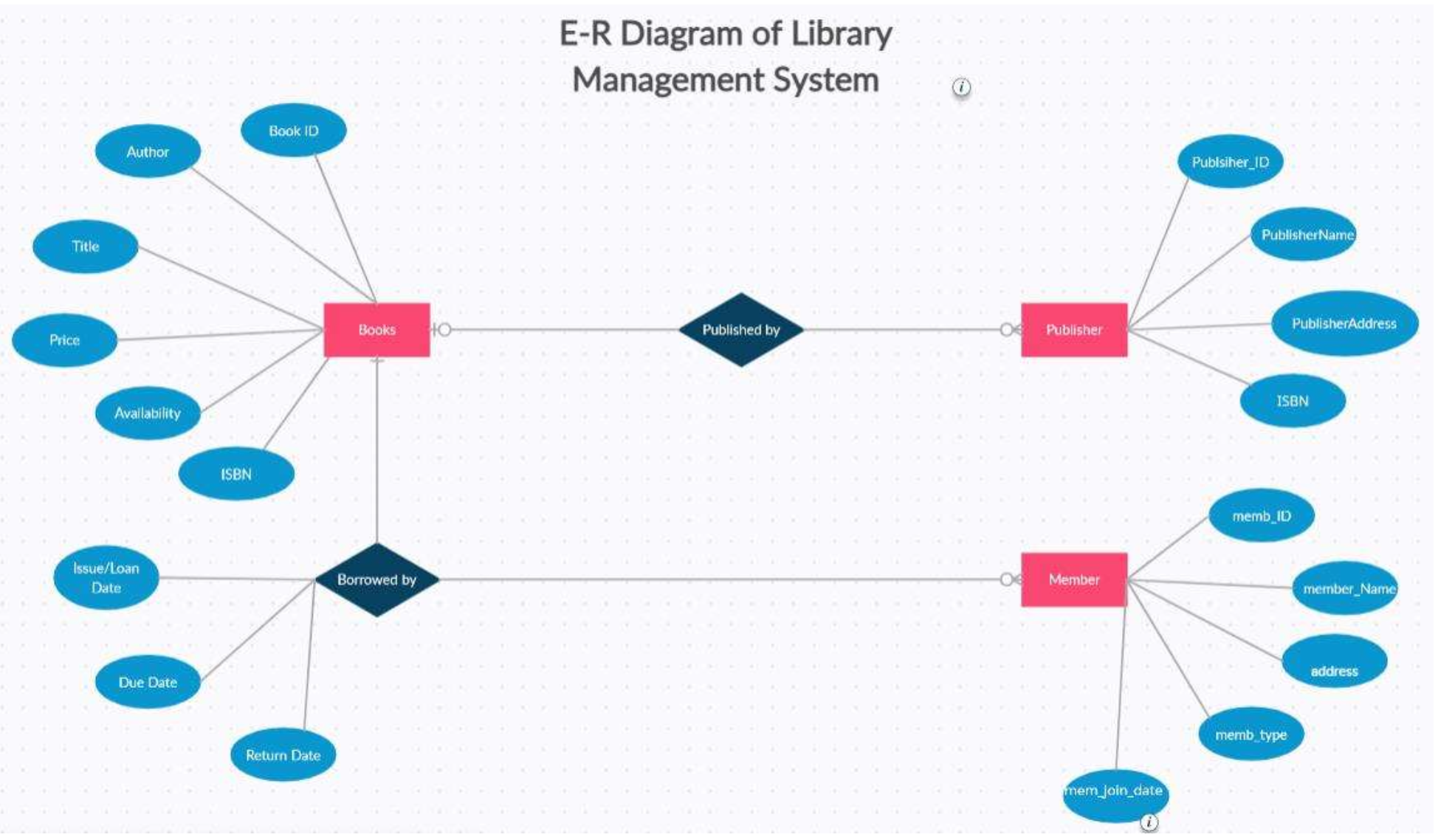
Luis Martins

National College of Ireland

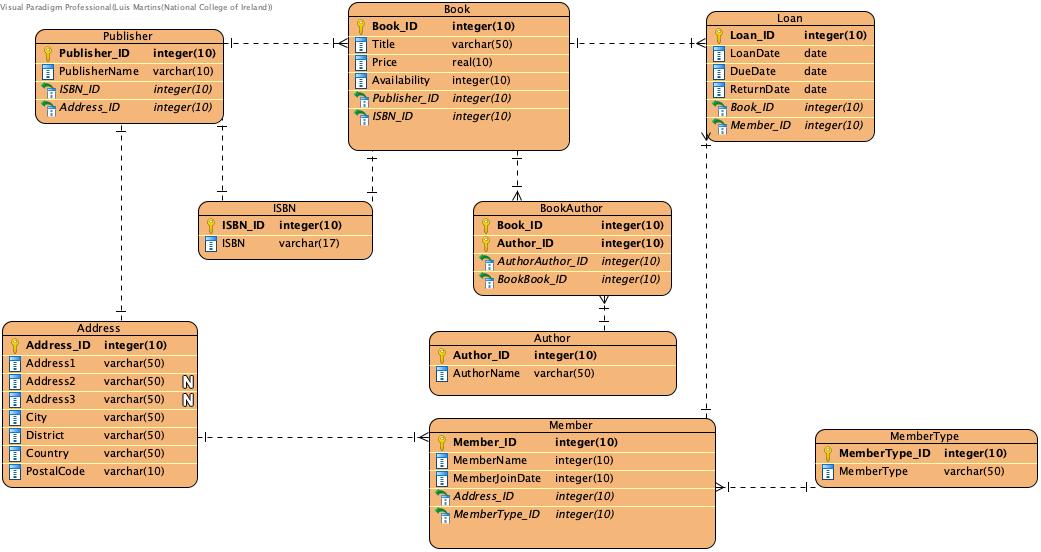
HDWD\_SEP23OL

Databases taba





# Transform the conceptual design (ER diagram) into a relational model by converting the entities and relationships into their appropriate tables. Show if your tables are normalized using 1st, 2nd, and 3rd normal form.



All tables are all normalized to 3NF.

1NF:

* Contain only atomic values (values that cannot be divided/single values)
* No repeating groups

2NF:

* Meet 1NF
* No Partial Dependencies: Non-key attributes (columns that are not part of the primary key) are fully dependent on the entire primary key, not just a part of it. This is achieved in the schema as all tables have single-attribute primary keys, eliminating the possibility of partial dependencies.

3NF:

* Meet 1NF & 2NF
* No Transitive Dependencies: Non-key attributes do not depend on other non-key attributes. They depend only on the primary key. This is also satisfied in the schema as the tables are structured to avoid transitive dependencies. For example, all the attributes in the “Books” table directly depend on the primary key "Book\_ID", rather than depending on each other.

## Create a database called LMS and convert all the resulting logical tables from question 1 into a physical database design using DDL. Choose the appropriate datatype, primary and foreign keys for the attributes. Fill your table with some data of your choice (you can use Mockaroo to create the data). Provide detailed assumptions for any of your design decisions.

## DDL

**CREATE TABLE *Book* (**

**Book\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**Title varchar(50) NOT NULL,**

**Price real(10) NOT NULL,**

**Availability integer(10) NOT NULL,**

**Publisher\_ID integer(10) NOT NULL,**

**ISBN\_ID integer(10) NOT NULL,**

**FOREIGN KEY(Publisher\_ID) REFERENCES Publisher(Publisher\_ID),**

**FOREIGN KEY(ISBN\_ID) REFERENCES ISBN(ISBN\_ID)**

**);**

**CREATE TABLE *BookAuthor* (**

**Book\_ID INTEGER NOT NULL,**

**Author\_ID INTEGER NOT NULL,**

**PRIMARY KEY (Book\_ID, Author\_ID),**

**FOREIGN KEY (Book\_ID) REFERENCES Book(Book\_ID),**

**FOREIGN KEY (Author\_ID) REFERENCES Author(Author\_ID)**

**);**

**CREATE TABLE *Publisher* (**

**Publisher\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**PublisherName varchar(10) NOT NULL,**

**ISBN\_ID integer(10) NOT NULL,**

**Address\_ID integer(10) NOT NULL,**

**FOREIGN KEY(Address\_ID) REFERENCES Address(Address\_ID),**

**FOREIGN KEY(ISBN\_ID) REFERENCES ISBN(ISBN\_ID)**

**);**

**CREATE TABLE *Member* (**

**Member\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**MemberName varchar(50) NOT NULL, *-- Changed from integer to varchar***

**MemberJoinDate date NOT NULL, *-- Changed from integer to date***

**Address\_ID integer(10) NOT NULL,**

**MemberType\_ID integer(10) NOT NULL,**

**FOREIGN KEY(Address\_ID) REFERENCES Address(Address\_ID),**

**FOREIGN KEY(MemberType\_ID) REFERENCES MemberType(MemberType\_ID)**

**);**

**CREATE TABLE *Loan* (**

**Loan\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**LoanDate date NOT NULL,**

**DueDate date NOT NULL,**

**ReturnDate date NOT NULL,**

**Book\_ID integer(10) NOT NULL,**

**Member\_ID integer(10) NOT NULL,**

**FOREIGN KEY(Book\_ID) REFERENCES Book(Book\_ID),**

**FOREIGN KEY(Member\_ID) REFERENCES Member(Member\_ID)**

**);**

**CREATE TABLE *Address* (**

**Address\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**Address1 varchar(50) NOT NULL,**

**Address2 varchar(50),**

**Address3 varchar(50),**

**City varchar(50) NOT NULL,**

**District varchar(50) NOT NULL,**

**Country varchar(50) NOT NULL,**

**PostalCode varchar(10) NOT NULL**

**);**

**CREATE TABLE *ISBN* (**

**ISBN\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**ISBN varchar(17) NOT NULL**

**);**

**CREATE TABLE *MemberType* (**

**MemberType\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**MemberType varchar(50) NOT NULL**

**);**

**CREATE TABLE *Author* (**

**Author\_ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,**

**AuthorName varchar(50) NOT NULL**

**);**

## DML

***-- Insert data into Address table***

**INSERT INTO Address (Address1, Address2, Address3, City, District, Country, PostalCode)**

**VALUES ('123 Main Street', 'Apt 4', NULL, 'Cityville', 'District A', 'Country X', '12345'),**

**('456 Oak Avenue', NULL, NULL, 'Townsville', 'District B', 'Country Y', '56789');**

***-- Insert data into ISBN table***

**INSERT INTO ISBN (ISBN)**

**VALUES ('978-1-123456-78-9'), ('978-2-234567-89-0');**

***-- Insert data into MemberType table***

**INSERT INTO MemberType (MemberType)**

**VALUES ('Regular'), ('Premium');**

***-- Insert data into Author table***

**INSERT INTO Author (AuthorName)**

**VALUES ('John Doe'), ('Jane Smith');**

***-- Insert data into Publisher table***

**INSERT INTO Publisher (PublisherName, ISBN\_ID, Address\_ID)**

**VALUES ('ABC Publications', 1, 1), ('XYZ Press', 2, 2);**

***-- Insert data into Book table (excluding Author\_ID)***

**INSERT INTO Book (Title, Price, Availability, Publisher\_ID, ISBN\_ID)**

**VALUES ('Introduction to SQL', 29.99, 50, 1, 1),**

**('Data Science Essentials', 39.99, 25, 2, 2),**

**('The Art of Programming', 45.00, 30, 1, 3); *-- New book with multiple authors***

***-- Insert data into BookAuthor table to link books and authors***

**INSERT INTO BookAuthor (Book\_ID, Author\_ID)**

**VALUES (1, 1), *-- Introduction to SQL by John Doe***

**(2, 2), *-- Data Science Essentials by Jane Smith***

**(3, 1), *-- The Art of Programming by John Doe***

**(3, 2); *-- The Art of Programming by Jane Smith (multiple authors)***

***-- Insert data into Member table***

**INSERT INTO Member (MemberName, MemberJoinDate, Address\_ID, MemberType\_ID)**

**VALUES ('Alice Johnson', '2023-01-15', 1, 1),**

**('Bob Smith', '2022-05-20', 2, 2);**

***-- Insert data into Loan table***

**INSERT INTO Loan (LoanDate, DueDate, ReturnDate, Book\_ID, Member\_ID)**

**VALUES ('2023-02-01', '2023-02-15', '2023-02-10', 1, 1),**

**('2022-06-10', '2022-06-25', '2022-06-20', 2, 2);**

**Book:**

* **Book\_ID (INTEGER, PRIMARY KEY, AUTOINCREMENT):** Numeric ID ensures efficient referencing for unique book identification. Auto-incrementing simplifies data insertion.
* **Title (VARCHAR(50), NOT NULL):** Text string accommodates varying book title lengths, supporting diverse naming conventions.
* **Price (REAL(10), NOT NULL):** Decimal value accurately represents book prices, including potential fractional amounts (e.g., €19.99).
* **Availability (INTEGER(10), NOT NULL):** Whole number efficiently tracks available copies, aligning with the concept of countable inventory.
* **Publisher\_ID (INTEGER, NOT NULL, FOREIGN KEY REFERENCES Publisher(Publisher\_ID)):** Numeric ID facilitates efficient linking to publisher data, ensuring referential integrity.
* **ISBN\_ID (INTEGER(10), NOT NULL, FOREIGN KEY REFERENCES ISBN(ISBN\_ID)):** Numeric ID facilitates efficient linking to ISBN data, ensuring referential integrity.

**BookAuthor:**

* **Book\_ID (INTEGER, NOT NULL, FOREIGN KEY REFERENCES Book(Book\_ID)):** Numeric ID aligns with Book table for efficient relationship establishment.
* **Author\_ID (INTEGER, NOT NULL, FOREIGN KEY REFERENCES Author(Author\_ID)):** Numeric ID aligns with Author table, enabling efficient author association.
* **PRIMARY KEY (Book\_ID, Author\_ID):** Composite primary key using numeric IDs ensures uniqueness for each book-author combination.

**Publisher:**

* **Publisher\_ID (INTEGER, PRIMARY KEY, AUTOINCREMENT):** Numeric ID guarantees unique publisher identification and efficient referencing. Auto-incrementing simplifies data insertion.
* **PublisherName (VARCHAR(10), NOT NULL):** Text string accommodates varying publisher names, allowing for flexibility in naming conventions.
* **ISBN\_ID (INTEGER(10), NOT NULL, FOREIGN KEY REFERENCES ISBN(ISBN\_ID)):** Numeric ID aligns with ISBN table, enabling efficient linking to ISBN data.
* **Address\_ID (INTEGER(10), NOT NULL, FOREIGN KEY REFERENCES Address(Address\_ID)):** Numeric ID aligns with Address table, facilitating efficient connection to publisher address information.

**Member:**

* **Member\_ID (INTEGER, PRIMARY KEY, AUTOINCREMENT):** Numeric ID ensures unique member identification and efficient referencing. Auto-incrementing simplifies data insertion.
* **MemberName (VARCHAR(50), NOT NULL):** Text string accommodates varying name lengths and formats, respecting diverse naming conventions.
* **MemberJoinDate (DATE, NOT NULL):** Date format accurately captures and stores specific membership start dates, enabling temporal queries and analysis.
* **Address\_ID (INTEGER(10), NOT NULL, FOREIGN KEY REFERENCES Address(Address\_ID)):** Numeric ID aligns with Address table, efficiently linking members to their address information.
* **MemberType\_ID (INTEGER(10), NOT NULL, FOREIGN KEY REFERENCES MemberType(MemberType\_ID)):**Numeric ID aligns with MemberType table, enabling efficient categorisation of members for potential benefits or differentiated services.

**Loan:**

* **Loan\_ID (INTEGER, PRIMARY KEY, AUTOINCREMENT):** Numeric ID guarantees unique loan identification and efficient referencing. Auto-incrementing simplifies data insertion.
* **LoanDate (DATE, NOT NULL):** Date format accurately captures and stores specific loan dates, enabling temporal queries and analysis.
* **DueDate (DATE, NOT NULL):** Date format precisely specifies when a borrowed book is expected to be returned, facilitating overdue tracking and reminders.
* **ReturnDate (DATE, NOT NULL):** Date format accurately records the book's return date, enabling historical loan tracking and analysis.
* **Book\_ID (INTEGER, NOT NULL, FOREIGN KEY REFERENCES Book(Book\_ID)):** Numeric ID aligns with Book table, efficiently associating loans with specific borrowed books.
* **Member\_ID (INTEGER, NOT NULL, FOREIGN KEY REFERENCES Member(Member\_ID)):** Numeric ID aligns with Member table, efficiently linking loans to borrowing members.

**Address:**

* **Address\_ID (INTEGER, PRIMARY KEY, AUTOINCREMENT):** Numeric ID ensures unique address identification and efficient referencing. Auto-incrementing simplifies data insertion.
* **Address1 (VARCHAR(50), NOT NULL):** First line of the address, essential for location identification.
* **Address2 (VARCHAR(50)):** Optional second line for additional address details.
* **Address3 (VARCHAR(50)):** Optional third line for further address details.
* **City (VARCHAR(50), NOT NULL):** City name, crucial for geographic location information.
* **District (VARCHAR(50), NOT NULL):** District/neighborhood/state/county name, providing more specific location details.
* **Country (VARCHAR(50), NOT NULL):** Country name, important for national location context.