# CSE-A1200 Databases

Project Report

Database for Helsinki area public library system

https://mail.aalto.fi/owa/14.3.174.1/themes/resources/clear1x1.gifhttps://mail.aalto.fi/owa/14.3.174.1/themes/resources/clear1x1.gif

# ****Kamyar Hasanzadeh****

# ****Anna Gorodetskaya****

# [anna.gorodetskaya@aalto.fi](mailto:anna.gorodetskaya@aalto.fi) ****(336172)****

## **Spring 2014**

## Part 1

### Tasks

• Draw an E/R-diagram based on the information in the following sections.

• Convert the E/R-diagram to the relational data model. Present the schemas of the relations and

underline the attributes which form the key for each relation.

• Provide answers to the following questions: What are the functional dependencies of the

database? Are there any form of redundancy or other anomalies in the database structure? Is

your database in Boyce-Codd Normal Form? If it is not, please use the decomposition algorithm.

Fee

Returns

Loans

Items

Reservationsa

Library

Customers

### 1. E/R Data model

Library

Customers

Reservationsa

Fee

Items

Loans

Returns

### 2. Relational model

Our database consists of several entities: libraries, customers and items.

Each library has a unique name with address and phone number that is recorded in library relational table.

Information about all the items from all the libraries is collected in one relational table. Each item has its unique ID and information about its name, type, author, year, and home library. The items with the same name and type have the same group ID. Additionally, it is possible to know if the item is available in exact moment and if it is reservable or not (consequently available and reservable). Some items can be loaned but not reserved and in opposite. For each item also the duration of loan is recorded.

All customers have their unique customer ID, information about their names, home address and phone number.

Our relational model saves the information about following operations in library system: loaning, reserving, collecting fees and returning.

For each loan with unique ID the information about customer and borrowed item is recorded together with the start and the due date of the loan and information about returning status of the item (returned or not). In case if the due date is over than today`s date and the returned status is ‘not returned’, the customer is started to be charged for overdue loan and information about fee is recorded into Fee relational table.

Each reservation is also recorded with unique reservation ID, customer ID and group ID: because for customer the title of the item is more important rather than the exact item itself. So the customer can reserve any available in exact moment item from any library and point out the most convenient library for picking out the item where the item will be delivered from home library. Additionally customer gets the position in the queue of people who wants to reserve the item with the same name. The date of reservation is also fixed and the fee for reservation service is recorded for reserving customer in Fee relational table.

In case of returning, the information about loan ID, returning date and returning library is recorded.

The relational table Fee contains information about the transaction ID, reason of fee (whether it is fee for loan overdue or for reservation), amount of money and status of fee (was it paid or not). Until the customer hasn`t paid for overdue loan and hasn`t returned the book to the library, the sum of fee is growing every day by a fixed amount.

Our relational model consists of the following relational tables (the keys for each table are underlined):

Library (name, address, tel)

Customers (CID, name, address, tel)

Items (IID, GID, available, item\_type, name, author, item\_year, loanduration, reservable, homelib)

Loans (loanID, IID, CID, Sdate, Ddate, returned)

Reserve (RID, CID, GID, DelivLib, resdate, quepos)

Fee (reason, transID, amount, status)

Returning (loanID, retdate, retlib)

### 3. Functional dependencies

The existing functional dependences for our database are following:

Library:

name address, tel

Customers:

CID name, address, tel

Items:

IID GID, available, item\_type, name, author, item\_year, loanduration, reservable, homelib

Loans:

loanID IID, CID, Sdate, Ddate, returned

Reserve:

RID CID, GID, DelivLib, resdate, quepos

Fee:

reason, transID amount, status

Returning:

loanID retdate, retlib

### 4. Redundancy and other anomalies, following BCNF.

According to (Garcia­Molina et al., 2008)[[1]](#footnote-1): “A Relation R is in BCNF if and only if: whenever there is a nontrivial FD A1 A2 ... An → B1 B2 ... Bm for R, it is the case that {A1 A2 ... An } is a superkey for R”.

 Our relational model with its functional dependences is following the above mentioned condition that means that our relation is already in Boyce-Codd Normal Form (BCNF). This fact is eliminating the presence of redundancy or other anomalies in our case.

## Part 2

### Tasks

• Define the relation schema in SQL. The schema must contain same information as the project

part 1 (library database). Write the CREATE statements to the document. Use data types that are

reasonable. Justify your solutions.

• Check the keys and other constraints in your database. Verify that the primary keys and foreign keys are well defined.

• Which kind of SQL queries might be typical for the library information system? Create reasonable indexing which supports the purposeful use of the database. In addition, add at least

one useful view definition to your database.

• Create some use cases for the library information system. Write a description of the use case,

explain which information needs to be selected/updated/inserted and write the SQL queries that

are needed to fulfill the use case. The number of SQL queries written in the document must be

at least 15. Try to find at least one use case where you need to use a little bit more complicated

SQL query.

• Run the SQL statements (creating the tables, inserting example data to them and the SQL

queries) in SQLite environment. Sqlite3 is available in Aalto IT Linux computers (at least in

most of them) and you can also download it to your own computer, see www.sqlite.com. Note

that there are some limitations in SQL statements implemented by SQLite.

Justify all your solutions carefully!

Note that you are not asked to write a user interface for the database. You can create the database and perform all your queries by just writing the SQL statements.

### Solution

1. Garcia­Molina, H.; Ullman, J. D.; Widom, J. 2008. Database  Systems – The Complete Book. Second Edition, New Jersey, NY: Pearson ­ Prentice Hall. [↑](#footnote-ref-1)