Fall 2024

**Course Project Technical Report**

Version 0.4

Fall 2024

COMP 2714-BCIT

Course Project Technical Report

**Team (Project) Name:**

Project-24

**Team Members:**

|  |  |  |
| --- | --- | --- |
|  | Student Name | Student-ID |
| 1 | Jiarui Xing | A01354731 |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |

[About the Project 3](#_Toc145930805)

[Project Goal 3](#_Toc145930806)

[Project Evaluation 3](#_Toc145930807)

[Project Timeline 3](#_Toc145930808)

[Milestone1 4](#_Toc145930809)

[Milestone1-Task1: Universe of Discourse (Mini-World) Description 4](#_Toc145930810)

[Milestone1-Task2: Conceptual Design using ER/EER Diagram 4](#_Toc145930811)

[Milestone1-Task3: Conceptual Design using UML Notation [Optional but Recommended] 4](#_Toc145930812)

[Milestone1-Task4: Defining functional dependencies in your mini-world 4](#_Toc145930813)

[Milestone2 6](#_Toc145930814)

[Milestone2-Task1: Updated Universe of Discourse (Mini-World) Description 6](#_Toc145930815)

[Milestone2-Task2: Updated Conceptual Design 6](#_Toc145930816)

[Milestone2-Task3: Updating functional dependencies in your mini-world 6](#_Toc145930817)

[Milestone2-Task4: Creating the relational model of your conceptual model 6](#_Toc145930818)

[Milestone3 7](#_Toc145930819)

[Milestone3-Task1: Update your conceptual and relational models 7](#_Toc145930820)

[Milestone3-Task1: Create your database schema 7](#_Toc145930821)

[Milestone3-Task2: Populate your tables with some sample data 7](#_Toc145930822)

[Milestone3-Task3: Write SQL Statement 7](#_Toc145930823)

[Milestone4 9](#_Toc145930824)

[Milestone4-Task1:Identify full, partial and transitive functional dependencies in your design 9](#_Toc145930825)

[Milestone4-Task2: Highest Normal Form 9](#_Toc145930826)

[Milestone4-Task3: Converting to 3NF 10](#_Toc145930827)

[Milestone4-Task4: Converting to BCNF 10](#_Toc145930828)

# About the Project

## Project Goal

The project activity aims to provide this opportunity to create a database system from the beginning to the database development. In this project, you will:

1. Come up with a universe of discourse and a corresponding conceptual model
2. Convert your conceptual model to a relational model
3. Use a SQL product (such as MySQL) to create your database, populate it with some data and write some DML statements
4. Normalize your database schema

The project could be done in groups of up to 4 people. Depending on the number of people in the group the scope of the project would change.

## Project Evaluation

Each Milestone will be evaluated independently.

Once you join a group, it is expected to stay in the group until the end of the project.

The project final grade for each individual will be based on the following metrics:

1. The completion of the Milestones 1-4 on time and completeness of this technical report
2. Evaluation of teammates in a group

## Project Timeline

|  |  |
| --- | --- |
| **Milestones** | **Due date** |
| Milestone 1 | Please refer to Learning Hub |
| Milestone 2 | Please refer to Learning Hub |
| Milestone 3 | Please refer to Learning Hub |
| Milestone 4 | Please refer to Learning Hub |

# Milestone1

* You need to describe a mini world.
* You can come up with a new idea for the mini-world or describe an existing application.
* A good mini-world is one with a conceptual model including all the topics we have studied in Module 1:
  + Entities, Weak entities, Total and partial participation, Classes and sub-classes, composite attributes, derived attributes, super/subclasses, …
* The conceptual model in 2 formats (ER/EER) and UML Class Notation
  + UML Notation is Optional (for practice only).

**Note:** The scope of the project is adjusted based on how many people work on a project in a group.

If you work individually (in groups of 1) on this project:

* Then, between 8-10+ entity types are expected in your conceptual model.

If you work in a group of 2 on the project:

* Then, it is expected that there will be between 12 and 15+ entity types in your conceptual model.

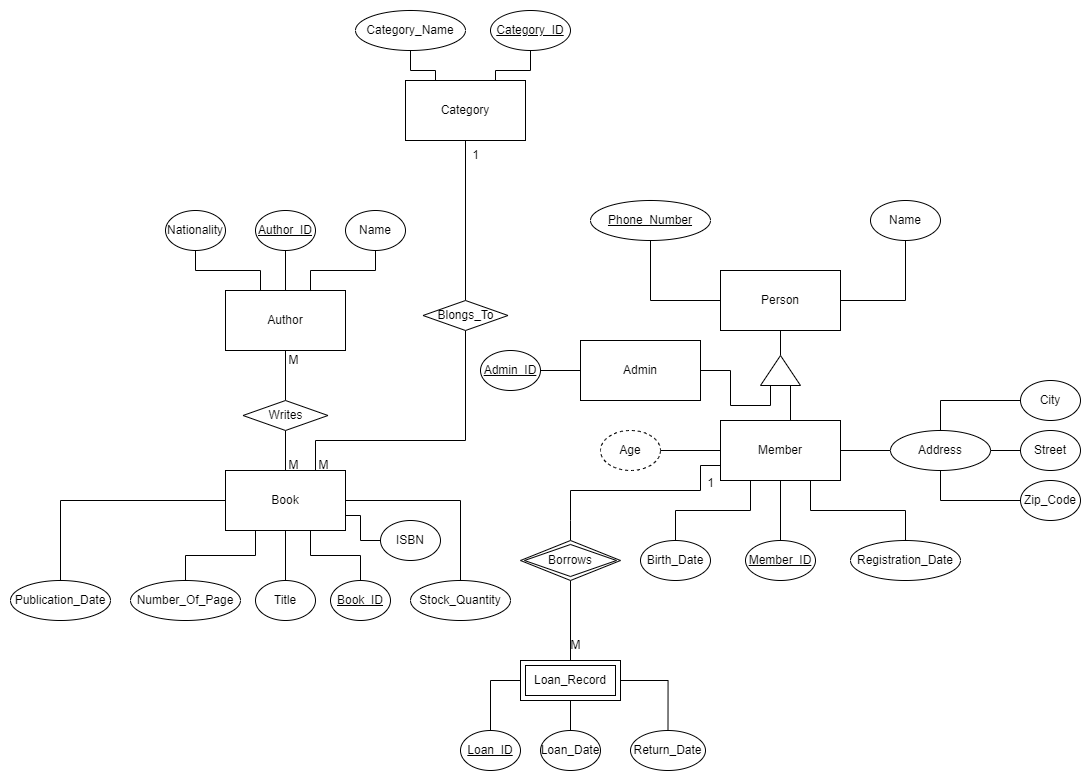
If you work in a group of 3/4 on the project:

* Then, between 18-20+ entity types are expected in your conceptual model.

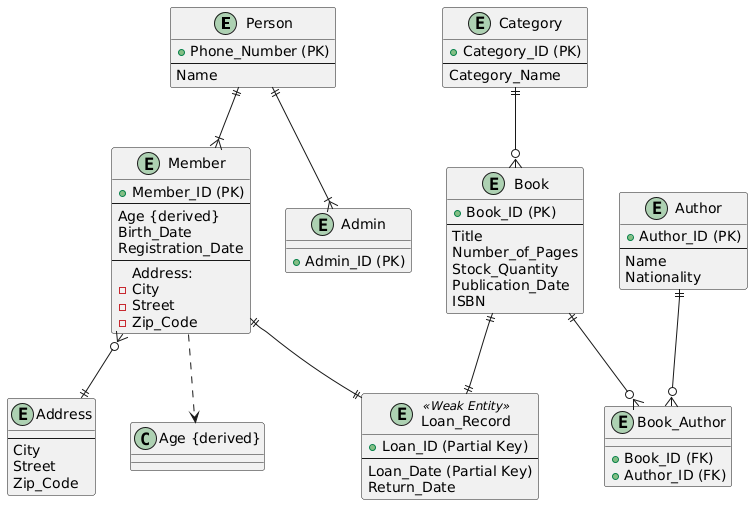
## Milestone1-Task1: Universe of Discourse (Mini-World) Description

The library management system is designed to efficiently manage various library resources and services efficiently, ensuring the smooth operation of functions such as book borrowing, returning, inventory management, and member management. The primary users of the system include library administrators and members.

## Milestone1-Task2: Conceptual Design using ER/EER Diagram

****

## Milestone1-Task3: Conceptual Design using UML Notation [Optional but Recommended]



## Milestone1-Task4: Defining functional dependencies in your mini-world

If you work individually (in groups of 1) on this project:

* Then, it is expected to have between 4+ functional dependencies in your mini-world.

If you work in a group of 2 on the project:

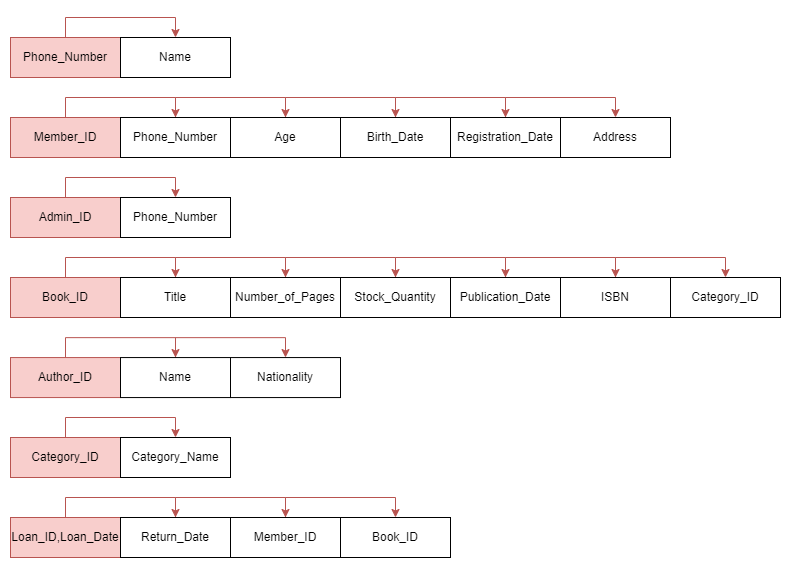
* Then, it is expected to have between 6+ functional dependencies in your mini-world.

If you work in a group of 3/4 on the project:

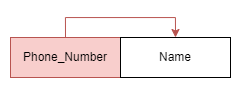
* Then, it is expected to have between 7+ functional dependencies in your mini-world.

**List all functional dependencies here. Make sure to include visuals as discussed during the lecture:**

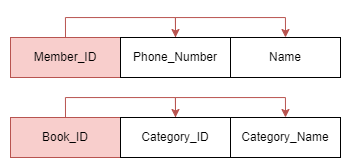
1. Primary-Key Functional Dependencies: List them here:



1. Partial Functional Dependencies: List them here



1. Transitive Functional Dependencies: List them here



# Milestone2

In Milestone 2, you are going to continue working on what you have done in Milestone1.

## Milestone2-Task1: Updated Universe of Discourse (Mini-World) Description

* [Optional]: If your mini-world needs to be updated for any reasons, you have the chance to update it here. Please notice, if the original version of the mini-World is not comprehensive enough (having different types of entities and relationships and … and enough number of entities), you need to update the original mini-World and submit it in this Milestone.
* The Milestone2-Task1 is not going to be graded, yet an update is a must if needed based on the comment above. Please also see Task 3 below.

**[Write your answer here]**

## Milestone2-Task2: Updated Conceptual Design

* If an update to the Conceptual design is needed, please do it and add it here. (Particularly If you have to update your original mini-world in Task1 above)

**[Write your answer here]**

## Milestone2-Task3: Updating functional dependencies in your mini-world

If you work individually (groups of 1) on this project:

* Then, it is expected to have between 4+ functional dependencies in your mini-world.

If you work in a group of 2 on the project:

* Then, it is expected to have between 6+ functional dependencies in your mini-world.

If you work in a group of 3/4 on the project:

* Then, it is expected to have between 7+ functional dependencies in your mini-world.

**Note**: If you cannot extract enough functional dependencies in your mini-world, you will need to update it and submit it under Milestone2-Task1.

**[Write your answer here]**

## Milestone2-Task4: Creating the relational model of your conceptual model

* You need to create the relational models following the steps we defined in the lecture.

**Relations**  
Person: [Phone\_Number, Name]  
Member: [Member\_ID, Phone\_Number, Birth\_Date, Registration\_Date, City, Street, Zip\_Code]  
Admin: [Admin\_ID, Phone\_Number]  
Book: [Book\_ID, Title, Number\_of\_Pages, Stock\_Quantity, Publication\_Date, ISBN, Category\_ID]  
Author: [Author\_ID, Name, Nationality]  
Category: [Category\_ID, Category\_Name]  
Loan\_Record: [Member\_ID, Book\_ID, Loan\_Date, Return\_Date]  
Book\_Author: [Book\_ID, Author\_ID]

**Foreign Keys**  
Member.Phone\_Number references Person.Phone\_Number  
Admin.Phone\_Number references Person.Phone\_Number  
Book.Category\_ID references Category.Category\_ID  
Loan\_Record.Member\_ID references Member.Member\_ID  
Loan\_Record.Book\_ID references Book.Book\_ID  
Book\_Author.Book\_ID references Book.Book\_ID  
Book\_Author.Author\_ID references Author.Author\_ID

# Milestone3

In Milestone 3, In Milestone 3 you are doing to work with the database that you have designed.

**Note:** By the end of Milestone 2, it is expected that you have designed your relational model (database).

## Milestone3-Task1: Update your conceptual and relational models

If you have received any feedback from the evaluator of Milestone 1 and 2, that you need to make some changes in your conceptual and relation models, you need to do it now and insert the updated answer:

## Milestone3-Task1: Create your database schema

* Please use the SQL DDL statements to create your database schema.
* Provide a name for each table.
* Include all attributes
* Define PK and FK.
* Please add a snapshot of the statement you have written and the results
* CREATE DATABASE *LMS*;  
    
  USE *LMS*;  
  CREATE TABLE *Person* (  
   Phone\_Number VARCHAR(15) PRIMARY KEY,  
   Name VARCHAR(50) NOT NULL  
  );  
    
  CREATE TABLE *Member* (  
   Member\_ID INT PRIMARY KEY,  
   Phone\_Number VARCHAR(15) NOT NULL,  
   Birth\_Date DATE NOT NULL,  
   Registration\_Date DATE NOT NULL,  
   City VARCHAR(50),  
   Street VARCHAR(100),  
   Zip\_Code VARCHAR(10),  
   FOREIGN KEY (Phone\_Number) REFERENCES *Person*(Phone\_Number)  
  );  
    
  CREATE TABLE *Admin* (  
   Admin\_ID INT PRIMARY KEY,  
   Phone\_Number VARCHAR(15) NOT NULL,  
   FOREIGN KEY (Phone\_Number) REFERENCES *Person*(Phone\_Number)  
  );  
    
  CREATE TABLE *Category* (  
   Category\_ID INT PRIMARY KEY,  
   Category\_Name VARCHAR(50) NOT NULL  
  );  
    
  CREATE TABLE *Book* (  
   Book\_ID INT PRIMARY KEY,  
   Title VARCHAR(100) NOT NULL,  
   Number\_of\_Pages INT,  
   Stock\_Quantity INT,  
   Publication\_Date DATE,  
   ISBN VARCHAR(20),  
   Category\_ID INT,  
   FOREIGN KEY (Category\_ID) REFERENCES *Category*(Category\_ID)  
  );  
    
  CREATE TABLE *Author* (  
   Author\_ID INT PRIMARY KEY,  
   Name VARCHAR(50) NOT NULL,  
   Nationality VARCHAR(50)  
  );  
    
  CREATE TABLE *Loan\_Record* (  
   Member\_ID INT,  
   Book\_ID INT,  
   Loan\_Date DATE,  
   Return\_Date DATE,  
   PRIMARY KEY (Member\_ID, Book\_ID, Loan\_Date),  
   FOREIGN KEY (Member\_ID) REFERENCES *Member*(Member\_ID),  
   FOREIGN KEY (Book\_ID) REFERENCES *Book*(Book\_ID)  
  );  
    
  CREATE TABLE *Book\_Author* (  
   Book\_ID INT,  
   Author\_ID INT,  
   PRIMARY KEY (Book\_ID, Author\_ID),  
   FOREIGN KEY (Book\_ID) REFERENCES *Book*(Book\_ID),  
   FOREIGN KEY (Author\_ID) REFERENCES *Author*(Author\_ID)  
  );

## 图形用户界面, 文本, 应用程序 描述已自动生成

## Milestone3-Task2: Populate your tables with some sample data

* Insert some sample data to your tables
* The volume of inserted data is not important. Just insert enough amount of data that makes sense for your project.
* Please add a snapshot of the tables with some data in them

INSERT INTO *Person* VALUES ('1234567890', 'Alice');  
INSERT INTO *Person* VALUES ('0987654321', 'Bob');  
INSERT INTO *Person* VALUES ('1122334455', 'Charlie');  
  
INSERT INTO *Member* VALUES (1, '1234567890', '1990-01-01', '2023-01-01', 'New York', '5th Ave', '10001');  
INSERT INTO *Member* VALUES (2, '0987654321', '1985-05-05', '2023-05-05', 'Los Angeles', 'Sunset Blvd', '90001');  
INSERT INTO *Member* VALUES (3, '1122334455', '1992-07-07', '2023-07-07', 'Chicago', 'Lake Shore Dr', '60601');  
  
INSERT INTO *Admin* VALUES (101, '1234567890');  
INSERT INTO *Admin* VALUES (102, '0987654321');  
  
INSERT INTO *Category* VALUES (1, 'Computer Science');  
INSERT INTO *Category* VALUES (2, 'Literature');  
INSERT INTO *Category* VALUES (3, 'History');  
  
INSERT INTO *Book* VALUES (201, 'Database Systems', 600, 10, '2020-10-10', '123-456-789', 1);  
INSERT INTO *Book* VALUES (202, 'Introduction to Algorithms', 1312, 5, '2019-07-15', '987-654-321', 1);  
INSERT INTO *Book* VALUES (203, 'Pride and Prejudice', 432, 7, '1813-01-28', '111-222-333', 2);  
  
INSERT INTO *Author* VALUES (301, 'Edgar Codd', 'British');  
INSERT INTO *Author* VALUES (302, 'Thomas H. Cormen', 'American');  
INSERT INTO *Author* VALUES (303, 'Jane Austen', 'British');  
  
INSERT INTO *Loan\_Record* VALUES (1, 201, '2024-11-25', '2024-12-05');  
INSERT INTO *Loan\_Record* VALUES (2, 202, '2024-11-26', NULL);  
INSERT INTO *Loan\_Record* VALUES (3, 203, '2024-11-27', '2024-12-07');  
  
INSERT INTO *Book\_Author* VALUES (201, 301);  
INSERT INTO *Book\_Author* VALUES (202, 302);  
INSERT INTO *Book\_Author* VALUES (203, 303);  
  
INSERT INTO *Loan\_Record* VALUES (1, 202, '2024-11-28', NULL);  
  
INSERT INTO *Book* VALUES (204, 'The Great Gatsby', 218, 4, '1925-04-10', '444-555-666', 2);  
  
INSERT INTO *Person* VALUES ('2233445566', 'David');  
INSERT INTO *Member* VALUES (4, '2233445566', '1995-03-03', '2024-01-01', 'Houston', 'Main St', '77001');  
  
SELECT b.Title, lr.Loan\_Date  
FROM *Loan\_Record* lr  
JOIN *Book* b ON lr.Book\_ID = b.Book\_ID  
WHERE lr.Member\_ID = 1 AND lr.Return\_Date IS NULL;  
  
SELECT b.Title  
FROM *Book* b  
LEFT JOIN *Loan\_Record* lr ON b.Book\_ID = lr.Book\_ID  
WHERE lr.Book\_ID IS NULL;  
  
DELETE FROM *Member*WHERE Member\_ID = 4  
AND Member\_ID NOT IN (SELECT Member\_ID FROM *Loan\_Record*);  
  
SELECT *\** FROM *Member* WHERE Member\_ID = 4;

图形用户界面, 文本, 应用程序

描述已自动生成

电脑屏幕截图

描述已自动生成

## Milestone3-Task3: Write SQL Statement

* You need to define several use-cases and write their corresponding queries.
* At least 10 queries should be written
* For group of 2 or more people, in addition to the first 10 queries, for each group member you need to add extra 3 use-cases.
* Write the SQL Statements, run them and take a snapshot from the results and insert them here.
* So the following items are expected:

1. The usecase
2. The SQL query
3. The screenshot of the data

Example:

1. Use case: As an admin, I need to know the name and SSN of the managers of all department.
2. SQL Statement: Please see below
3. Snapshot: Please see below

Graphical user interface, application

Description automatically generated

1. **As an admin, I need to know the name and Phone Number of all members.**

SELECT m.Member\_ID, p.Name, p.Phone\_Number  
FROM *Member* m  
JOIN *Person* p ON m.Phone\_Number = p.Phone\_Number;

图形用户界面, 文本, 应用程序

描述已自动生成

1. **As an admin, I need to retrieve the list of all books along with their categories.**

SELECT b.Title, c.Category\_Name  
FROM *Book* b  
JOIN *Category* c ON b.Category\_ID = c.Category\_ID;

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

1. **As an admin, I need to retrieve the list of all books along with their categories.**

SELECT b.Title, lr.Loan\_Date  
FROM *Loan\_Record* lr  
JOIN *Book* b ON lr.Book\_ID = b.Book\_ID  
WHERE lr.Member\_ID = 1 AND lr.Return\_Date IS NULL;

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

1. **As an admin, I need to find out how many books each author has written.**

SELECT a.Name AS Author\_Name, *COUNT*(ba.Book\_ID) AS Books\_Written  
FROM *Author* a  
JOIN *Book\_Author* ba ON a.Author\_ID = ba.Author\_ID  
GROUP BY a.Name;

图形用户界面, 文本, 应用程序

描述已自动生成

1. **As a librarian, I want to view all loan records including member names and book titles.**

SELECT p.Name AS Member\_Name, b.Title, lr.Loan\_Date, lr.Return\_Date  
FROM *Loan\_Record* lr  
JOIN *Member* m ON lr.Member\_ID = m.Member\_ID  
JOIN *Person* p ON m.Phone\_Number = p.Phone\_Number  
JOIN *Book* b ON lr.Book\_ID = b.Book\_ID;

图形用户界面, 应用程序

描述已自动生成

1. **As an admin, I need to list all books with stock quantity below 5.**

SELECT Title, Stock\_Quantity  
FROM *Book*WHERE Stock\_Quantity < 5;

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

1. **As a manager, I want to see the number of members in each city.**

SELECT City, *COUNT*(Member\_ID) AS Member\_Count  
FROM *Member*GROUP BY City;

图形用户界面, 文本, 应用程序

描述已自动生成

1. **As an admin, I need to find all books that have never been borrowed.**

SELECT b.Title  
FROM *Book* b  
LEFT JOIN *Loan\_Record* lr ON b.Book\_ID = lr.Book\_ID  
WHERE lr.Book\_ID IS NULL;

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

1. **As a librarian, I want to update the stock quantity of a specific book after a new shipment arrives.**

UPDATE *Book*SET Stock\_Quantity = Stock\_Quantity + 10  
WHERE Book\_ID = 201;

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

1. **As an admin, I need to delete a member who has not borrowed any books.**

DELETE FROM *Member*WHERE Member\_ID = 3  
AND Member\_ID NOT IN (SELECT Member\_ID FROM *Loan\_Record*);

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

# Milestone4

In Milestone 4, you are going to normalize the database (relational model) you have developed in Milestone 2.

**Note:** It is expected in this Milestone to change the relational model you have completed in Milestone 2 and consequently the database you have developed in Milestone 3. While it is expected to normalize your database schema in this milestone, it is not expected you to re-create your database and update the sql statements you have developed in module 3.

## Milestone4-Task1:Identify full, partial and transitive functional dependencies in your design

* In Module 4, we learned about full, partial and transitive functional dependencies and we discussed how normalization process deals with such functional dependencies.
* The Milestone4-Task1: List all functional dependencies you have identified in Milestone 2. For each functional dependency identify whether it is full, partial or transitive functional dependencies and briefly explain why. Complete the following table. (Expand the table as needed)

|  |  |
| --- | --- |
| Functional Dependency | **Phone\_Number → Name**  Full  Phone\_Number is PK and uniquely determines Name. |
| Partial, full or transitive? And why |
| Functional Dependency | **Member\_ID → Phone\_Number, Birth\_Date, Registration\_Date, City, Street, Zip\_Code**  Full  Member\_ID is PK and uniquely determines all attributes. |
| Partial, full or transitive? And why |
| Functional Dependency | **Zip\_Code → City**  Transitive  Zip\_Code determines City, creating a transitive dependency from Member\_ID. |
| Partial, full or transitive? And why |
| Functional Dependency | **Admin\_ID → Phone\_Number**  Full  Admin\_ID is PK, and uniquely determines Phone\_Number. |
| Partial, full or transitive? And why |
| Functional Dependency | **Category\_ID → Category\_Name**  Full  Category\_ID is PK and uniquely determines Category\_Name. |
| Partial, full or transitive? And why |
| Functional Dependency | **Book\_ID → Title, Number\_of\_Pages, Stock\_Quantity, Publication\_Date, ISBN, Category\_ID**  Full  Book\_ID is PK and uniquely determines all attributes. |
| Partial, full or transitive? And why |
| Functional Dependency | **ISBN → Title, Number\_of\_Pages, Publication\_Date, Category\_ID**  Partial  ISBN is not PK but determines some attributes. |
| Partial, full or transitive? And why |
| Functional Dependency | **Book\_ID → ISBN → Title, Number\_of\_Pages, Publication\_Date, Category\_ID**  Transitive  Book\_ID determines ISBN, which in turn determines other attributes. |
| Partial, full or transitive? And why |
| Functional Dependency | **Author\_ID → Name, Nationality**  Full  Author\_ID is PK and uniquely determines all attributes. |
| Partial, full or transitive? And why |
| Functional Dependency | **(Member\_ID, Book\_ID, Loan\_Date) → Return\_Date**  Full  The composite PK (Member\_ID, Book\_ID, Loan\_Date) uniquely determines Return\_Date. |
| Partial, full or transitive? And why |

## Milestone4-Task2: Highest Normal Form

* Take your relational models and test them against the Normalization tests and complete this section:

Relation NOT in 1NF: (List all relations that are NOT even in 1NF, if any):

-

Relation with 1NF as highest normal form: (List all relations that are in 1NF but not 2NF)

-

Relation with 2NF as highest normal form: (List all relations that are in 2NF but not 3NF)

* Member
* Book

Relation with 3NF as highest normal form: (List all relations that are in 3NF but not BCNF)

-

Relation with BCNF as highest normal form:

* Person
* Admin
* Category
* Author
* Loan\_Record
* Book\_Author

## Milestone4-Task3: Converting to 3NF

In this task you will normalize all your relations to 3NF and BCNF. If a relation is already in BCNF or 3NF, no change is needed.

-

## Milestone4-Task4: Converting to BCNF

In this task you will normalize all your relations to BCNF. If a relation is already in BCNF, no change is needed.

* Check whether there are any functional dependencies that are not preserved as a result of normalization to BCNF.

*-- Member*CREATE TABLE *Zip\_Code\_Info* (  
 Zip\_Code VARCHAR(10) PRIMARY KEY,  
 City VARCHAR(50) NOT NULL  
);  
  
INSERT INTO *Zip\_Code\_Info* (Zip\_Code, City)  
SELECT DISTINCT Zip\_Code, City  
FROM *Member*;  
  
ALTER TABLE *Member* DROP COLUMN City;  
  
ALTER TABLE *Member*ADD CONSTRAINT FK\_Member\_Zip\_Code FOREIGN KEY (Zip\_Code) REFERENCES *Zip\_Code\_Info*(Zip\_Code);  
  
*-- Book*CREATE TABLE *Book\_Details* (  
 ISBN VARCHAR(20) PRIMARY KEY,  
 Title VARCHAR(100) NOT NULL,  
 Number\_of\_Pages INT,  
 Publication\_Date DATE,  
 Category\_ID INT,  
 FOREIGN KEY (Category\_ID) REFERENCES *Category*(Category\_ID)  
);  
  
INSERT INTO *Book\_Details* (ISBN, Title, Number\_of\_Pages, Publication\_Date, Category\_ID)  
SELECT DISTINCT ISBN, Title, Number\_of\_Pages, Publication\_Date, Category\_ID  
FROM *Book*;  
  
ALTER TABLE *Book* DROP FOREIGN KEY *book\_ibfk\_1*;  
  
ALTER TABLE *Book*DROP COLUMN Title,  
DROP COLUMN Number\_of\_Pages,  
DROP COLUMN Publication\_Date,  
DROP COLUMN Category\_ID;  
  
ALTER TABLE *Book*ADD CONSTRAINT FK\_Book\_ISBN FOREIGN KEY (ISBN) REFERENCES *Book\_Details*(ISBN);

图形用户界面, 文本, 应用程序

描述已自动生成