PG-DAC Aug 25 Batch
Assignment: ER-Diagram Design

ANSWERS

Question 1: University Management System

Problem Statement: A university wants to maintain data about Students, Courses, Faculty, and Departments. The university management has provided the following requirements: 1. Each student has a unique roll number, name, email, phone number, and date of admission. 2. A student can enroll in many courses, and each course can have many students. 3. Each course has a course ID, course name, credits, and is offered by exactly one department. 4. Each department has a department ID, department name, and location. 5. Each faculty member has a faculty ID, name, designation, and email. 6. A faculty can teach many courses, but a course can be taught by only one faculty. 7. A student is assigned to exactly one faculty advisor.

Tasks for Students:

1. Identify Entities & Attributes – List out all entities and their attributes (mention primary keys). 2. Define Relationships – Define the relationships between entities along with cardinalities (1:1, 1:M, M:N). 3. Draw the ER Diagram – Use symbols for entities, attributes, relationships, and cardinalities. 4. Convert to Tables (Optional Advanced Task) – Convert your ER diagram into relational schema (tables with PK and FK).

1. Identify Entities and Attributes

A. Student
Primary Key: Roll_No
Attributes: Name, Email, Phone_No, Date_of_Admission
Relationships: Enrolls in Courses, Assigned to Faculty Advisor
Associative entity: Enrollment, Attributes: Roll_No (FK), Course_ID (FK)

B. Course
Primary Key: Course_ID
Attributes: Course_Name, Credits
Relationships: Belongs to one Department, Taught by one Faculty, Enrolled by many Students

C. Department
Primary Key: Dept_ID
Attributes: Dept_Name, Location
Relationships: Offers many Courses

D. Faculty
Primary Key: Faculty_ID
Attributes: Name, Designation, Email
Relationships: Teaches many Courses, Advisor to many Students

2. Define Relationships

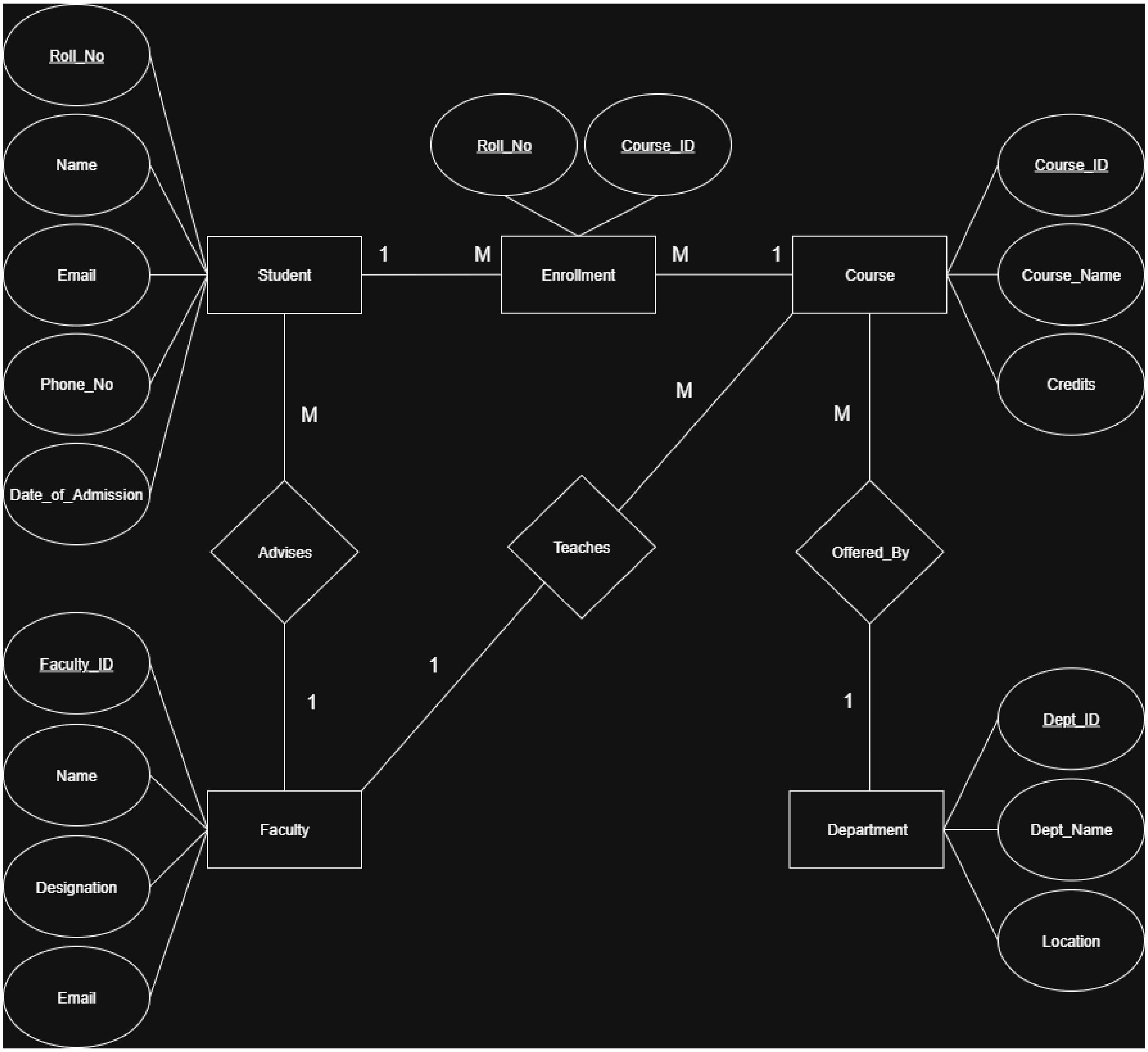
A. Student – Course
Relationship: Enrolls
Cardinality: M:N

B. Course – Department
Relationship: Offered_By
Cardinality: M:1

C. Faculty – Course
Relationship: Teaches
Cardinality: 1:M

D. Student – Faculty (Advisor)
Relationship: Advised_By
Cardinality: M:1

3. Entity Relationship Diagram



4. Relational Schema

A. Department Table

```
CREATE TABLE Department (
    Dept_ID INT PRIMARY KEY,
    Dept_Name VARCHAR(100) NOT NULL,
    Location VARCHAR(100)
);
```

B. Faculty Table

```
CREATE TABLE Faculty (
    Faculty_ID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Designation VARCHAR(50),
    Email VARCHAR(100) UNIQUE
);
```

C. Student Table

```
CREATE TABLE Student (
    Roll_No INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Email VARCHAR(100) UNIQUE,
    Phone_No VARCHAR(15),
    Date_of_Admission DATE,
    Faculty_ID INT,
    FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID)
);
```

D. Course Table

```
CREATE TABLE Course (
    Course_ID INT PRIMARY KEY,
    Course_Name VARCHAR(100) NOT NULL,
    Credits INT,
    Dept_ID INT,
    Faculty_ID INT,
    FOREIGN KEY (Dept_ID) REFERENCES Department(Dept_ID),
    FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID)
);
```

E. Enrollment Table (Associative table for Student-Course M:N relationship)

```
CREATE TABLE Enrollment (
    Roll_No INT,
    Course_ID INT,
    PRIMARY KEY (Roll_No, Course_ID),
    FOREIGN KEY (Roll_No) REFERENCES Student(Roll_No),
    FOREIGN KEY (Course_ID) REFERENCES Course(Course_ID)
);
```

Question 2: Online Shopping System

Problem Statement: An online shopping portal wants to maintain information about Customers, Orders, Products, Payments, and Delivery. The requirements are as follows: 1. Each customer has a customer ID, name, email, phone, and address. 2. A customer can place many orders, but each order belongs to only one customer. 3. An order has an order ID, order date, total amount, and status. 4. Each order can contain multiple products, and a product can appear in many orders (many-to-many relationship). 5. Each product has a product ID, name, description, price, and stock quantity. 6. Each order must have one payment, which includes payment ID, payment date, amount, and mode of payment (UPI, Credit Card, COD). 7. Each order is linked to a delivery, which includes delivery ID, delivery date, delivery status, and delivery address.

Tasks for Students:

1. Identify Entities & Attributes – List entities and attributes (with primary keys). 2. Define Relationships – Show all relationships with correct cardinalities. 3. Draw the ER Diagram – Use proper ER notation for entities, attributes, and relationships. 4. Optional Task – Convert ER diagram into relational schema (tables with PK and FK).

1. Identify Entities and Attributes

A. Customer
Primary Key: Customer_ID
Attributes: Name, Email, Phone, Address
Relationships: Places many Orders

B. Order
Primary Key: Order_ID
Attributes: Order_Date, Total_Amount, Status
Relationships: Belongs to one Customer, Contains many Products, Has one Payment, Linked to one Delivery
Associative entity: Order_Product, Attributes: Order_ID (FK), Product_ID (FK), Quantity

C. Product
Primary Key: Product_ID
Attributes: Name, Description, Price, Stock_Quantity
Relationships: Appears in many Orders

D. Payment
Primary Key: Payment_ID
Attributes: Payment_Date, Amount, Payment_Mode
Relationships: Belongs to exactly one Order

E. Delivery
Primary Key: Delivery_ID
Attributes: Delivery_Date, Delivery_Status, Delivery_Address
Relationships: Linked to exactly one Order

2. Define Relationships

A. Customer – Order
Relationship: Places
Cardinality: 1:M (One customer can place many orders, each order belongs to one customer)

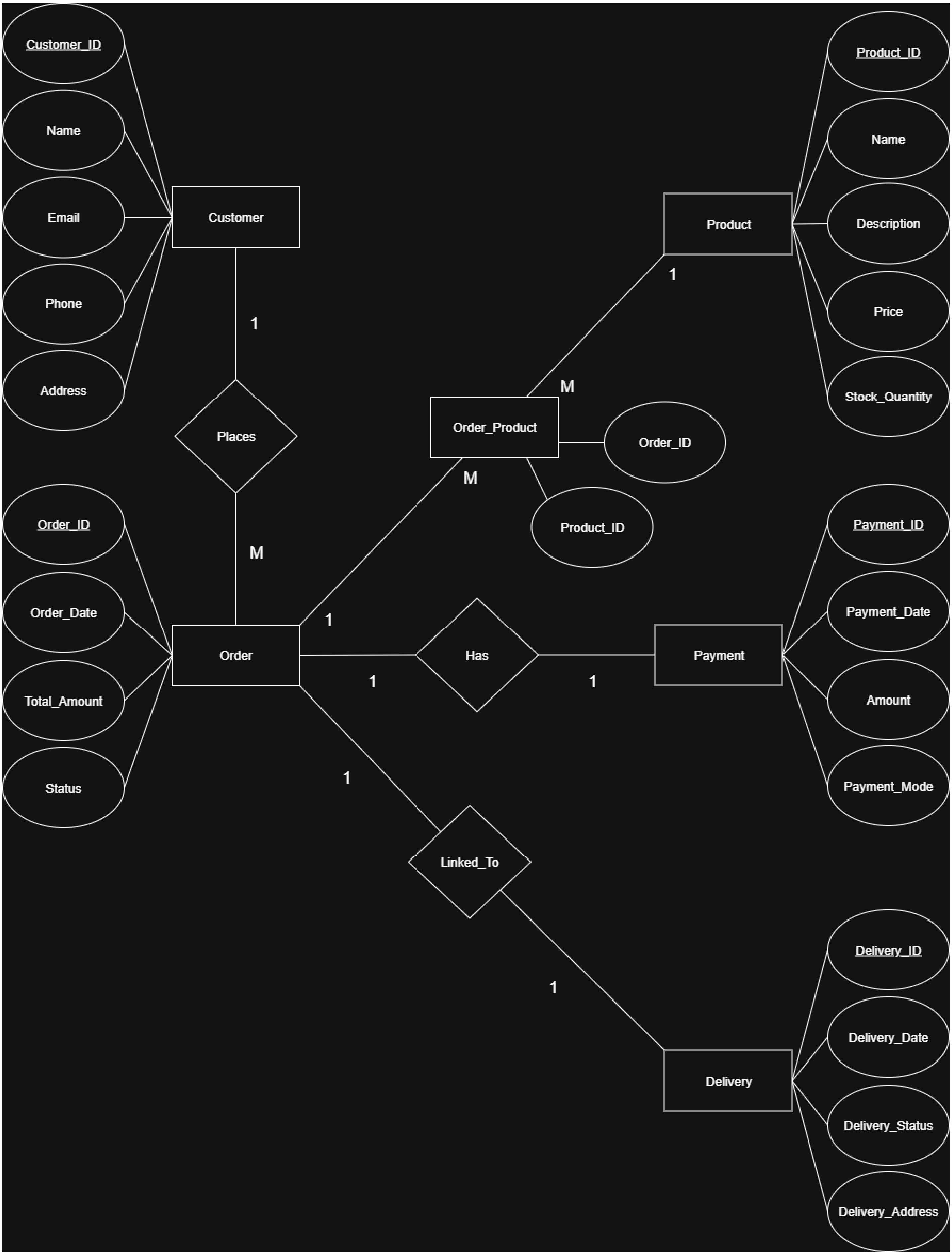
B. Order – Product

Relationship: Contains
Cardinality: M:N (One order can have many products, and one product can appear in many orders)

C. Order – Payment
Relationship: Has
Cardinality: 1:1 (Each order has exactly one payment)

D. Order – Delivery
Relationship: Linked_To
Cardinality: 1:1 (Each order has exactly one delivery)

3. Entity Relationship Diagram



4. Relational Schema

A. Customer Table

```
CREATE TABLE Customer (
    Customer_ID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Email VARCHAR(100) UNIQUE,
    Phone VARCHAR(15),
    Address VARCHAR(200)
);
```

B. Product Table

```
CREATE TABLE Product (
    Product_ID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Description TEXT,
    Price DECIMAL(10,2),
    Stock_Quantity INT
);
```

C. Orders Table

```
CREATE TABLE Orders (
    Order_ID INT PRIMARY KEY,
    Order_Date DATE,
    Total_Amount DECIMAL(10,2),
    Status VARCHAR(50),
    Customer_ID INT,
    FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID)
);
```

D. Payment Table

```
CREATE TABLE Payment (
    Payment_ID INT PRIMARY KEY,
    Payment_Date DATE,
    Amount DECIMAL(10,2),
    Payment_Mode VARCHAR(50),
    Order_ID INT UNIQUE,
    FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID)
);
```

E. Delivery Table

```
CREATE TABLE Delivery (
    Delivery_ID INT PRIMARY KEY,
    Delivery_Date DATE,
    Delivery_Status VARCHAR(50),
    Delivery_Address VARCHAR(200),
    Order_ID INT UNIQUE,
    FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID)
);
```

F. Order_Product Table (Associative table for Order–Product M:N relationship)

```
CREATE TABLE Order_Product (
    Order_ID INT,
    Product_ID INT,
    PRIMARY KEY (Order_ID, Product_ID),
    FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID),
    FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID)
);
```