**Conceptual ERD Model:**

- Correctly identified entities and relationships.

- Consider adding an entity for "Category" to further normalize the data.

**Logical ERD Model:**

- Well-defined attributes for each entity.

- Consider adding a "Status" attribute to the Enrollment entity (e.g., "Enrolled", "Completed", "InProgress").

**Physical ERD Model (Database Schema):**

- Proper use of primary keys, foreign keys, and data types.

- Consider indexing columns used in WHERE, JOIN, and ORDER BY clauses for improved performance.

Additional Suggestions:

1. Consider adding a "Rating" attribute to the Course entity to store user ratings.

2. Add a "Timestamp" attribute to the Completion entity to track completion time.

3. Create a separate table for "Questions" to store individual quiz questions.

4. Use ENUM data type for "Category" and "Status" attributes.

5. Use VARCHAR(255) for "Password" to accommodate hashed passwords.

SQL Enhancements:

1. Use CONSTRAINTS to enforce data integrity.

2. Create INDEXES on columns used in queries.

3. Consider using VIEWs to simplify complex queries.

Next Steps:

1. Implement the database schema.

2. Create CRUD (Create, Read, Update, Delete) operations for each entity.

3. Develop queries for reporting and analytics.

Would you like me to:

1. Review and refine the database schema?

2. Assist with CRUD operation implementation?

3. Develop queries for reporting and analytics?

4. Discuss database security and backup strategies?

**Conceptual ER Diagram:**

**Restaurants:**

Restaurant\_ID

Name

Location

**Tables:**

Table\_ID

Restaurant\_ID

Capacity

Status

**Guests:**

Guest\_ID

Name

Email

Phone

**Reservations:**

Reservation\_ID

Table\_ID

Guest\_ID

Date

Time

Status

**Payments :**

Payment\_ID

Reservation\_ID

Amount

Method

Status

**Logical ER Diagram:**

CREATE TABLE Restaurants (

Restaurant\_ID INT PRIMARY KEY,

Name VARCHAR(100),

Location VARCHAR(200)

);

CREATE TABLE Tables (

Table\_ID INT PRIMARY KEY,

Restaurant\_ID INT,

Capacity INT,

Status VARCHAR(20) CHECK(Status IN ('Available', 'Reserved', 'Occupied'))

FOREIGN KEY (Restaurant\_ID) REFERENCES Restaurants(Restaurant\_ID)

);

CREATE TABLE Guests (

Guest\_ID INT PRIMARY KEY,

Name VARCHAR(100),

Email VARCHAR(100),

Phone VARCHAR(20)

);

CREATE TABLE Reservations (

Reservation\_ID INT PRIMARY KEY,

Table\_ID INT,

Guest\_ID INT,

Date DATE,

Time TIME,

Status VARCHAR(20) CHECK(Status IN ('Pending', 'Confirmed', 'Canceled')),

FOREIGN KEY (Table\_ID) REFERENCES Tables(Table\_ID),

FOREIGN KEY (Guest\_ID) REFERENCES Guests(Guest\_ID)

);

CREATE TABLE Payments (

Payment\_ID INT PRIMARY KEY,

Reservation\_ID INT,

Amount DECIMAL(10, 2),

Method VARCHAR(20),

Status VARCHAR(20) CHECK(Status IN ('Pending', 'Paid', 'Refunded')),

FOREIGN KEY (Reservation\_ID) REFERENCES Reservations(Reservation\_ID)

);

**Physical ER Diagram:**

CREATE TABLE Restaurants (

Restaurant\_ID INT PRIMARY KEY,

Name VARCHAR(100),

Location VARCHAR(200)

);

CREATE TABLE Tables (

Table\_ID INT PRIMARY KEY,

Restaurant\_ID INT,

Capacity INT,

Status VARCHAR(20) CHECK(Status IN ('Available', 'Reserved', 'Occupied')),

FOREIGN KEY (Restaurant\_ID) REFERENCES Restaurants(Restaurant\_ID)

);

CREATE TABLE Guests (

Guest\_ID INT PRIMARY KEY,

Name VARCHAR(100),

Email VARCHAR(100),

Phone VARCHAR(20)

);

CREATE TABLE Reservations (

Reservation\_ID INT PRIMARY KEY,

Table\_ID INT,

Guest\_ID INT,

Date DATE,

Time TIME,

Status VARCHAR(20) CHECK(Status IN ('Pending', 'Confirmed', 'Canceled')),

FOREIGN KEY (Table\_ID) REFERENCES Tables(Table\_ID),

FOREIGN KEY (Guest\_ID) REFERENCES Guests(Guest\_ID)

);

CREATE TABLE Payments (

Payment\_ID INT PRIMARY KEY,

Reservation\_ID INT,

Amount DECIMAL(10, 2),

Method VARCHAR(20),

Status VARCHAR(20) CHECK(Status IN ('Pending', 'Paid', 'Refunded')),

FOREIGN KEY (Reservation\_ID) REFERENCES Reservations(Reservation\_ID)

);

Stored Procedures

CREATE PROCEDURE sp\_make\_reservation

@Table\_ID INT,

@Guest\_ID INT,

@Date DATE,

@Time TIME

AS

BEGIN

INSERT INTO Reservations (Table\_ID, Guest\_ID, Date, Time, Status)

VALUES (@Table\_ID, @Guest\_ID, @Date, @Time, 'Pending');

END;

CREATE PROCEDURE sp\_update\_reservation

@Reservation\_ID INT,

@Table\_ID INT,

@Guest\_ID INT,

@Date DATE,

@Time TIME

AS

BEGIN

UPDATE Reservations

SET Table\_ID = @Table\_ID, Guest\_ID = @Guest\_ID, Date = @Date, Time = @Time

WHERE Reservation\_ID = @Reservation\_ID;

END;

CREATE PROCEDURE sp\_cancel\_reservation

@Reservation\_ID INT

AS

BEGIN

UPDATE Reservations

SET Status = 'Canceled'

WHERE Reservation\_ID = @Reservation\_ID;

END;

Triggers

CREATE TRIGGER tr\_update\_table\_availability

ON Reservations

AFTER INSERT, UPDATE

AS

BEGIN

UPDATE Tables

SET Status = 'Reserved'

WHERE Table\_ID = INSERTED.Table\_ID;

END;

CREATE TRIGGER tr\_update\_payment\_status

ON Payments

AFTER INSERT, UPDATE

AS