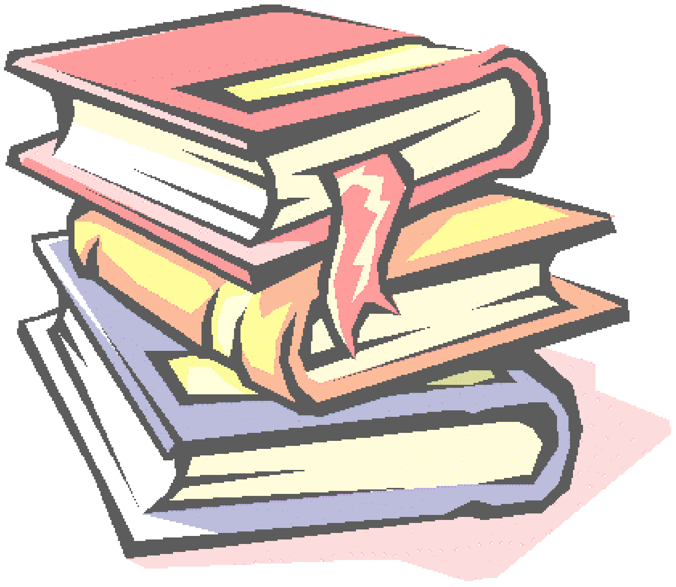
THE BOOKSTORE

Project Databases IK
Bachelor INFORMATIEKUNDE
Year ’04-‘05

Faculty FNWI
University of Amsterdam

29th of June 2005



Instructors:	Dr. Irina Neaga / Dr. Hamideh Afsarmanesh (coordinator)
Assistants:	G. de Vries / R. Zevenhuijzen

Team members:

Giedo Kingma	Jasper Schulte	Michiel de Jager	Mark Baas	Robert Rinnooy Kan
0305758	0146145	956008	0305715	0102288
giedokingma@hotmail.com	zjaaspoer@hotmail.com	mdejager1977@gmail.com	mgjbaas@gmail.com	Robert.RinnooyKan@student.uva.nl

Table of Contents

This document presents a project about a Bookstore database application. In this project we have build the bookstore as a java application. This document describes the whole project by first giving the background of the project, than the design of the Database by an ER diagram and after that we give a description of our work. Our work is dived into two separate issues: first the designs of the graphical user interfaces and second the java coding. After that we will show the results of our project and also we will describe a few scenarios where we will test our database application. There is also a manual available in this document, describing how to work with the application. In the end we will give our conclusions and a reflection on the whole project, especially working in a team.

1. Introduction

During this month we have worked on a project following the Database course in semester 2b. The goal of this project is to gain additional knowledge about databases. We did that by learning to design and implement a real life database application, the bookstore. Because the GUI on top of the database and the connection to the database had to be programmed in Java, it was also important to improve our programming skills in Java. Besides all the technical work we have also learned to work in a team and learned to divide tasks among or team members.

2. Background of the Project/Research

Generally, a bookstore application is a database system which has the capabilities of storing and retrieving information about books, customers, and orders. In this project, we are considering to develop a sample centralized relational Bookstore application database for the clerks and managers at a book store. The overview of this application is given in Figure 1. This application may be further used for additional developments. This database can be used to store the records of customers and their preferences, the technical/non-technical books, magazines to which customers can be subscribed, and the customer-orders to the shop (e.g. telephone orders of customer for books, etc.), to be sent to their address for example, by regular mail.

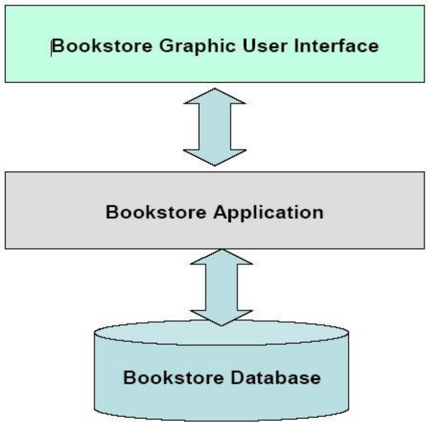


Figure 1: Architecture of the Bookstore

Our bookstore application is a typical e-commerce application. In the future it can be used for data mining. The bookstore application can be extended with an advertisement module. The advertisements send to customers can be generated by data mining processes. Data mining can do this by finding relationships between the books that someone ordered in the past and forecast what books someone would like in the future and send the customer a personalized advertisement about those books (Laura Squier, What is data mining, 2001).

3. Description of Work

Our work method followed the proposed project steps quite closely. The division of the time over the three main project phases although didn't come close to the prognosis given in the schedule. More then three quarters of the total time spent on the project went into completing to some extent Phase Three, which consists of the design of our final user-interface client.

3.1 ER Diagram

The design of the entity relationship model marked the start of our project. In this first step the scope of our final program was decided on. The reason for this is that every object added to this diagram would mean programming a whole new table in the final application. As work progressed the diagram was slimmed down a bit and two objects that we decided using at first were dropped later due to massive time pressure.

Our final diagram consists of five objects, which all have their own attributes, ranging from just three to ten. The relationships between different objects are quite straightforward, but for completeness we will discuss them here. *Customer* has a one to many relationship with *Orders*, for the reason that one and the same order can't be placed by different customers, but one customer is allowed to place multiple orders. *Orders* has a one to one relationship with *Book* for the reason that as to complexity reasons each order can only contain one book. Furthermore *Book* has a many to one relationship with *Author*, for the simple reason that one author can write multiple books, but in our simplified world multiple authors can't write the same book. The existence of co-authoring is hereby disregarded. The last object is *Magazine*, whereto customers can have a subscription, which explains the many to many relationship.
The model is represented in Figure 2.

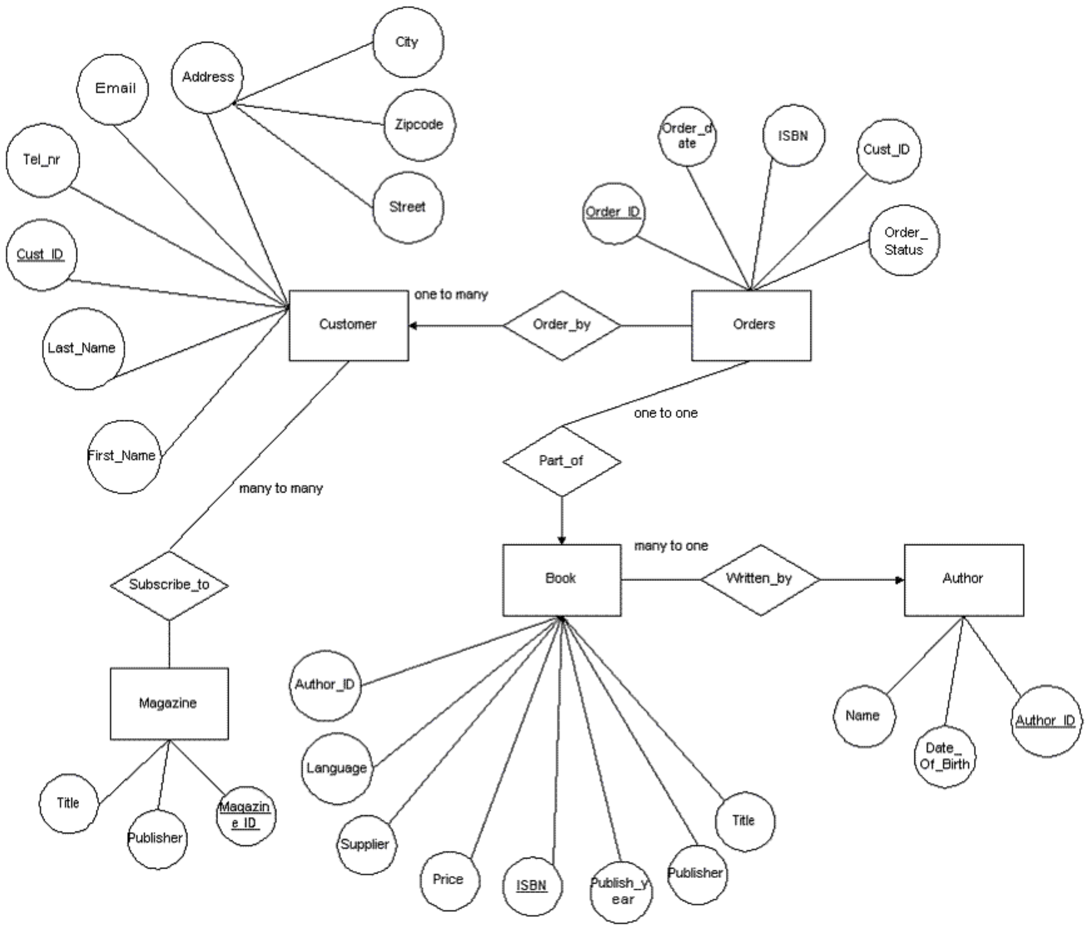
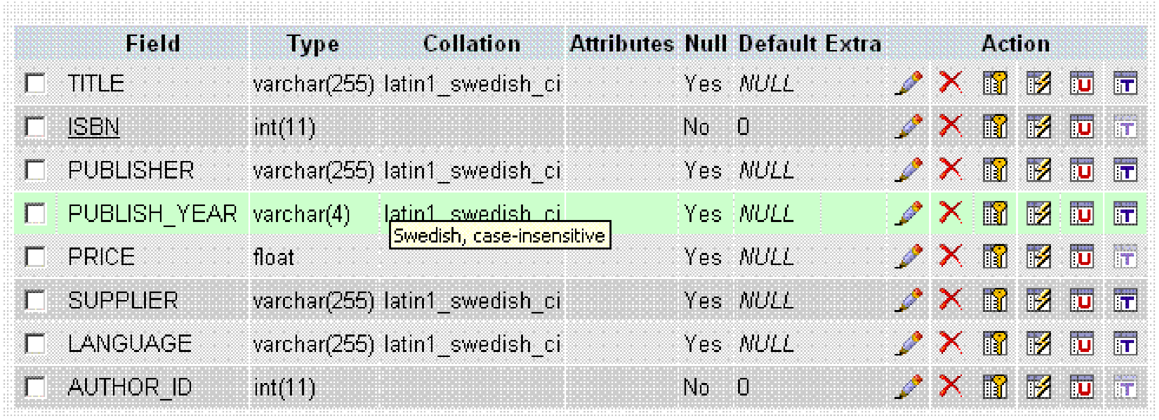


Figure 2: Entity Relationship model

3.2 Database design

Using the program SqlStudio we were able to input all the tables derived from the ER-diagram. After successfully instantiating these, some simple sample entries were entered into the tables again using SqlStudio. After a server migration accessing our tables became even easier through PhpMyAdmin.



Field	Type	Collation	Attributes	Null	Default	Extra	Action
TITLE	varchar(255)	latin1_swedish_ci		Yes	NULL		
ISBN	int(11)			No	0		
PUBLISHER	varchar(255)	latin1_swedish_ci		Yes	NULL		
PUBLISH_YEAR	varchar(4)	latin1_swedish_ci		Yes	NULL		
PRICE	float			Yes	NULL		
SUPPLIER	varchar(255)	latin1_swedish_ci		Yes	NULL		
LANGUAGE	varchar(255)	latin1_swedish_ci		Yes	NULL		
AUTHOR_ID	int(11)			No	0		

Swedish, case-insensitive

Figure 3: Example of one of our tables in PhpMyAdmin

3.3 Java Coding

The coding of the java and especially the coding of the interaction between the database and our code started right from the beginning and was completed quite soon. Simple command-line actions like interaction with the database was possible a lot sooner then we expected, but then…

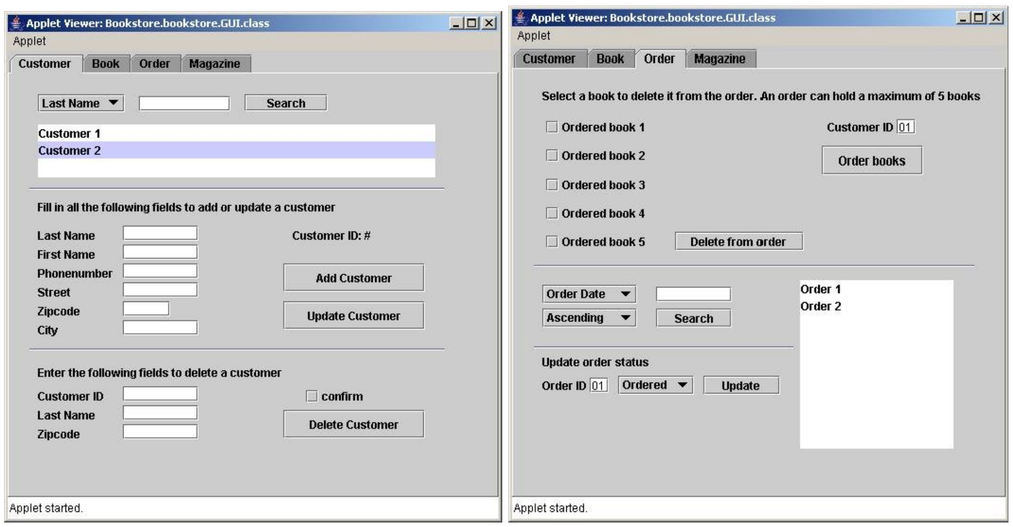
3.4 GUI Design

The user interface went through several lifecycles, which wasn't much of a problem, though still a lot of work using a WYSIWYG plug-in for Eclipse called Jigloo. Because the users-interface offers a so much larger scope of option and listings, then the command-line like java code, which was up and running so soon, the implementation of all the new features proposed by the user interface has taken the maximum of our programmers (in terms of time and effort). During the whole process we were trying to use as much as possible the guidelines in the Java book (Deitel) concerning user interfaces.

4. Presentation of the Client Interface and results

As can be seen in figure 4, the idea of creating an interface with different tabs was adopted from the very beginning. The interface has evolved since then from an empty container, to a much more logical and user friendly way to easily interact with our database, without ever knowing how this interaction took place (derived sql statements). Especially listing the result in comprehensible dynamic tables turned out to be a nightmare in daytime, swallowing up two precious programming days of our lead programmer in the final week.

As we will demonstrate further down with a scenario, the client workflow is now intuitive and comprehensible with descriptive labeling on the buttons, enough space in the text fields to show all that is inside these, and a smart system that restricts the number of columns needed and still is able to show al content of the specific row.



Fill in all the following fields to add or update a customer

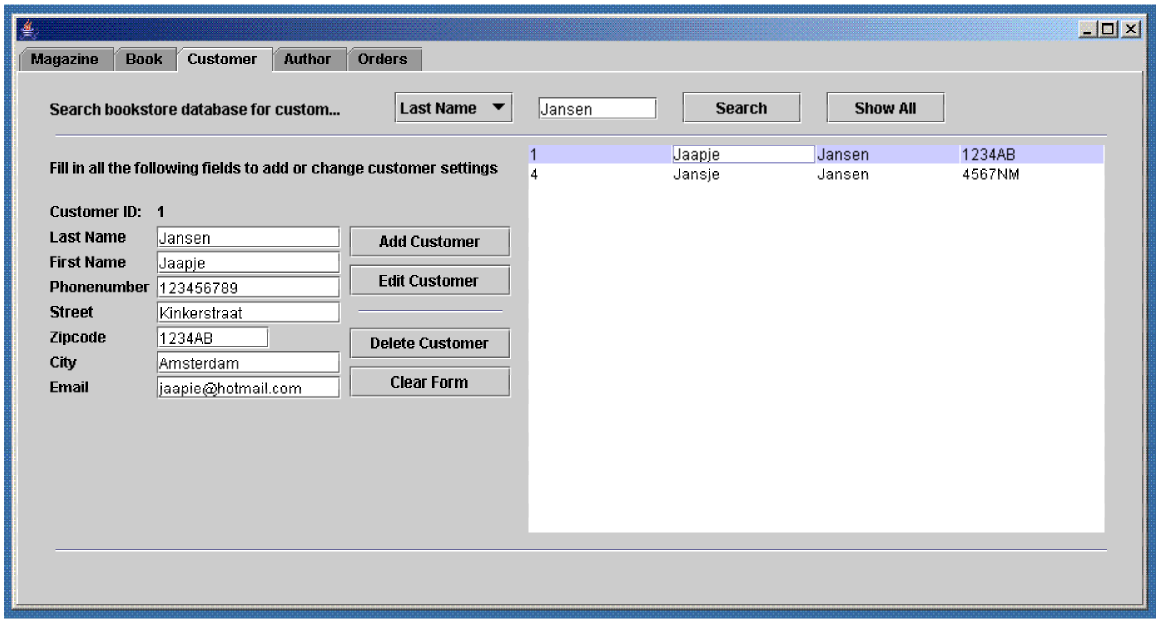
Enter the following fields to delete a customer

Select a book to delete it from the order. An order can hold a maximum of 5 books

Figure 4: Two screenshots of some of our earlier GUI's

Where in the first GUI designs every tab had its own distinct layout, our final layout shows a consistency between different tabs, which one would expect to find in professional user interfaces. Every tab start with a search dialog and beneath this the fields specifically needed for a certain object. For the customer tab this means: after a customer is found using the search option, or is selected from the full list after clicking the show all button, all the attributes which were not shown in de table are shown on the left in the fully editable attribute fields. New customers are automatically assigned to a unique customer ID and easily added at the bottom of the table. Deleting is also as easy as selecting a customer and hitting the right button.

Further description of the interface and its features can be found in the scenario (chapter 5) and manual (appendix B).



Search bookstore database for custom... Last Name: Jansen

Fill in all the following fields to add or change customer settings

Customer ID:	1
Last Name	Jansen
First Name	Jaapje
Phonenumber	123456789
Street	Kinkerstraat
Zipcode	1234AB
City	Amsterdam
Email	jaapie@hotmail.com

1	Jaapje	Jansen	1234AB
4	Jansje	Jansen	4567NM

Figure 5: A screenshot of our final GUI

5. Case Studies/Scenarios Description

For the evaluation of our software project, as directed, we chose to use scenario based software checking. While designing the user interface the usability was constantly kept in mind, but the flow of usability between different screens was as always easily forgotten. Our scenario's therefore stretch on between the different screens of our application. Our first scenario was rather straightforward and checked the two basic and most used functionalities: adding a first time customer to the database and ordering a book for a customer. In that scenario the book was ordered for the first time customer, but the procedure would be roughly the same as when ordering a book for a customer who was already present in the database. This first scenario, which actually helped us a lot in improving our interface, can be found in appendix A.

5.1 Scenario 1

A first time customer *Don Jonson* calls the bookstore. He wants to order the book *The Da Vinci Code* and subscribe to the magazine *Spongebob*. His address is *Honingerdijk 35, 3062 NW Rotterdam*. His telephone number is *010-4127618* and his email *donjonson@hotmail.com*.

The clerk starts our bookstore application. Because he wants to add a customer he clicks the **customer** tab. After entering all the information given by Mister Jonson he clicks the **add customer** button. Mister Jonson's name and customer ID appear at the bottom of the table on the right.

Next the clek click the **Magazine** tab, because Mister Jonson wants to subscribe to one. He simply clicks on Mister Jonsons name in the left table and then on the Spongebob magazine on the right and then hits the **subscribe** button.

Next the clerk clicks the **Book** tab because Mister Jonson wants to order a book. At the top of this tab the clerk chooses **Title** from the drop down list and enters the book name. Then he hits the **search** button. The right book appears at the top of the list, and the clerk click on it, and then on Mister Jonson name in the right of the two smaller tables on the bottom left of this tab.

The clerk now clicks on **order** tab to check if the order can be seen in the table of orders, and it is at the top.

Thank you for your orders Mister Jonson!

5.2 Scenario 2

The clerk gets a phone call from the supplier *cb* that a book *The Da Vinci Code* has been delivered to a customer *Don Jonson*, and knows he needs to update the order status on this order.

The clerk starts our bookstore application. Because he wants to check an order he clicks to the **order** tab. From the dropdown list at the top the clerk selects **name**. The he enters Mister Jonsons name in the field next to it and hits the **search** button.

The first hit is the order of the book mentioned by the supplier. The clerk selects this order, and then changes the order status on the right into **delivered**. Then the clerk hits the **update selected order** button to complete this task.

Thank you for your call Mister Supplier!

5.3 Scenario 3

The clerk gets a phone call from a customer *Don Jonson* who wishes to change his address *Honingerdijk 35* to *Honingerdijk 34* so that in the future his books don't get delivered to the neighbors.

The clerk starts our bookstore application. Because he wants to update a customer he clicks the **customer** tab. From the dropdown list at the top the clerk selects **name**. The he enters Mister Jonsons name in the field next to it and hits the **search** button.

After clicking the first hit on the list, Mister Jonson's customer settings appear on the left. After changing the adres, the clerk hits the **edit customer** button to finalize the task.

Thank you for your call Mister Jonson!

5.4 Scenario 4

The clerk gets another phone call from the supplier *cb* that he can now offer a book *El amor en los tiempos del cólera* from a new writer *Gabriel Garcia Marquez*. The book is in *Spanish* and the further info needed is: ISBN *90 290 76097*, Publisher *Meulenhoff*, Publish date *2004*, and the price *49.99*. Gabriels date of birth is *March the 6th 1927*.

Because the clerk knows, that before adding a book of a new author he has to adds the author first (he has read the manual from appendix B) he first click to the **author** tab.

He enters the name and date of birth of the author into the two field on the left and hits the **add author** button.

Now he can enter the new book. For this he clicks the **book** tab. Now he enters all the information supplied by the supplier in the fields on the left. Fanally he selects the author of the book from the table on the bottom left and clicks **add book** to complete the task.

Thank you for your call Mister Supplier!

6. Conclusions and reflection

In this project we learned a lot of practical things about databases. After the theoretical database course in semester 2b we learned how to build a real database ourselves. Building a database application from scratch was a hard thing to do, but with help of the lab instructors we managed to write the whole application in Java. In the beginning we didn’t have too much Java experience, as mentioned during the forming of the teams. But after one month we have learned a lot about Java in combination with Databases during this course.

The theoretical lectures were of great use as background information about databases and graphical user interface design. We tried to implement all the information of the classroom lectures in the Bookstore application. We realize there is still room for improvement and extension of the application with extra modules, like the advertisement module mentioned before.

As a group we learned to divide the work according to everyone’s skills. We learned what it is to work on a project this size and how to work as a team during 4 intense weeks.

7. References

A. Silverschatz, H. Korth, and S. Sudarshan, Database System Concepts,
4th ed: McGraw-Hill, 2002.

Deitel, H.M., Deitel, P.J. – Java How to Program, Prentice Hall, Pearson Education Int’l, 5th Edition, 2003.

Swing tutorial: http://java.sun.com/docs/books/tutorial/uiswing

JDBC technology:
http://developer.java.sun.com/developer/onlineTraining/Database/JDBC20Intro/

L. Squier, “What is Data Mining?” 2001

8. Annexes: Java Code

Appendix A. Old GUI Scenario

A first time client enters the bookstore, and wishes to order a book and subscribe to a magazine.

The clerk starts our bookstore database application.

First he enters the customers’ personal information into the first application screen.

After hitting the "Add Customer" button he remembers the unique customer ID and then clicks the "BOOK" tab.

The clerk asks the customer which book he wishes to order.

After learning the name of the author and the title he enters the into the "BOOK" tab fields.

After hitting the search button the book appears on the right side.

He enters the ISBN number into the ISBN field, enters the newly assigned customer ID in the ID field and hits the order button, which places the book in the customers "Shopping Cart".

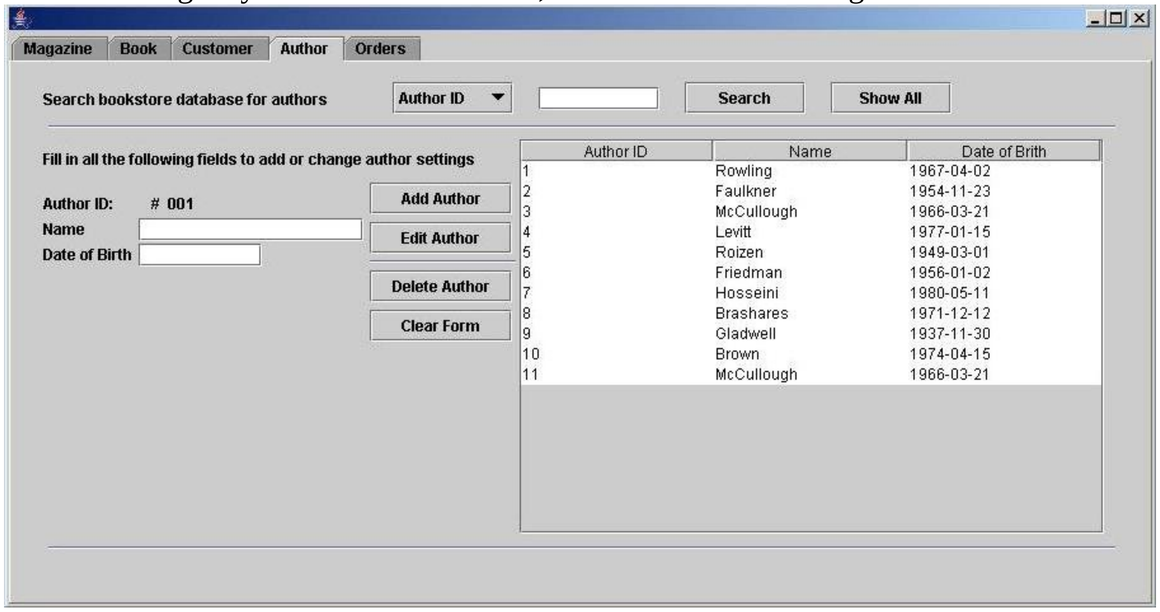
Hereafter the clerk clicks on to the "ORDER" tab, and checks if the customer is content with the order, and then enters the ID once more and hits the order button to finalize the process.

Appendix B. User Manual

Welcome to our Bookstore Database! In this short manual we will explain to you how to use the bookstore application. As you can see in at the top of our screenshots, our application contains 5 tabs: author, book, customer, magazine and orders. You can switch between the tabs by just clicking one. In the next sections we will describe the functions of the database application per tab. All the tabs have the same layout, all containing a “Show All” and a “Clear Form” button. There is also a “Search” button in all tabs, which you can use if you know a specific customer, book or author, etc. etc. and you want to find it in the database.

One thing is very important: if you want to add a new book to the database, you must first insert the author if the book is written by an author that’s not already in the database!!

Now we will give you the first screenshot, it’s the Author tab in figure A:


Magazine
Book
Customer
Author
Orders
Search bookstore database for authors
Author ID
Search
Show All
Fill in all the following fields to add or change author settings
Add Author
Edit Author
Delete Author
Clear Form

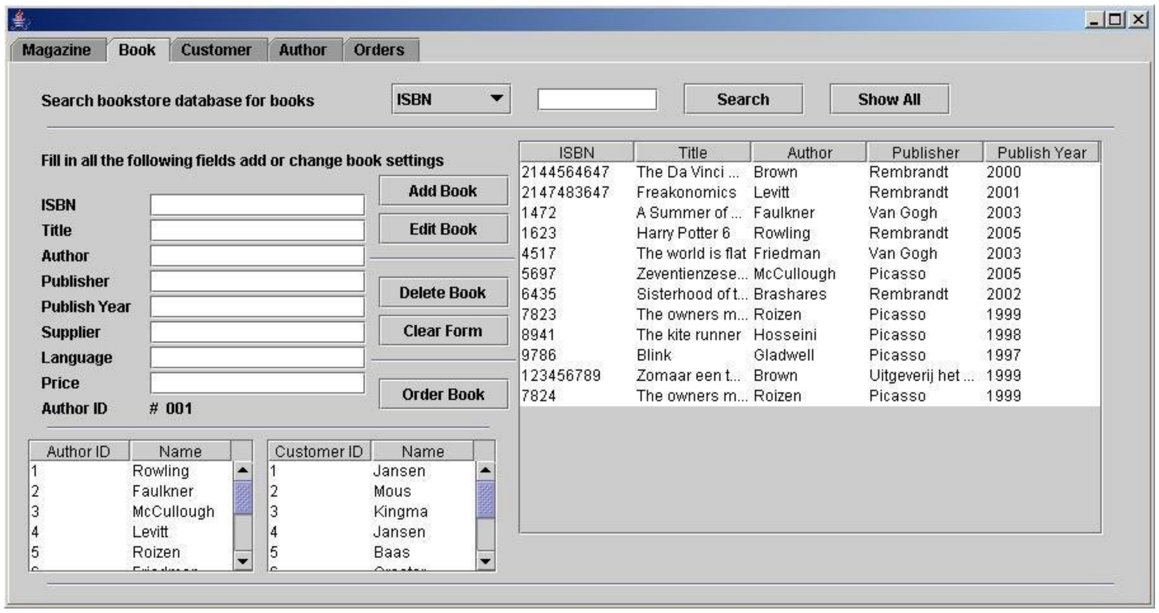
Author ID: # 001
Name
Date of Birth

Author ID	Name	Date of Brith
1	Rowling	1967-04-02
2	Faulkner	1954-11-23
3	McCullough	1966-03-21
4	Levitt	1977-01-15
5	Roizen	1949-03-01
6	Friedman	1956-01-02
7	Hosseini	1980-05-11
8	Brashares	1971-12-12
9	Gladwell	1937-11-30
10	Brown	1974-04-15
11	McCullough	1966-03-21

Figure A: the Author tab

New author:	fill in the name and date of birth and click “Add Author” button.
Edit author:	first find the author you want to edit by clicking the “show all” button and select the author you want to edit or by searching the author. The data appears in the left input boxes and there you can edit the data. After that, click the “Edit Author” button.
Delete author:	first click “show all” button and select the author you want to delete. The data appears in the left input boxes. After that, click the “Delete Author” button.

Next, a screenshot of the Book tab in figure B:



Search bookstore database for books   ISBN

Fill in all the following fields add or change book settings

ISBN	
Title	
Author	
Publisher	
Publish Year	
Supplier	
Language	
Price	
Author ID	# 001

ISBN	Title	Author	Publisher	Publish Year
2144564647	The Da Vinci ...	Brown	Rembrandt	2000
2147483647	Freakonomics	Levitt	Rembrandt	2001
1472	A Summer of ...	Faulkner	Van Gogh	2003
1623	Harry Potter 6	Rowling	Rembrandt	2005
4517	The world is flat	Friedman	Van Gogh	2003
5697	Zeventienzese...	McCullough	Picasso	2005
6435	Sisterhood of t...	Brashares	Rembrandt	2002
7823	The owners m...	Roizen	Picasso	1999
8941	The kite runner	Hosseini	Picasso	1998
9786	Blink	Gladwell	Picasso	1997
123456789	Zomaar een t...	Brown	Uitgeverij het ...	1999
7824	The owners m...	Roizen	Picasso	1999

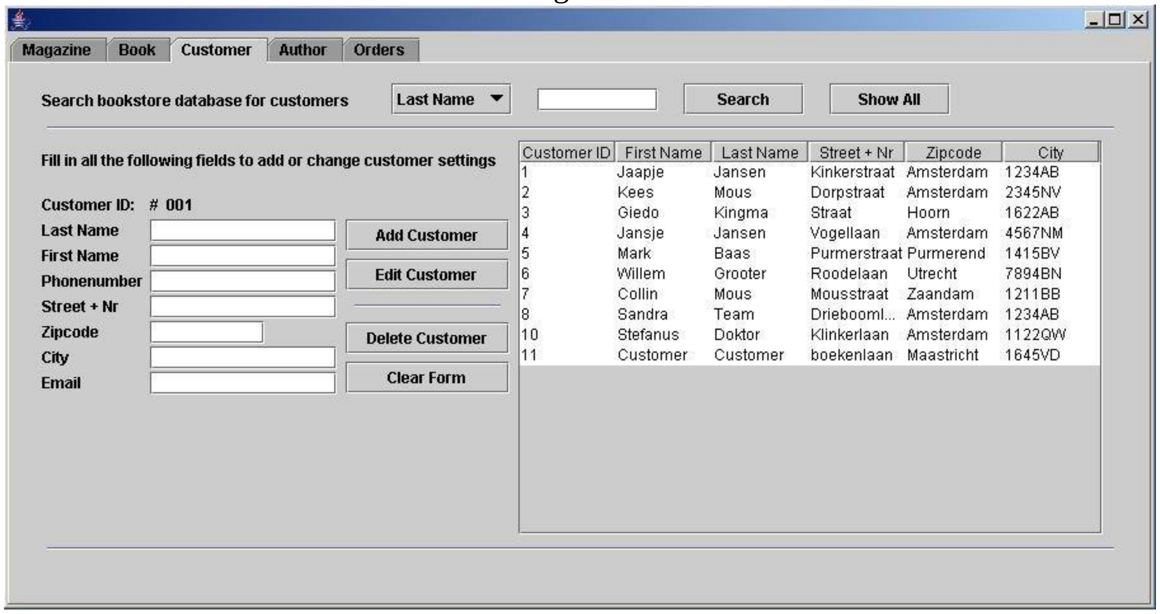
Author ID	Name
1	Rowling
2	Faulkner
3	McCullough
4	Levitt
5	Roizen

Customer ID	Name
1	Jansen
2	Mous
3	Kingma
4	Jansen
5	Baas

Figure B: the Book tab

The add, edit and delete actions are the same as in the Author tab. But in this screen you also see two search boxes in bottom left corner of the screenshot. Here you can easily look up author ID’s and if you want to order a book you first select a book and then a customer and then you click the “Order Book” button.

Next a screenshot of the Customer tab in figure C:



Search bookstore database for customers   Last Name

Fill in all the following fields to add or change customer settings

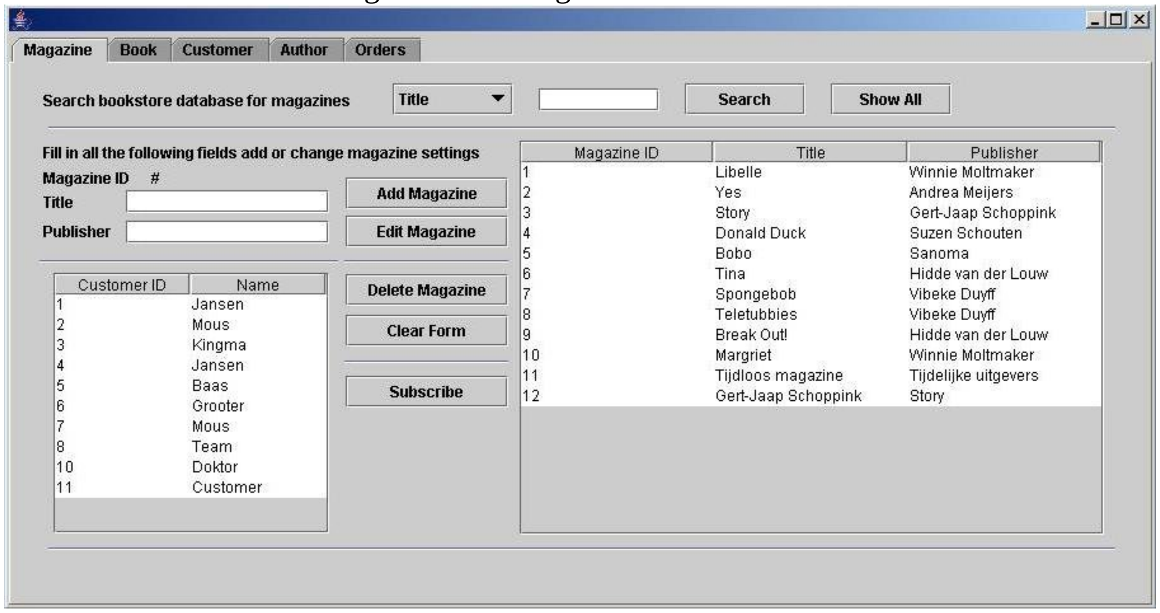
Customer ID:	# 001
Last Name	
First Name	
Phonenumber	
Street + Nr	
Zipcode	
City	
Email	

Customer ID	First Name	Last Name	Street + Nr	Zipcode	City
1	Jaapje	Jansen	Kinkerstraat	Amsterdam	1234AB
2	Kees	Mous	Dorpstraat	Amsterdam	2345NV
3	Giedo	Kingma	Straat	Hoorn	1622AB
4	Jansje	Jansen	Vogellaan	Amsterdam	4567NM
5	Mark	Baas	Purmerstraat	Purmerend	1415BV
6	Willem	Grooter	Roodelaan	Utrecht	7894BN
7	Collin	Mous	Mousstraat	Zaandam	1211BB
8	Sandra	Team	DrieboomI...	Amsterdam	1234AB
10	Stefanus	Doktor	Klinkerlaan	Amsterdam	1122QW
11	Customer	Customer	boekenlaan	Maastricht	1645VD

Figure C: the Customer tab

In the customer tab the add, edit and delete buttons work the same way as in the author tab and need no extra explanation.

Next a screenshot of the Magazine tab in figure D:



Search bookstore database for magazines: Title

Fill in all the following fields add or change magazine settings

Magazine ID #

Title

Publisher

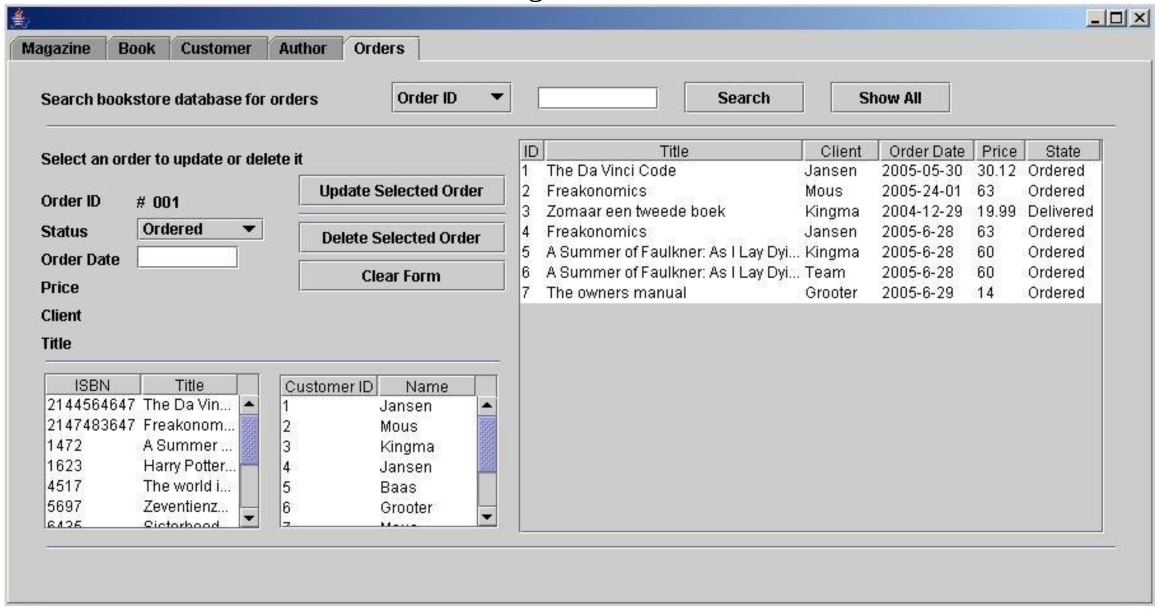
Customer ID	Name
1	Jansen
2	Mous
3	Kingma
4	Jansen
5	Baas
6	Grooter
7	Mous
8	Team
10	Doktor
11	Customer

Magazine ID	Title	Publisher
1	Libelle	Winnie Moltmaker
2	Yes	Andrea Meijers
3	Story	Gert-Jaap Schoppink
4	Donald Duck	Suzen Schouten
5	Bobo	Sanoma
6	Tina	Hidde van der Louw
7	Spongebob	Vibeke Duyff
8	Teletubbies	Vibeke Duyff
9	Break Out!	Hidde van der Louw
10	Margriet	Winnie Moltmaker
11	Tijdloos magazine	Tijdelijke uitgevers
12	Gert-Jaap Schoppink	Story

Figure D: the Magazine tab

If you want to subscribe a certain customer to a magazine, you first select the magazine in the right box. After that you select the customer in the bottom left box and then you click the “Subscribe” button.

Our last screenshot is the Orders tab in figure E:



Search bookstore database for orders: Order ID

Select an order to update or delete it

Order ID # 001

Status: Ordered

Order Date

Price

Client

Title

ISBN	Title
2144564647	The Da Vin...
2147483647	Freakonom...
1472	A Summer ...
1623	Harry Potter...
4517	The world i...
5697	Zeventienz...

Customer ID	Name
1	Jansen
2	Mous
3	Kingma
4	Jansen
5	Baas
6	Grooter

ID	Title	Client	Order Date	Price	State
1	The Da Vinci Code	Jansen	2005-05-30	30.12	Ordered
2	Freakonomics	Mous	2005-24-01	63	Ordered
3	Zomaar een tweede boek	Kingma	2004-12-29	19.99	Delivered
4	Freakonomics	Jansen	2005-6-28	63	Ordered
5	A Summer of Faulkner: As I Lay Dyi...	Kingma	2005-6-28	60	Ordered
6	A Summer of Faulkner: As I Lay Dyi...	Team	2005-6-28	60	Ordered
7	The owners manual	Grooter	2005-6-29	14	Ordered

Figure E: the Orders tab

In this tab you can find all the orders in the bookstore database. After you clicked the “Order Book” button in the Book tab, the order appears here in the Orders tab.

Appendix C. Shared.java

```
package bookstore;

import java.sql.*;
import javax.swing.*;

public class Shared {

//
//              FindID
//
public int FindID(DBconnection con, String table) {
                // list all books in the database
                String column = "";
                if( table == "AUTHOR"   ) column = "AUTHOR_ID";
                if( table == "CUSTOMER" ) column = "CUST_ID";
                if( table == "MAGAZINE" ) column = "MAGAZINE_ID";
                if( table == "ORDERS"   ) column = "ORDER_ID";

                if (con != null) {
                                try {
                                                Statement stmt = con.con.createStatement();
                                                String insertSQL =
                                                                "select "
                                                                + column
                                                                + " from "
                                                                + table
                                                                + " order by "
                                                                + column;

                                                ResultSet result = stmt.executeQuery(insertSQL);

                                                int foundID = 1;

                                                while (result.next()) {
                                                                foundID = result.getInt(column) + 1;

                                                }

                                                return foundID;

                                }
                                catch (SQLException ex) {
                                                System.err.println(ex.getMessage());
                                                return 0;
                                }

                }
                return 0;
}

//
//              Search
//
public ResultSet Search(DBconnection con, String table, String column, String value) {

                if (con != null) {
                                try {
                                                Statement stmt = con.con.createStatement();
                                                String insertSQL;

                                                //"select * from ORDERS, BOOK, CUSTOMER where ORDERS.ISBN = BOOK.ISBN and
ORDERS.CUST_ID = CUSTOMER.CUST_ID order by ORDER_ID";
                                                if ( table == "ORDERS" )
                                                                insertSQL =
                                                                                "select * from ORDERS, BOOK, CUSTOMER where "
                                                                                + column
                                                + "='"
                                                + value
                                                + "' and ORDERS.ISBN = BOOK.ISBN and ORDERS.CUST_ID
= CUSTOMER.CUST_ID order by ORDER_ID";
                        else if ( table == "BOOK" )
                                insertSQL =
                                        "select * from BOOK, AUTHOR where "
                                                + column
                                                + "='"
                                                + value
                                                + "' and BOOK.AUTHOR_ID = AUTHOR.AUTHOR_ID order by
ISBN";
                        else
                        insertSQL =
                                "select * from "
                                        + table
                                        + " where "
                                        + column
                                        + "='"
                                        + value
                                        + "'";

                        ResultSet result = stmt.executeQuery(insertSQL);

                        if (result==null) {
                                        return null;
                                }
                                else {
                                        return result;
                                }

                } catch (SQLException ex) {JOptionPane.showMessageDialog (     null, "updateString:" + ex.getMessage());
                        return null;
                }


        } else {
                JOptionPane.showMessageDialog (              null, "updateString: Try Failed");
                return null;
        }

}

//
//  Update
//
public void Update(DBconnection con, String table, String column, int id, String value) {
        String idColumn = "";
        if( table == "AUTHOR"   ) idColumn = "AUTHOR_ID";
        if( table == "BOOK"     ) idColumn = "ISBN";
        if( table == "CUSTOMER" ) idColumn = "CUST_ID";
        if( table == "MAGAZINE" ) idColumn = "MAGAZINE_ID";
        if( table == "ORDERS"   ) idColumn = "ORDER_ID";

        if (con != null) {
                try {
                        Statement stmt = con.con.createStatement();
                        String insertSQL =
                                "update "
                                + table
                                + " set "
                                + column
                                + "='"
                                + value
                                + "' where "
                                + idColumn
                                + "="
                                + id;

                        int rowcount = stmt.executeUpdate(insertSQL);
                        if (rowcount == 0) {
                                JOptionPane.showMessageDialog (          null, "updateString: Nothing Updated");
                        }

                } catch (SQLException ex) {JOptionPane.showMessageDialog (    null, "updateString:" + ex.getMessage());
                }


        } else {
                JOptionPane.showMessageDialog (          null, "updateString: Try Failed");
        }

}

//
//  Delete
//
public void Delete(DBconnection con, String table, int ID) {
        String column = "";
        if( table == "AUTHOR"   ) column = "AUTHOR_ID";
        if( table == "BOOK"     ) column = "ISBN";
        if( table == "CUSTOMER" ) column = "CUST_ID";
        if( table == "MAGAZINE" ) column = "MAGAZINE_ID";
        if( table == "ORDERS"   ) column = "ORDER_ID";

        if (con != null) {
                try {
                        Statement stmt = con.con.createStatement();
                        String insertSQL =
                                "delete from "
                                + table
                                + " where "
                                + column
                                + "="
                                + ID;

                        int rowcount = stmt.executeUpdate(insertSQL);
                        if (rowcount == 0) {
                                JOptionPane.showMessageDialog (          null, "Delete: Nothing Deleted");
                        }

                } catch (SQLException ex) {
                        JOptionPane.showMessageDialog (          null, "Delete:" + ex.getMessage());
                }

        } else {
                JOptionPane.showMessageDialog (          null, "Delete: Deletion unsuccesfull");
        }

}



}
```

Appendix D BookHandler.java

```
package bookstore;
import jav.sql.*;
public class BookHandler {
		public ResultSet ListAllBooks(DBconnection con) {
				// list all books in the database
				if (con != null) {
						try {
								Statement stmt = con.con.createStatement();
								String insertSQL =
										"select TITLE, NAME, ISBN, PUBLISHER, PUBLISH_YEAR, PRICE,
LANGUAGE from BOOK, AUTHOR where BOOK.AUTHOR_ID = AUTHOR.AUTHOR_ID order by TITLE";

								ResultSet result = stmt.executeQuery(insertSQL);

								if (result == null) {
										System.err.println("No books found");
										return null;

								} else {
										return result;
								}

						} catch (SQLException ex) {
								System.err.println(ex.getMessage());
								return null;
						}
				} else {
						return null;
				}
		}

		public ResultSet GetBook(DBconnection con, int id) {
						if (con != null) {
								try {
										Statement stmt = con.con.createStatement();
										String insertSQL = "select * from BOOK, AUTHOR where
BOOK.AUTHOR_ID = AUTHOR.AUTHOR_ID and ISBN=" + id;

										ResultSet result = stmt.executeQuery(insertSQL);

										if (result==null) {
												Systemyou are crazyn("No book found");
												return null;
										}

										else {
												return result;
										}

								}
								catch (SQLException ex) {
										System.err.println(ex.getMessage());
										return null;
								}
						} else {
								return null;
						}
				}

		public void AddBook(DBconnection con, String TITLE, int ISBN, String PUBLISHER, String PUBLISH_YEAR, float
PRICE, String SUPPLIER, String LANGUAGE,		int AUTHOR_ID)
		{
				if (con != null) {
						try {
								Statement stmt = con.con.createStatement();
```

```
                                                String insertSQL =
                                                            "insert into BOOK
(TITLE,ISBN,PUBLISHER,PUBLISH_YEAR,PRICE,SUPPLIER,LANGUAGE,AUTHOR_ID) values ('"
                                                                        + TITLE
                                                                        + "','"
                                                                        + ISBN
                                                                        + "','"
                                                                        + PUBLISHER
                                                                        + "','"
                                                                        + PUBLISH_YEAR
                                                                        + "','"
                                                                        + PRICE
                                                                        + "','"
                                                                        + SUPPLIER
                                                                        + "','"
                                                                        + LANGUAGE
                                                                        + "','"
                                                                        + AUTHOR_ID
                                                                        + "')";
                                                int rowcount = stmt.executeUpdate(insertSQL);

                                                if (rowcount == 0) {
                                                            System.err.println("No authors added");
                                                }

                                    } catch (SQLException ex) {
                                                System.err.println(ex.getMessage());
                                    }
                        } else {
                                    System.err.println("Add book failed");
                        }
            }
}
```