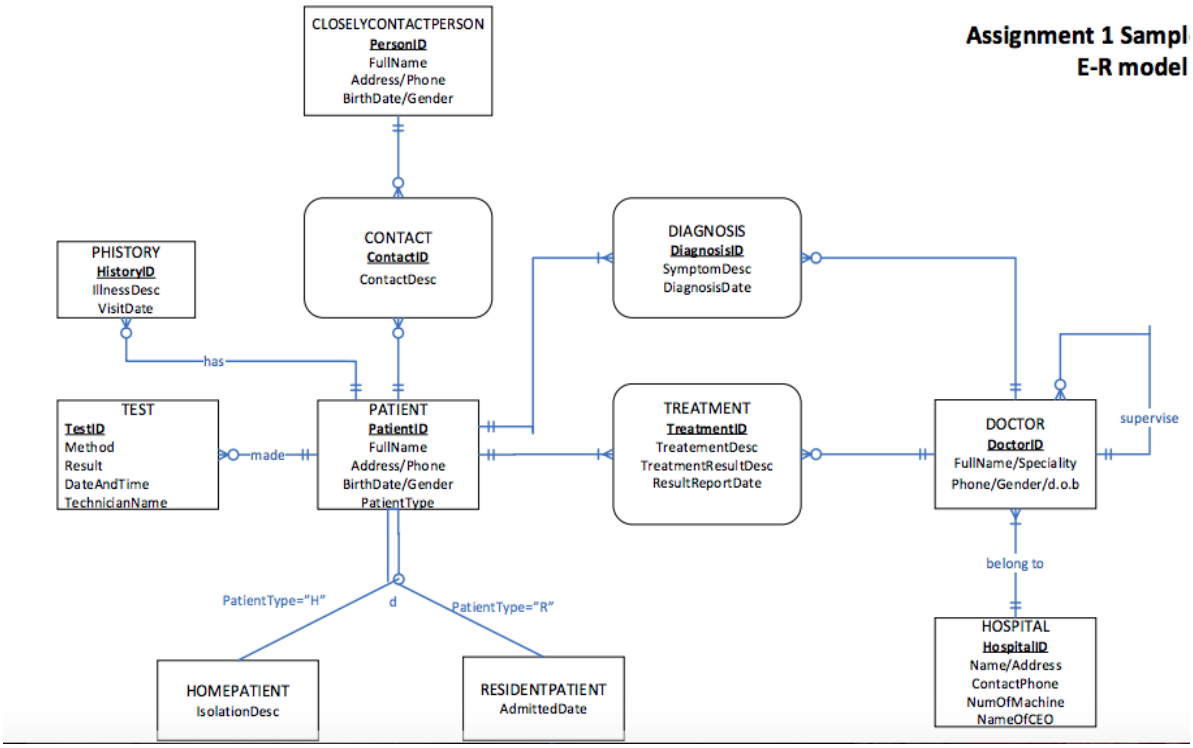
Assignment 1: Analyze a given business scenario and create an ER diagram that includes entities, relationships, attributes, and cardinality. Ensure that the diagram reflects proper normalization up to the third normal form.



Assignment 2: Design a database schema for a library system, including tables, fields, and constraints like NOT NULL, UNIQUE, and CHECK. Include primary and foreign keys to establish relationships between tables.

```
CREATE TABLE Authors (
    author_id INT PRIMARY KEY,
    author_name VARCHAR(100) NOT NULL
);

CREATE TABLE Publishers (
    publisher_id INT PRIMARY KEY,
    publisher_name VARCHAR(100) NOT NULL
);

CREATE TABLE Books (
    book_id INT PRIMARY KEY,
    title VARCHAR(100) NOT NULL,
    author_id INT NOT NULL,
    publisher_id INT NOT NULL,
    publication_year INT,
    genre VARCHAR(50),
    UNIQUE (title, author_id),
    FOREIGN KEY (author_id) REFERENCES Authors(author_id),
    FOREIGN KEY (publisher_id) REFERENCES Publishers(publisher_id)
);

CREATE TABLE Members (
    member_id INT PRIMARY KEY,
    member_name VARCHAR(100) NOT NULL,
    email VARCHAR(100),
```

```
    phone VARCHAR(20),
    address VARCHAR(255)
);

CREATE TABLE Borrowings (
    borrowing_id INT PRIMARY KEY,
    book_id INT NOT NULL,
    member_id INT NOT NULL,
    borrow_date DATE NOT NULL,
    return_date DATE,
    CHECK (return_date >= borrow_date),
    FOREIGN KEY (book_id) REFERENCES Books(book_id),
    FOREIGN KEY (member_id) REFERENCES Members(member_id)
);
```

Assignment 3: Explain the ACID properties of a transaction in your own words. Write SQL statements to simulate a transaction that includes locking and demonstrate different isolation levels to show concurrency control.

Atomicity: Imagine you're buying a book online. Either all the steps go smoothly (you pay, the order is processed, and the book is shipped), or none of them happen (your payment fails, and the order isn't placed), ensuring you don't end up paying without getting your book.

Consistency: Picture a bank transfer. Your account balance should be consistent before and after the transfer, even if something goes wrong during the transaction, preventing your money from disappearing into thin air or magically appearing out of nowhere.

Isolation: Think of two people withdrawing money from the same ATM simultaneously. Isolation ensures that one person's transaction doesn't interfere with the other's, preventing one person from seeing the other's account balance or withdrawing money that has already been withdrawn.

Durability: Consider updating your social media status. Once you hit "post," your status update should remain even if your phone dies or the app crashes, ensuring your updates are saved and permanent.

```
CREATE TABLE Accounts (
    account_id INT PRIMARY KEY,
    balance DECIMAL(10, 2) NOT NULL
);

BEGIN TRANSACTION;

-- Deduct funds from account A
UPDATE Accounts SET balance = balance - 100 WHERE account_id = 1;

-- Add funds to account B
UPDATE Accounts SET balance = balance + 100 WHERE account_id = 2;

COMMIT;
```

Read Uncommitted

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
```

Read Committed:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

Repeatable Read:

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

Serializable

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

Assignment 4: Write SQL statements to CREATE a new database and tables that reflect the library schema you designed earlier. Use ALTER statements to modify the table structures and DROP statements to remove a redundant table.

Create a New Database:

```
CREATE DATABASE LibrarySystem;
```

Create Tables:

```
USE LibrarySystem;


Create Authors Table
CREATE TABLE Authors (
    author_id INT PRIMARY KEY AUTO_INCREMENT,
    author_name VARCHAR(100) NOT NULL
);


 Create Publishers Table
CREATE TABLE Publishers (
    publisher_id INT PRIMARY KEY AUTO_INCREMENT,
    publisher_name VARCHAR(100) NOT NULL
);


-Create Books Table
CREATE TABLE Books (
```

```
    book_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(100) NOT NULL,
    author_id INT NOT NULL,
    publisher_id INT NOT NULL,
    publication_year INT,
    genre VARCHAR(50),
    UNIQUE (title, author_id),
    FOREIGN KEY (author_id) REFERENCES Authors(author_id),
    FOREIGN KEY (publisher_id) REFERENCES Publishers(publisher_id)
);
```

Assignment 2: Demonstrate the creation of an index on a table and discuss how it improves query performance. Use a DROP INDEX statement to remove the index and analyze the impact on query execution.

```
-Create Members Table
CREATE TABLE Members (
    member_id INT PRIMARY KEY AUTO_INCREMENT,
    member_name VARCHAR(100) NOT NULL,
    email VARCHAR(100),
    phone VARCHAR(20),
    address VARCHAR(255)
);

 Create Borrowings Table
CREATE TABLE Borrowings (
    borrowing_id INT PRIMARY KEY AUTO_INCREMENT,
```

```
    book_id INT NOT NULL,
    member_id INT NOT NULL,
    borrow_date DATE NOT NULL,
    return_date DATE,
    CHECK (return_date >= borrow_date),
    FOREIGN KEY (book_id) REFERENCES Books(book_id),
    FOREIGN KEY (member_id) REFERENCES Members(member_id)
);
```

Alter Table Structures:

```
ALTER TABLE Books
ADD COLUMN language VARCHAR(50) AFTER genre;
```

Drop Redundant Table:

```
DROP TABLE Publishers;
```

Create an Index:

```
CREATE INDEX idx_title ON Books(title);
```

Query Performance Improvement:

By creating an index on the `title` column, the database engine creates a separate data structure that stores the values of the `title` column in sorted order. This allows the database to quickly locate rows based on the `title` column when executing queries that involve filtering, sorting, or joining based on the `title` column.

SELECT * FROM Books WHERE title = ' hell and heaven';

Remove the Index:

DROP INDEX idx_title ON Books;

Analyze Impact on Query Execution:

After removing the index, queries that rely on filtering, sorting, or joining based on the `title` column may experience slower performance. Without the index, the database engine has to perform a full table scan to locate rows based on the `title` column, which can be slower, especially for large tables.

SELECT * FROM Books WHERE title = 'hell and heaven';

Assignment 6: Create a new database user with specific privileges using the CREATE USER and GRANT commands. Then, write a script to REVOKE certain privileges and DROP the user.

CREATE USER 'thanuja'@'DESKTOP-O7HDQ23' IDENTIFIED BY 'Saanviee@99';

GRANT ALL PRIVILEGES ON mydatabase.* TO 'thanuja'@'DESKTOP-O7HDQ23';

FLUSH PRIVILEGES;

REVOKE DELETE ON mydatabase.* FROM 'thanuja'@''DESKTOP-O7HDQ23'';

FLUSH PRIVILEGES;

DROP USER 'thanuja'DESKTOP-O7HDQ23';

Assignment 7: Prepare a series of SQL statements to INSERT new records into the library tables, UPDATE existing records with new information, and DELETE records based on specific criteria. Include BULK INSERT operations to load data from an external source.

```
INSERT INTO Books (Title, Author, Publisher, PublicationDate)
VALUES ('Tail', 'leo', 'Publisher Inc.', '2020-01-01');
INSERT INTO Authors (Name, Bio)
VALUES ('leo', 'leo is famous');
INSERT INTO Borrowers (Name, Email, Phone)
VALUES ('leo', 'leo@example.com', 9876565656');
```

```
UPDATE Books
SET PublicationDate = '2022-02-01'
WHERE Title = 'hell and  heaven';
```

```
UPDATE Authors
SET Bio = ''
WHERE Name = 'pitter';
```