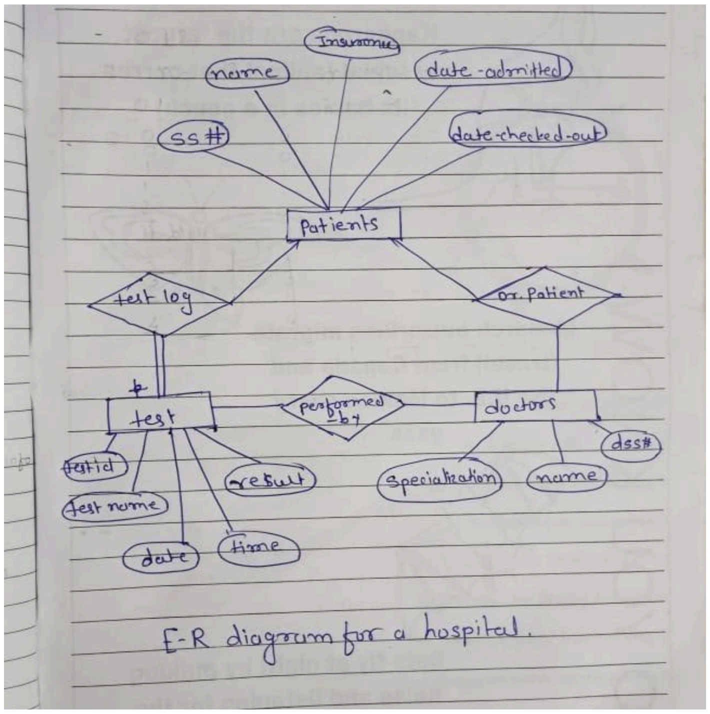
SQL Part 1 Assignment

Assignment 1

Name: Akshada Baad

Batch - CPPE

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

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.

Tables, Fields, and Constraints:

1. Books:
- BookID (PK), Title (NOT NULL), AuthorID (FK, NOT NULL), ISBN (UNIQUE, NOT NULL), PublishedYear, CategoryID (FK, NOT NULL)

2. Authors:
- AuthorID (PK), Name (NOT NULL)

3. Categories:
- CategoryID (PK), CategoryName (UNIQUE, NOT NULL)

4. Members:
- MemberID (PK), Name (NOT NULL), Email (UNIQUE, NOT NULL), Phone

5. Loans:

- LoanID (PK), BookID (FK, NOT NULL), MemberID (FK, NOT NULL), LoanDate (NOT NULL), ReturnDate

Primary and Foreign Keys:

- Books.AuthorID -> Authors.AuthorID
- Books.CategoryID -> Categories.CategoryID
- Loans.BookID -> Books.BookID
- Loans.MemberID -> Members.MemberID

SQL Schema:

```
CREATE TABLE Authors (
   AuthorID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL
);

CREATE TABLE Categories (
   CategoryID INT PRIMARY KEY,
   CategoryName VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE Books (
   BookID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
   AuthorID INT NOT NULL,
   ISBN VARCHAR(13) UNIQUE NOT NULL,
   PublishedYear INT,
   CategoryID INT NOT NULL,
   FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),
   FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
);

CREATE TABLE Members (
   MemberID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   Email VARCHAR(100) UNIQUE NOT NULL,
   Phone VARCHAR(15)
);

CREATE TABLE Loans (
   LoanID INT PRIMARY KEY,
   BookID INT NOT NULL,
   MemberID INT NOT NULL,
   LoanDate DATE NOT NULL,
   ReturnDate DATE,
   FOREIGN KEY (BookID) REFERENCES Books(BookID),
   FOREIGN KEY (MemberID) REFERENCES Members(MemberID)
);
```

Assignment 3

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.

- Loans.BookID -> Books.BookID
- Loans.MemberID -> Members.MemberID

ACID Properties:

Atomicity: Ensures that all operations within a transaction are completed; if not, the transaction is aborted and no changes are made.

Consistency: Ensures that the database remains in a consistent state before and after the transaction.

Isolation: Ensures that transactions are securely and independently processed at the same time without interference.

Durability: Ensures that once a transaction has been committed, it will remain so, even in the event of a power loss, crash, or error.

SQL Transaction with Locking and Isolation Levels:

```
-- Transaction to update book and member details
START TRANSACTION;

-- Lock the book and member rows to ensure consistency
SELECT * FROM Books WHERE BookID = 1 FOR UPDATE;
SELECT * FROM Members WHERE MemberID = 1 FOR UPDATE;

-- Update book and member details
UPDATE Books SET Title = 'New Title' WHERE BookID = 1;
UPDATE Members SET Name = 'New Name' WHERE MemberID = 1;

-- Commit the transaction
```

```
COMMIT;

-- Set Isolation Level
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

-- Transaction with high isolation
START TRANSACTION;

-- Operations
SELECT * FROM Books WHERE BookID = 2 FOR UPDATE;
UPDATE Books SET PublishedYear = 2020 WHERE BookID = 2;

COMMIT;
```

Assignment 4

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 database and tables:

```
CREATE DATABASE Library;

USE Library;

CREATE TABLE Authors (
   AuthorID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL
);

CREATE TABLE Categories (
   CategoryID INT PRIMARY KEY,
   CategoryName VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE Books (
   BookID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
   AuthorID INT NOT NULL,
   ISBN VARCHAR(13) UNIQUE NOT NULL,
   PublishedYear INT,
   CategoryID INT NOT NULL,
   FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),
   FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
);

CREATE TABLE Members (
   MemberID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   Email VARCHAR(100) UNIQUE NOT NULL,
   Phone VARCHAR(15)
);

CREATE TABLE Loans (
   LoanID INT PRIMARY KEY,
   BookID INT NOT NULL,
   MemberID INT NOT NULL,
   LoanDate DATE NOT NULL,
   ReturnDate DATE,
   FOREIGN KEY (BookID) REFERENCES Books(BookID),
   FOREIGN KEY (MemberID) REFERENCES Members(MemberID)
);
```

Alter and drop statements:

```
-- Add a new column to Members
ALTER TABLE Members ADD COLUMN Address VARCHAR(255);

-- Modify column in Books
ALTER TABLE Books MODIFY COLUMN PublishedYear YEAR;

-- Drop an unnecessary table
DROP TABLE IF EXISTS TemporaryTable;
```

Assignment 5

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.

Index creation:

```
-- Create an index on the Title column of Books table
CREATE INDEX idx_title ON Books(Title);

-- Query to demonstrate improved performance
```

```
SELECT * FROM Books WHERE Title LIKE 'The%';
```

Drop index and impact:

```
-- Drop the index
DROP INDEX idx_title ON Books;

-- Query performance without index
SELECT * FROM Books WHERE Title LIKE 'The%';
```

Impact Analysis: With the index, the query searching for books by title is significantly faster as the index allows the database to quickly locate the rows. Without the index, the database performs a full table scan, which is much slower especially as the table grows in size.

Assignment 6

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 and grant privilege:

```
CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'password123';

GRANT SELECT, INSERT, UPDATE, DELETE ON Library.* TO 'library_user'@'localhost';
```

Revoke privilege and drop user:

```
REVOKE INSERT, UPDATE ON Library.* FROM 'library_user'@'localhost';

DROP USER 'library_user'@'localhost';
```

Assignment 7

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 records:

```
INSERT INTO Authors (AuthorID, Name) VALUES (1, 'George Orwell');
INSERT INTO Categories (CategoryID, CategoryName) VALUES (1, 'Dystopian');
INSERT INTO Books (BookID, Title, AuthorID, ISBN, PublishedYear, CategoryID) VALUES (1,
'1984', 1, '1234567890123', 1949, 1);
INSERT INTO Members (MemberID, Name, Email, Phone) VALUES (1, 'John Doe',
'johndoe@example.com', '1234567890');
INSERT INTO Loans (LoanID, BookID, MemberID, LoanDate, ReturnDate) VALUES (1, 1, 1,
'2023-05-01', NULL);
```

Update records :

```
UPDATE Books SET PublishedYear = 1950 WHERE BookID = 1;
UPDATE Members SET Phone = '0987654321' WHERE MemberID = 1;
```

Delete record:

```
DELETE FROM Loans WHERE LoanID = 1;
DELETE FROM Members WHERE MemberID = 1;
```

Bulk Insert:

```
LOAD DATA IN
```