Systems Analysis

and Design

Instructor : Huang, Chuen-Min

**Teamwork2 ver.1**

Group 6

|  |  |
| --- | --- |
| ID | Name |
| A10523006 | Maggie |
| A10523049 | Peggy |
| B10423003 | Kurumi |
| B10423029 | Bean |
| B10523020 | Kendy |
| B10523030 | Jerry |
| B10523053 | Lynn |
| M10723001 | Joe |
|  |  |
|  |  |
| Date 2018/05/29 | |

Content

[**1)** 1](#_Toc515317875)

[**2)** 5](#_Toc515317880)

[1. Control Coupling 5](#_Toc515317881)

[2. Interaction, Data Coupling 6](#_Toc515317882)

[3. Stamp Coupling 7](#_Toc515317883)

[**3)** 8](#_Toc515317884)

[1. Function Cohesion 8](#_Toc515317885)

[2. Temporal Cohesion 10](#_Toc515317886)

[3. Logical Chhesion 11](#_Toc515317887)

[**4)** 13](#_Toc515317888)

[1. Type of Class Connascence 13](#_Toc515317889)

[2. Name Connascence 13](#_Toc515317890)

[3. Convention Connascence 14](#_Toc515317891)

[**5)** 15](#_Toc515317892)

[C**RC Card** 15](#_Toc515317893)

[**Text File** 16](#_Toc515317896)

[**6)** 17](#_Toc515317897)

[ **Contract** 17](#_Toc515317898)

[ **Method Specification** 18](#_Toc515317899)

[ **Activity Diagram** 19](#_Toc515317900)

[**7)** 20](#_Toc515317901)

[1. Coupling(Data Coupling) 20](#_Toc515317902)

[2. Cohesion(Cohesion) 21](#_Toc515317903)

[3. Connascence(Convention Connascence) 22](#_Toc515317904)

[**8)** 23](#_Toc515317905)

[**9)** 24](#_Toc515317906)

[ **Class Diagram** 24](#_Toc515317907)

[ **Zero Normal Form** 27](#_Toc515317911)

[ **First Normal Form** 28](#_Toc515317912)

[ **Second Normal Form** 29](#_Toc515317913)

[ **Third Normal Form** 30](#_Toc515317914)

[**10)** 31](#_Toc515317915)

[ **Denormalization** 31](#_Toc515317916)

[ **Class Diagram** 31](#_Toc515317917)

[**11)** 32](#_Toc515317918)

[**Participate In Assignments** 35](#_Toc515317919)

1. **Please explain the Law of Demeter (LoD) by using of your project.**
2. to itself (O itself)

|  |
| --- |
| class InitialController |
| If password equals data’s record, execute its own method. |

1. to objects contained in attributes of itself or a superclass (Any objects created/instantiated within M)

Class InitialController create object UserID & Password itself, then it can use getText() in its method.

|  |
| --- |
| class InitialController |
|  |

1. to an object that is passed as a parameter to the method (M’s parameters)

Method editdata() gets an ArrayList that named input. Then method editdata() changes the ArrayList’s name as AMember.

Method editdata() has an object Member, its name is storeMember. Then ArrayList<Member> AMember equals to input (let input’s type to Member) and get first AMember’s Array to storeMember.

|  |
| --- |
| class LibrarianDBMgr |
|  |

1. to an object that is created by the method (O’s direct component objects)

When addMemberButtonClick method execute, it will create object and use its method.

|  |
| --- |
| class LibrarianController |
| https://scontent-tpe1-1.xx.fbcdn.net/v/t1.15752-9/33746625_2626729484219648_1598431719122272256_n.png?_nc_cat=0&oh=828ca89f3d1f1a17a07031bac50307cf&oe=5B828A4F |

1. **There are six (or seven) types of interaction coupling, each falling on different parts of a good-to-bad continuum. Choose three pieces of your project to describe what types of the coupling they belong to.**
2. Control Coupling

If someone wants to login this system, we need to check user’s inputted ID. If the user uses librarian’s ID, then method onButtonClick() will send two variables to method searchData(). One is user’s ID, the other is what we need to find from database.

|  |
| --- |
| class InitialController |
|  |

1. Interaction, Data Coupling

This is a class for sending an E-mail to member. All the parameter is decided by basic variable. And in this class, it doesn’t need to call other class. It just handles send E-mail by its own.

|  |
| --- |
| class SendEmail |
| C:\Users\Nelson\AppData\Local\Microsoft\Windows\INetCache\Content.Word\123.jpg |

1. Stamp Coupling

System uses class LibraryDBMgr to change paperbook’s state. First of all, system creates member object changeMemberData. System will find a book it wants to change from database, and puts this data in variable havaPaperBook. Then use LibraryDBMgr’s method editData() to change book’s state. The member object changeMemberData will be its parameter.

|  |
| --- |
| class LibrarianController |
|  |

1. **There are seven types of method cohesion, choose three pieces of your project to describe what types of the cohesion they belong to.**
2. Function Cohesion

This method’s only function is to check book borrow situation.

|  |
| --- |
| class CheckOverdueBook - 1 |
|  |

|  |
| --- |
| class CheckOverdueBook - 2 |
|  |

|  |
| --- |
| class CheckOverdueBook - 3 |
|  |

|  |
| --- |
| class CheckOverdueBook - 4 |
|  |

|  |
| --- |
| class CheckOverdueBook - 5 |
|  |

1. Temporal Cohesion

Member functions are called at the same time.

|  |
| --- |
| class Time |
|  |

1. Logical Chhesion

Method addDate() can add four data there are member, librarian, paperbook, ebook.

|  |
| --- |
| class LibraryDBMgr |
|  |

1. **Connascence generalized the ideas of cohesion and coupling, use three pieces of your project to describe what types of the connascence they belong to.**
2. Type of Class Connascence

If a class has an attribute of type A, it is tied to the type of the attribute. If the type of the attribute changes, the attribute declaration will have to be changed.

For example, if Stage class changes, LibrarianGUI class’s Stage() method will also be changed.

|  |
| --- |
| class |
|  |

1. Name Connascence

If a method refers to an attribute, it is tied to the name of the attribute. If the attribute’s names changes, the content of the method will have to be changed.

|  |
| --- |
| class Time |
|  |

1. Convention Connascence

If the value’s range changes, every method that used the value would have to be modified.

|  |
| --- |
| class |
| https://scontent-tpe1-1.xx.fbcdn.net/v/t1.15752-9/33686742_2626739197552010_5926925448230469632_n.png?_nc_cat=0&oh=f26aca649126cfb8d9b24b3ce316ca5a&oe=5BC4DBFC |

1. **Use one class from your project that can create a set of invariants and add them to the CRC card or the class diagram.**

* **CRC Card**

Front

|  |  |  |  |
| --- | --- | --- | --- |
| **Class name:** LibrarianController | **ID:**1 | | **Type:** Concrete, Domain |
| **Description:** | | **Association Use Case:** | |
| This class provides librarian to save and edit data of memberships, paper book, and e-book. It also can help librarian search book information and provide book service. | | Manage Paper Book  Manage E-Book  Manage Member  Borrow Book  Return Book  Search Book | |
| **Responsibilities:** | | **Collaborators:** | |
| addMemberButtonClick | | PaperBook | |
| addEbookButtonClick | | Ebook | |
| addBookButtonClick | | Member | |
| editMemberButtonClick | | LibrarianGUI | |
| editEbookButtonClick | | SendEmail | |
| editBookButtonClick | | LibraryDBMgr | |
| deleteMemberButtonClick | | Search | |
| deleteEbookButtonClick | |  | |
| searchBookButtonClick | |  | |
| borrowBookButtonClick | |  | |
| returnBookButtonClick | |  | |
| updatePaperBookStateButtonClick | |  | |

Back

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attributes: | | | | |
| addMemberButton | | (1..1) | (Button) |  |
| addEbookButton | | (1..1) | (Button) |  |
| addBookButton | | (1..1) | (Button) |  |
| editMemberButton | | (1..1) | (Button) |  |
| editEbookButton | | (1..1) | (Button) |  |
| editBookButton | | (1..1) | (Button) |  |
| deleteMemberButton | | (1..1) | (Button) |  |
| deleteEbookButton | | (1..1) | (Button) |  |
| searchBookButton | | (1..1) | (Button) |  |
| borrowBookButton | | (1..1) | (Button) |  |
| returnBookButton | | (1..1) | (Button) |  |
| updatePaperBookStateButton | | (1..1) | (Button) |  |
| f | | (0..1) | (File) | {f = (File) Actionevent.getNewValue()} |
| Relationships: |  | | | |
| Generalization(a-kind-of): |  | | | |
| Aggregation(has-parts): | | | | |
|  | | | | |
| Other Associations: | | | | |
| Manage Paper Book  Manage E-Book  Manage Member  Borrow Book  Return Book  Search Book | | | | |

**Text File**

|  |
| --- |
| LibrarianController class invariants:  F = (File) Actionevent.getNewValue() |

1. **Use a method of a class from your project that can create a contract and describe its algorithm specification. Specify the pre- or post- condition and use both Structured English and an activity diagram to specify the algorithm.**

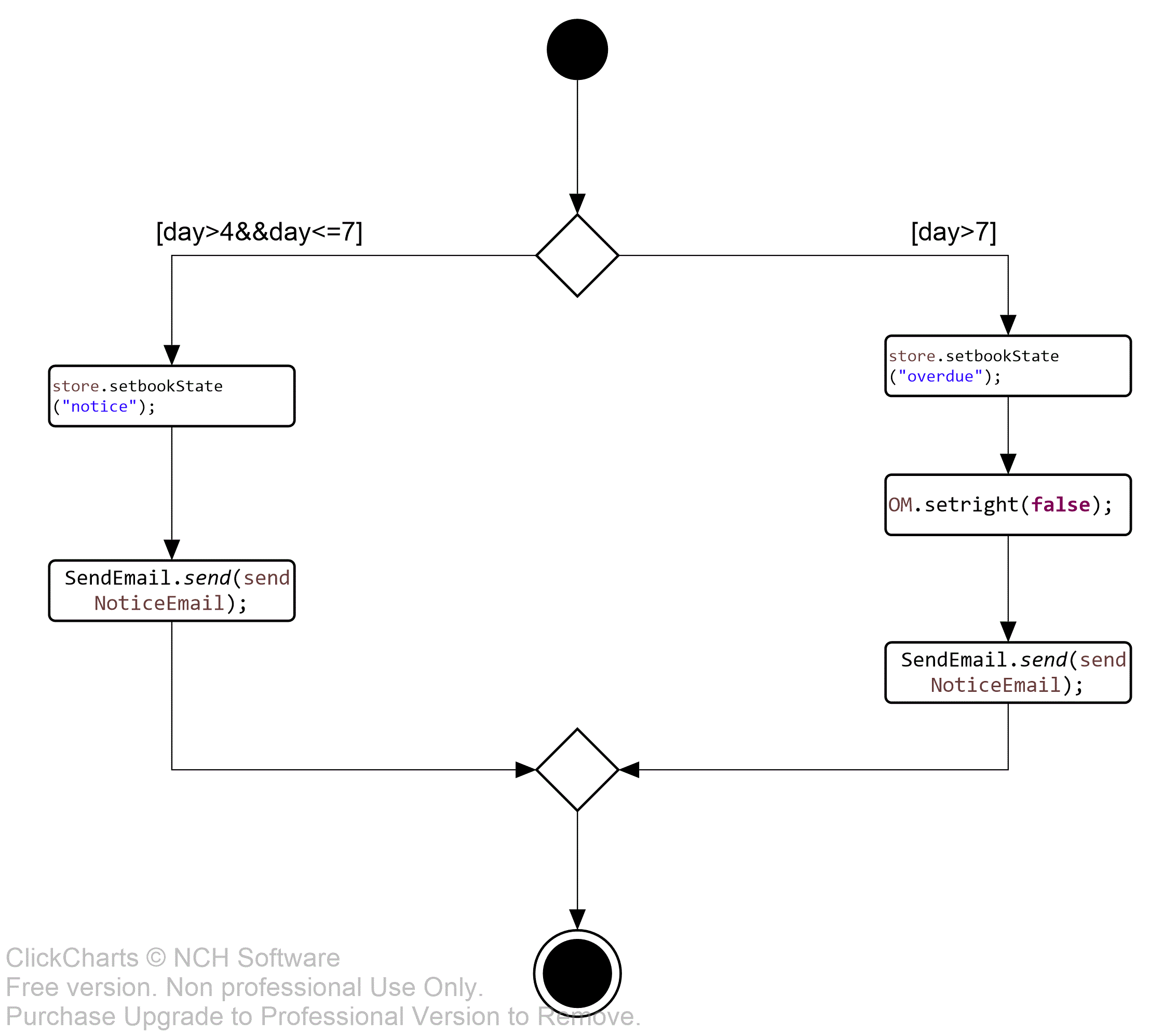
* **Contract**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Method Name:** | run() | | **Class Name:** | CheckOverdueBook | **ID:** | 1 |
| **Client(consumers):** | | Time | | | | |
| **Associated Use Case:** | | | | | | |
| Member | | | | | | |
| **Description of Responsibilities:** | | | | | | |
| We use class checkOverdueBook to calculate if the book that borrowed by member is overdue or not. | | | | | | |
| **Arguments Received:** | | | | | | |
| day:long | | | | | | |
| **Pre-Conditions:** | | | | | | |
| day=(now.getTime() - beginDate.getTime())/(24\*60\*60\*1000) | | | | | | |
| **Post-Conditions:** | | | | | | |
| if(day<=3)  bookState = ("notice");  setnumberOfNoticeBook -1  getnumberOfNoticeBook +1    else if(day<0)  bookState = ("overdue");  getnumberOfNoticeBook -1  getnumberOfOverdueBook +1  setright = (false) | | | | | | |

* **Method Specification**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method Name:** | Time() | | | | **Class Name:** | Time | | **ID:** |  | |
| **Contract ID:** |  | | | | **Programmer:** | Kendy | | Data Due: | | 05/28/2018 |
| **Programming Language:** | | | | | | | | | | |
| Java | | | | | | | | | | |
| **Triggers/Events:** | | | | | | | | | | |
| CheckOverdueBook task = new CheckOverdueBook(); | | | | | | | | | | |
| **Arguments Received:**  **Data Type:** | | | **Notes:** | | | | | | | |
| long | | | Borrowed day minus today | | | | | | | |
|  | | |  | | | | | | | |
| **Messages Sent & Argument Passed:**  **ClassName.MethodName:** | | | | **Data Type:** | | | **Notes:** | | | |
|  | | | |  | | |  | | | |
| **Arguments Returned:**  **Data Type:** | | **Notes:** | | | | | | | | |
| void | |  | | | | | | | | |
| **Algorithm Specification:** | | | | | | | | | | |
| if(day<=3)  {  store.setbookState("notice");    OM.setnumberOfNoticeBook(OM.getnumberOfBorrowBook()-1);  OM.setnumberOfNoticeBook(OM.getnumberOfNoticeBook()+1);  SEP.add(store);  }  else if(day<0)  {  store.setbookState("overdue");    OM.setnumberOfNoticeBook(OM.getnumberOfNoticeBook()-1);  OM.setnumberOfNoticeBook(OM.getnumberOfOverdueBook()+1);  OM.setright(false);  } | | | | | | | | | | |
| **Misc.Notes:** | | | | | | | | | | |
| None | | | | | | | | | | |

* **Activity Diagram**

****

1. **Please evaluate any piece of your project in terms of cohesion, coupling, and connascence perspective.**
2. Coupling(Data Coupling)

Method borrowBookButton() creates an object p, then uses it to read inputted book ID to change book data to borrowed.

|  |
| --- |
| class MemberController |
| MemberController |

1. Cohesion(Function Cohesion)

If member’s returning book day is close, system will send e-mail to notify member. System put the properties information at first, set subject and text to the e-mail and send it.

|  |
| --- |
| class SendEmail |
| send email |

1. Connascence(Convention Connascence)

If the value’s range changes, every method that used the value would have to be modified.

|  |
| --- |
| class PaperBook |
| c LibraryGUI |

1. **Assume that you are going to adopt RDBMs to your project, please describe the referential integrity.**

Table BookBorrowedRecord’s primary key is bookID and bookType, and foreign key is userID and userType which is used to record the book is borrowed by whom. userID and userType are primary key of table Member (userID and userType).

Foreign key’s value can be null because the book may not be borrowed. But if foreign key’s value isn’t null, the value must be Member table’s primary key – user ID’s value. Then table BookBorrowedRecord and table Member can refer each other.

If foreign key’s value isn’t member ID’s value, data will not refer. Refer will be wrong.



Foreign Key



Primary Key

Primary Key

Primary Key

Primary Key

1. **Using the steps of normalization, create a model that represents the file of your project in third normal form. Please make necessary assumptions to explain why the tables are related.**

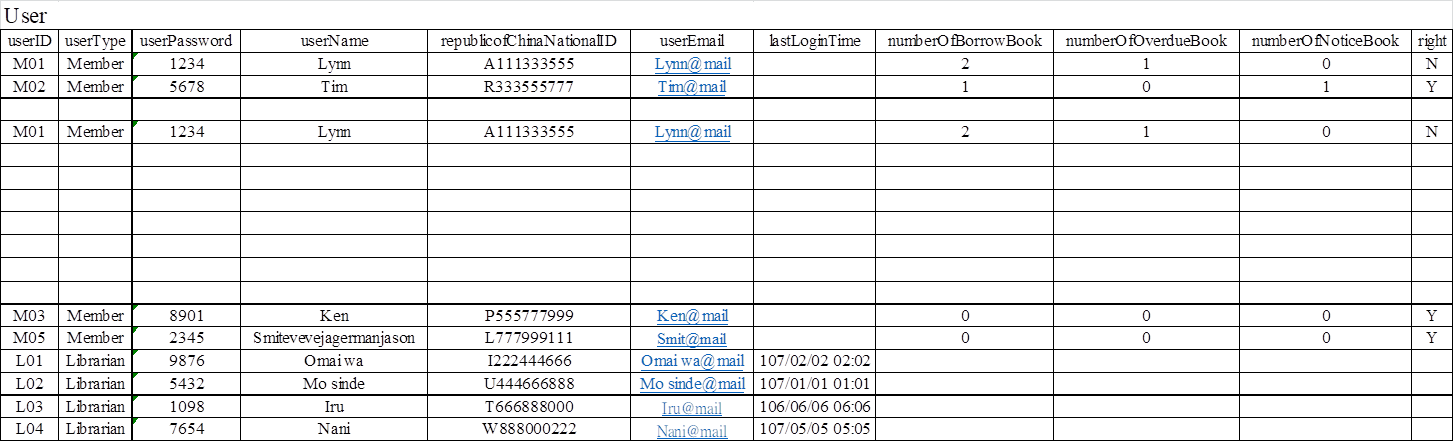
* **Class Diagram**

|  |
| --- |
| First Normal Form Class Diagram |
|  |

|  |
| --- |
| Second Normal Form Class Diagram |
|  |

|  |
| --- |
| Third Normal Form Class Diagram |
|  |

* **Zero Normal Form**

****

* **First Normal Form**



* **Second Normal Form**



* **Third Normal Form**

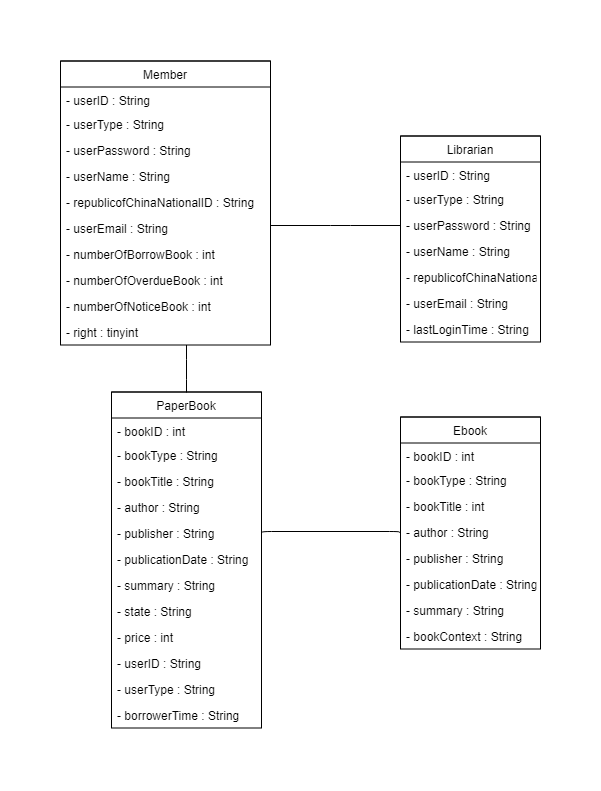


1. **Describe how you would denormalize the model that you created in question 9. Draw the new class diagram based on your suggested changes.**

* **Denormalization**

The third normalization has too many tables. First, it will lead to system’s running speed slower. Second, these tables don’t help us about manage data. So, we denormalize in order to improve system’s running speed.

* **Class Diagram**

****

1. **Examine the model that you created in question 10. Develop the inter-file clustering and index strategies. Describe how your clustering strategy will improve the performance of the database. List possible indices you would recommend and describe the reasons.**

We will create two indexes between Book Type and author. Because if we know which kind of book we need, then we can use index for us to search more quickly. Like, I want to find Harry Potter, if I don’t know which kind of book it is. Then system needs to find every book in system. It will waste too much time. But if we know it is E-book, then the system can just find E-book part. Saving our time and more efficient. So, such as index Book Type, we will create another index about author. Because in some situation, people just know the book’s author. So if we make an index about author. It will take less time than search every book data.

|  |
| --- |
| Book |
| BookID | BookType | BookTitle | Author | Publisher | PublicationDate | Summary |
| 1 | PaperBook | I'm No.1 | oneno | Red | 107/01 | this book makes you become no.1 |
| 2 | PaperBook | Second not bad | twowt | BANANA | 107/02 | don't always want be no.1, I'll tell you advantage of second |
| 1 | Ebook | Third what ever | threerht | Zebra | 107/03 | third means you just behind two people, don't think to much |
| 3 | PaperBook | Forth you better relex | fouruof | OK | 107/04 | If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax. |
| 4 | PaperBook | No fifth | fivevif | BANANA | 107/01 | No one care about fifth, just practice harder. |
| 5 | PaperBook | Sixth give up | sixis | BANANA | 107/03 | six is not a lucky number, give up will give you happy life |
| 2 | Ebook | Lucky seven | oneno | Zebra | 107/05 | you must a luck guy to get this number, let me aupluse to you |
| 6 | PaperBook | Super eight | sixis | OK | 107/02 | lying eight is unlimit, you are superman |
| 7 | PaperBook | number nine | ninenin | Red | 107/03 | no no no no, just nine just a number, No MORE |
| 8 | PaperBook | Top ten fact | seveneves | BANANA | 107/01 | Fact no.1：If you want to know, borrow me first |
| 9 | PaperBook | Uncountable | twowt | OK | 107/01 | I can't count anymore, don't ask me the number behind ten. |

|  |  |
| --- | --- |
| Book Type Index | |
| Book Type | Pointer |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| PaperBook |  |
| Ebook |  |
| Ebook |  |

|  |
| --- |
| Book |
| BookID | BookType | BookTitle | Author | Publisher | PublicationDate | Summary |
| 1 | PaperBook | I'm No.1 | oneno | Red | 107/01 | this book makes you become no.1 |
| 2 | PaperBook | Second not bad | twowt | BANANA | 107/02 | don't always want be no.1, I'll tell you advantage of second |
| 1 | Ebook | Third what ever | threerht | Zebra | 107/03 | third means you just behind two people, don't think to much |
| 3 | PaperBook | Forth you better relex | fouruof | OK | 107/04 | If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax. |
| 4 | PaperBook | No fifth | fivevif | BANANA | 107/01 | No one care about fifth, just practice harder. |
| 5 | PaperBook | Sixth give up | sixis | BANANA | 107/03 | six is not a lucky number, give up will give you happy life |
| 2 | Ebook | Lucky seven | oneno | Zebra | 107/05 | you must a luck guy to get this number, let me aupluse to you |
| 6 | PaperBook | Super eight | sixis | OK | 107/02 | lying eight is unlimit, you are superman |
| 7 | PaperBook | number nine | ninenin | Red | 107/03 | no no no no, just nine just a number, No MORE |
| 8 | PaperBook | Top ten fact | seveneves | BANANA | 107/01 | Fact no.1：If you want to know, borrow me first |
| 9 | PaperBook | Uncountable | twowt | OK | 107/01 | I can't count anymore, don't ask me the number behind ten. |

|  |  |
| --- | --- |
| Publisher Type Index | |
| Publisher Type | Pointer |
| BANANA |  |
| BANANA |  |
| BANANA |  |
| BANANA |  |
| Red |  |
| Red |  |
| Zebra |  |
| Zebra |  |
| OK |  |
| OK |  |
| OK |  |

**Participate In Assignments**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Participate | Responsibility |
| A10523006 | Maggie | 100% | Question3  Question4  Question6 |
| A10523049 | Peggy | 100% | Question11  PPT |
| B10423003 | Kurumi | 100% | Word  Question1  Question8  Question10 |
| B10423029 | Bean | 0% |  |
| B10523020 | Kendy | 100% | Java Code  Question1  Question2  Question6 |
| B10523030 | Jerry | 100% | Question7 |
| B10523053 | Lynn | 100% | Question1  Question5  Question9 |
| M10723001 | Joe | 100% | Java Code  Question2  Question9  Question11 |