ASSIGNMENT_RDBMS_DAY:07

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 normalizationup to the third normal form.

Entities:

Student

Attributes: StudentID (Primary Key), Name, Email, Phone

Instructor

Attributes: InstructorID (Primary Key), Name, Email, Phone

Course

Attributes: CourseID (Primary Key), Title, Credits

Section

Attributes: SectionID (Primary Key), CourseID (Foreign Key), InstructorID (Foreign Key), Schedule, Capacity

Relationships:

Registration

A student can register for multiple sections of courses. A section can have multiple students registered.

Attributes: StudentID (Foreign Key), SectionID (Foreign Key)

Teaching

An instructor can teach multiple sections of courses. A section is taught by one instructor.

Attributes: InstructorID (Foreign Key), SectionID (Foreign Key)

Cardinality:

Student - Registration - Section

One student can register for multiple sections. One section can have multiple students.

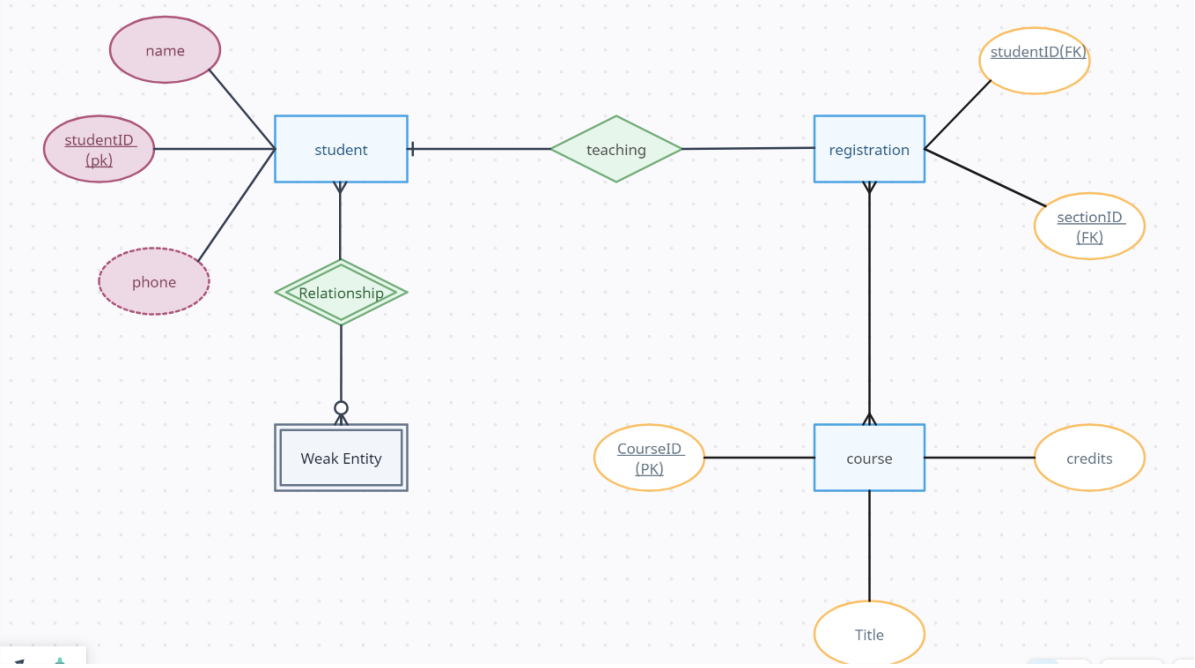
Many-to-Many relationship.

Instructor - Teaching - Section

One instructor can teach multiple sections. One section is taught by one instructor.

One-to-Many relationship.

ER Diagram:



Normalization:

1st Normal Form (1NF): All attributes are atomic, and there are no repeatinggroups.

2nd Normal Form (2NF): All non-key attributes are fully functional dependent on the primary key. In this ER diagram, all attributes depend on the primary keys of their respective entities.

3rd Normal Form (3NF): There are no transitive dependencies, meaning no non-key attributes depend on other non-key attributes. In this ER diagram,there are no transitive dependencies.

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 ( AuthorID
    INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Nationality VARCHAR(50)
);

CREATE TABLE Books ( BookID
    INT PRIMARY KEY,
    Title VARCHAR(255) NOT NULL,
    AuthorID INT,
    ISBN VARCHAR(20) UNIQUE,
    Genre VARCHAR(50),
    PublishedYear INT,
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID)
);

CREATE TABLE Borrowers ( BorrowerID
    INT PRIMARY KEY, Name
    VARCHAR(100) NOT NULL, Email
    VARCHAR(100) NOT NULL, Phone
    VARCHAR(15),
    UNIQUE (Email)
```

```
);

CREATE TABLE Transactions (
    TransactionID INT PRIMARY KEY,
    BookID INT,
    BorrowerID INT,
    TransactionDate DATE NOT NULL,
    DueDate DATE,
    ReturnDate DATE,
    FOREIGN KEY (BookID) REFERENCES Books(BookID),
    FOREIGN KEY (BorrowerID) REFERENCES Borrowers(BorrowerID), CHECK
    (ReturnDate >= TransactionDate)
);
```

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 showconcurrency control.

ACID Properties of a Transaction:

Atomicity:

Atomicity ensures that a transaction is treated as a single unit of work. It means that either all the operations within the transaction are successfullycompleted, or none of them are.

Consistency:

Consistency ensures that the database remains in a consistent state before and after the transaction. It means that the data must meet all integrity constraints, including referential integrity and domain constraints.

Isolation:

Isolation ensures that concurrent execution of transactions results in a system state that would be obtained if transactions were executed serially. It means that transactions are executed independently of each other, and the intermediate state of one transaction is not visible to other transactions until it is committed.

Durability:

Durability ensures that once a transaction is committed, its changes are permanent and survive system failures. It means that the changes made by the transaction are persistent and will not be lost even in the event of a system crash or restart.

SQL Statements for Transaction with Locking and Isolation Levels:

```
BEGIN TRANSACTION;

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

SELECT * FROM Accounts WHERE AccountID IN (123, 456) FOR UPDATE;

UPDATE Accounts SET Balance = Balance - 100 WHERE AccountID = 123; UPDATE

Accounts SET Balance = Balance + 100 WHERE AccountID = 456;COMMIT;
```

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.

Creating a New Database and Tables:

```
CREATE DATABASE LibraryDB;

USE LibraryDB;

CREATE TABLE Authors ( AuthorID
    INT PRIMARY KEY,

    Name VARCHAR(100) NOT NULL,
```

```
    Nationality VARCHAR(50));
CREATE TABLE Books (
    BookID INT PRIMARY KEY,
    Title VARCHAR(255) NOT NULL,
    AuthorID INT,
    ISBN VARCHAR(20) UNIQUE,
    Genre VARCHAR(50),
    PublishedYear INT,
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID));

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

CREATE TABLE Transactions (
    TransactionID INT PRIMARY KEY,
    BookID INT,
    BorrowerID INT,
    TransactionDate DATE NOT NULL,
    DueDate DATE,
    ReturnDate DATE,
    FOREIGN KEY (BookID) REFERENCES Books(BookID),
    FOREIGN KEY (BorrowerID) REFERENCES Borrowers(BorrowerID),
```

```
    CHECK (ReturnDate >= TransactionDate)
);
```

Modifying Table Structures with ALTER Statements:

ALTER TABLE Books ADD Description TEXT;

Dropping a Redundant Table:

DROP TABLE Borrowers;

Assignment 5: 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.

Creating an Index:

Suppose we want to create an index on the Title column of the Books table: CREATE INDEX idx_title ON Books (Title);

How it Improves Query Performance:

Creating an index on the Title column will speed up queries that involve searching, sorting, or filtering based on the book title. When a query is executed that involves the Title column, the database engine can use the index to quickly locate the relevant rows without having to scan the entire table sequentially. This improves query performance by reducing the time required to retrieve data, especially for large datasets.

Dropping the Index:

-- Drop the index on the Title column of the Books table

DROP INDEX idx_title ON Books;

Impact on Query Execution:

Without the index, queries that involve searching, sorting, or filtering based on the book title may experience slower performance compared to when the index was present.

The database engine will need to perform a full table scan to locate the relevant rows, which can be time-consuming, especially for large tables.

Queries that heavily rely on the Title column may suffer from degraded performance after dropping the index, as the database engine lacks the optimized access path provided by the index.

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

Create a new database user

CREATE USER 'new_user'@'localhost' IDENTIFIED BY 'password';

Grant specific privileges to the user on a specific database

GRANT SELECT, INSERT, UPDATE ON librarydb.* TO 'new_user'@'localhost';

Revoke certain privileges from the user

REVOKE INSERT ON librarydb.* FROM 'new_user'@'localhost';

Drop the user

DROP USER 'new_user'@'localhost';

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. IncludeBULK INSERT operations to load data from an external source.

INSERT New Records:

INSERT INTO Authors (AuthorID, Name, Nationality) VALUES (1, 'J.K. Rowling','British');

INSERT INTO Books (BookID, Title, AuthorID, ISBN, Genre, PublishedYear) VALUES (1, 'Harry Potter and the Philosopher''s

INSERT INTO Borrowers (BorrowerID, Name, Email, Phone) VALUES (1, 'John Doe', 'john.doe@example.com', '123-456-7890');

INSERT INTO Transactions (TransactionID, BookID, BorrowerID, TransactionDate, DueDate, ReturnDate) VALUES (1, 1, 1, '2024-05-21', '2024-
06-21', NULL);

UPDATE Existing Records:

UPDATE Books SET Title = 'Harry Potter and the Sorcerer''s Stone' WHEREBookID = 1;

UPDATE Borrowers SET Email = 'johndoe@example.com' WHERE BorrowerID =1;

DELETE Records Based on Specific Criteria:

DELETE FROM Books WHERE BookID = 1;

DELETE FROM Transactions WHERE TransactionDate < '2024-01-01';

BULK INSERT Operations:

LOAD DATA INFILE INTO TABLE Books

FIELDS TERMINATED BY LINES TERMINATED BY

IGNORE 1 ROWS;