, series, school)
14. Phdthesis_author(key, author)
15. www(key, mdate, editor, title, year, url, cite, note, crossref)
16. www_author(key, author)

2.1 Fillig the database

After the database structure is created, our XML parser starts to work. This XML parse is based on the standart javax.xml library. Basically, it takes the data from different XML tags and writes them into CSV file, in order to copy this data into our database using COPY FROM command. The COPY FROM command operates much faster than a normal INSERT command because the data is read as a single transaction directly to the target table. On the other hand, it is a very strict format, and the entire COPY procedure will fail if just one line is malformed. So our CSV file is accurately created using dblp.xml, all the data has ";" delimiters.

2.2 Functions

There are 4 different functions, that were created for Phase 1.

- Function that shows all articles that are related to each other by their authors.
- Function that shows all articles that are related to each other by their journal.
- Function that sorts the 'proceedings' table by 'title' column and returns first 'a_count' tuple (a_count is an argument that is equal to 100 by default).
- Function that sorts the 'proceedings' table by the count of 'crossref' of table 'article' and returns rows where this count is more or equal to the 'minimal_count_of_crossrefs' argument (minimal_count_of_crossrefs = 10 by default).

2.3 Other technical details

The project structure is shown on the Figure 2.

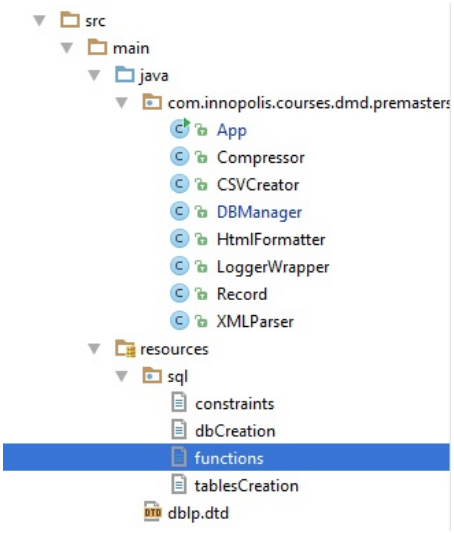

Figure 2: Project structure

The project is divided into 8 classes.

- App is a class with a 'main()' method.
- Compressor is responsible for zipping and unzipping files.
- CSVCreator is responsible for creating CSV files.
- DBManager is responsible for the database connection, creating database structure, filling database with data and execution of SQL commands.
- HTMLFormatter is responsible for creating HTML document with logs.
- LoggerWrapper is responsible for making proper logs.
- Record is used in XMLParser for proper data processing.
- XMLParser is responsible for parsing dblp.xml file in order to get data from it.

3. REFERENCES

[1] dblp: computer science bibliography. http://dblp.uni-trier.de/db/. Accessed: 2015-09-11.

[2] Postgresql. http://www.postgresql.org/. Accessed: 2015-09-12.

[3] Dbschema - universal database designer and sql tool. http://www.dbschema.com/. Accessed: 2015-09-12.

[4] Java programming language. https://www.oracle.com/java/index.html. Accessed: 2015-09-12.