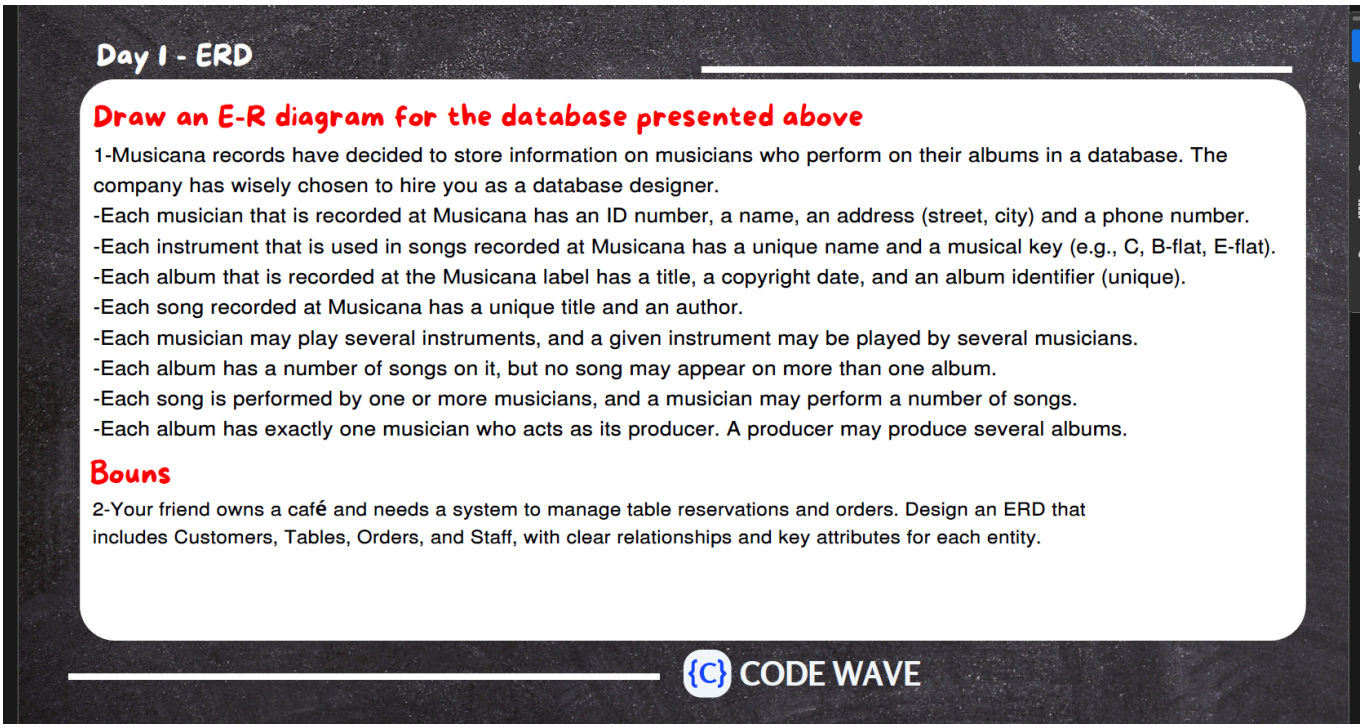
Code Wave Day1 & Day2

Task 1.1 :-



Solution :-

Entities and Attributes

1. **Musician**
    - **Attributes:**
        - `Musician_ID` (Primary Key)
        - `Name`
        - `Address` (Street, City)
        - `Phone_Number`
2. **Instrument**

- **Attributes:**
    - `Instrument_Name` (Primary Key)
    - `Musical_Key`

3. **Album**
    - **Attributes:**
        - `Album_ID` (Primary Key)
        - `Title`
        - `Copyright_Date`
4. **Song**
    - **Attributes:**
        - `Song_Title` (Primary Key)
        - `Author`

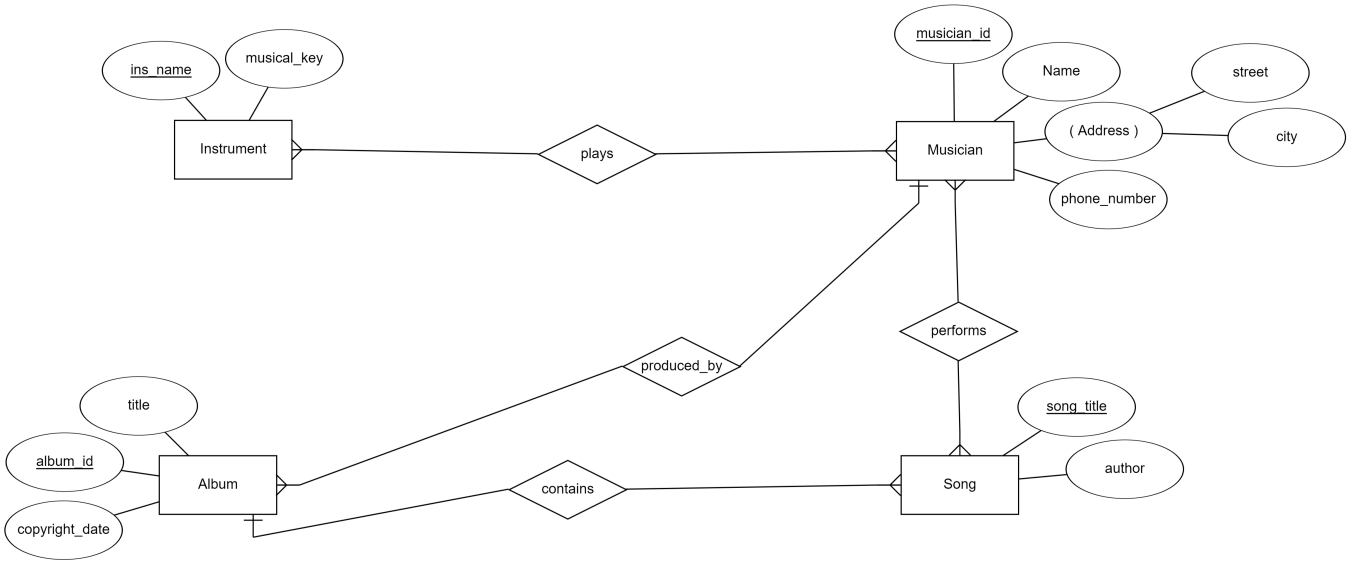
Relationships

1. **Plays (Many-to-Many between Musician and Instrument):**
    - A musician can play multiple instruments.
    - Each instrument can be played by multiple musicians.
    - **Bridge Table:** `Plays`
        - Attributes: `Musician_ID` (FK), `Instrument_Name` (FK)
2. **Performs (Many-to-Many between Musician and Song):**
    - A musician can perform multiple songs.
    - Each song can be performed by multiple musicians.
    - **Bridge Table:** `Performs`
        - Attributes: `Musician_ID` (FK), `Song_Title` (FK)
3. **Produced_By (One-to-Many between Musician and Album):**
    - Each album has one producer (a musician).

- A musician can produce multiple albums.
- **Foreign Key:** `Producer_ID` in `Album` (References `Musician_ID`)

4. **Contains (One-to-Many between Album and Song):**
    - An album contains multiple songs.
    - A song belongs to only one album.
    - **Foreign Key:** `Album_ID` in `Song` (References `Album_ID`)

Er diagram



Task 1.2 :- ERD for the Café System (bonus)

Entities and Attributes

1. **Customer**
    - **Attributes:**
        - `Customer_ID` (Primary Key)
        - `Name`

    - `Phone_Number`
    - `Email`
2. **Table**
    - **Attributes:**
        - `Table_ID` (Primary Key)
        - `Table_Number`
        - `Capacity` (Number of seats)
        - `Location` (e.g., indoors, outdoors)
3. **Order**
    - **Attributes:**
        - `Order_ID` (Primary Key)
        - `Order_Date` (Date and time of order)
        - `Total_Amount`
        - `Customer_ID` (Foreign Key referencing `Customer`)
4. **Staff**
    - **Attributes:**
        - `Staff_ID` (Primary Key)
        - `Name`
        - `Role` (e.g., Waiter, Manager, Chef)
        - `Phone_Number`
        - `Email`

Relationships

1. **Customer ↔ Table (Reserves):**

- A customer can reserve one or more tables.
- Each table can be reserved by multiple customers (at different times).
- **Bridge Table:** `Reservation`
  - Attributes: `Reservation_ID` (Primary Key), `Customer_ID` (FK), `Table_ID` (FK), `Reservation_Date`

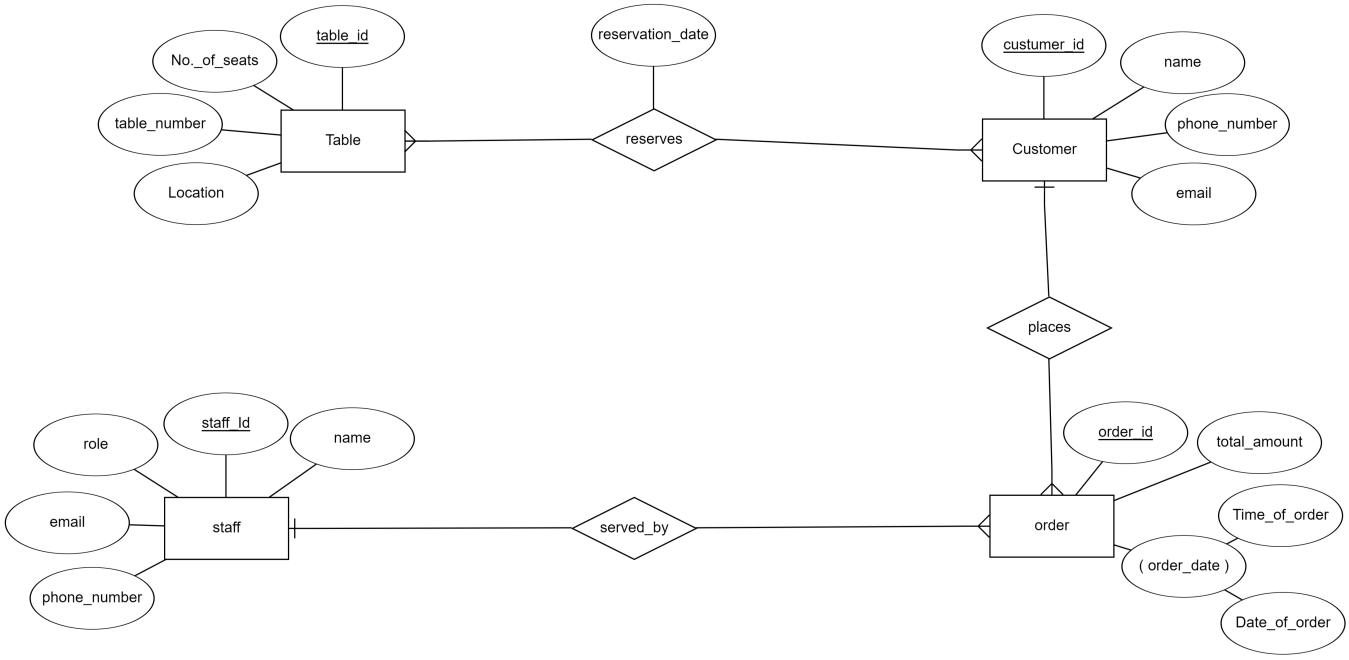
2. **Customer ↔ Order (belong):**
   - A customer can place multiple orders.
   - Each order belongs to one customer.
   - **Foreign Key:** `Customer_ID` in `Order` (References `Customer_ID`)
3. **Order ↔ Staff (Served_By):**
   - Each order is served by one staff member.
   - A staff member can serve multiple orders.
   - **Foreign Key:** `Staff_ID` in `Order` (References `Staff_ID`)

ERD Cardinality

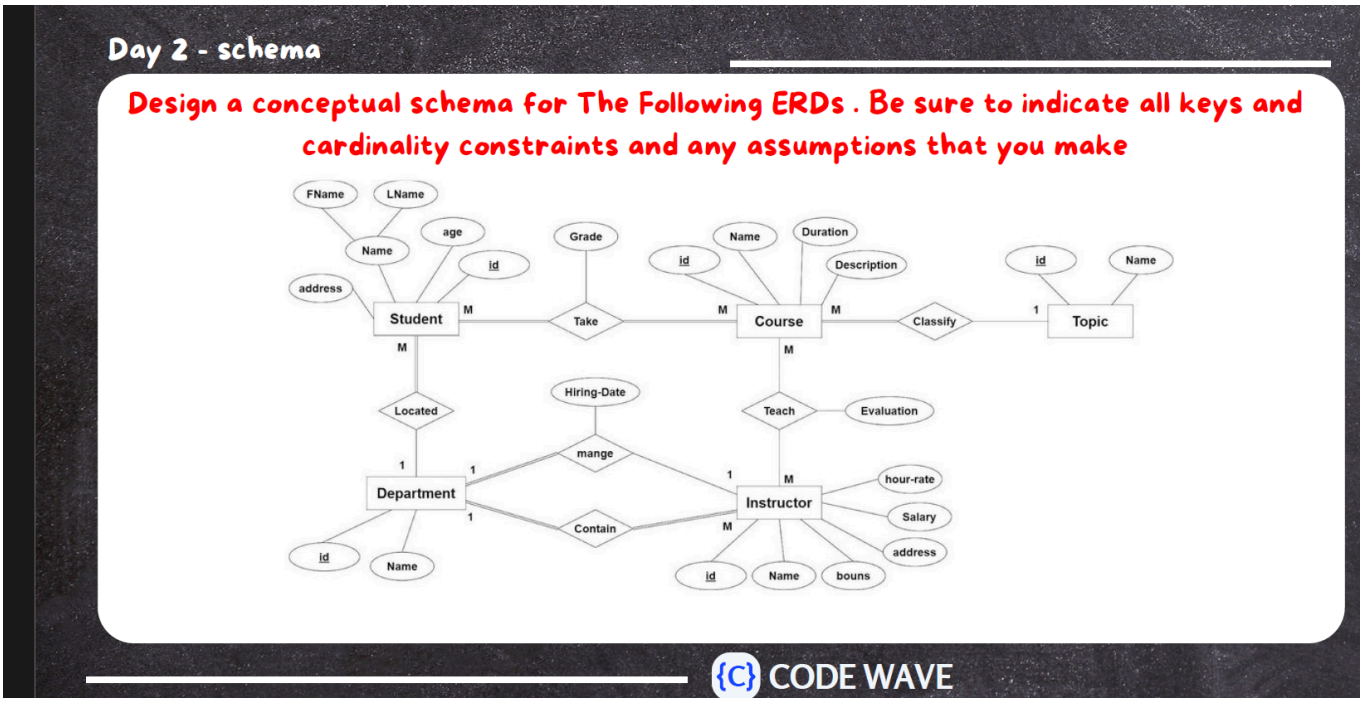
1. **Customer ↔ Table (Reserves):**
   - Crow's foot notation showing Many-to-Many resolved with the `Reservation` table.
2. **Customer ↔ Order:**
   - Crow's foot notation showing One-to-Many (Customer → Order).
3. **Order ↔ Staff (Served_By):**

- Crow's foot notation showing Many-to-One (Order → Staff).

Er diagram



Task 2 (Day 2)



Solution :-

Entities and Attributes

1. **Student**
    - **Attributes:**
        - `Student_ID` (PK)
        - `First_Name`
        - `Last_Name`
        - `Age`
        - `Address`
        - `Grade`
2. **Department**
    - **Attributes:**
        - `Department_ID` (PK)
        - `Department_Name`
3. **Course**
    - **Attributes:**
        - `Course_ID` (PK)
        - `Course_Name`
        - `Duration`
        - `Description`
4. **Instructor**
    - **Attributes:**
        - `Instructor_ID` (PK)
        - `Name`
        - `Hiring_Date`

        - `Address`
        - `Salary`
        - `Hourly_Rate`
5. **Topic**
    - **Attributes:**
        - `Topic_ID` (PK)
        - `Topic_Name`

Relationships and Cardinalities

1. **Student ↔ Course (Take)**
    - A student can take multiple courses, and each course can be taken by multiple students (Many-to-Many).
    - **Solution:**
        - Create a bridge table:
            - **Attributes:**
                - `Student_ID` (FK to Student)
                - `Course_ID` (FK to Course)
                - `Enrollment_Date`
2. **Instructor ↔ Course (Teach)**
    - An instructor can teach multiple courses, and a course can be taught by multiple instructors (Many-to-Many).
    - **Solution:**
        - Create a bridge table:
            - **Attributes:**
                - `Instructor_ID` (FK to Instructor)

- `Course_ID` (FK to Course)
- `Evaluation_Score`

3. **Course ↔ Topic (Classify)**
    - A course can have multiple topics, but each topic belongs to one course (One-to-Many).
    - **Solution:**
        - Add `Course_ID` as a Foreign Key in the `Topic` entity.
4. **Student ↔ Department (Located)**
    - A department can have multiple students, but each student belongs to only one department (One-to-Many).
    - **Solution:**
        - Add `Department_ID` as a Foreign Key in the `Student` entity.
5. **Instructor ↔ Department (Manage)**
    - A department can be managed by one instructor, but an instructor may manage multiple departments (One-to-Many).
    - **Solution:**
        - Add `Instructor_ID` as a Foreign Key in the `Department` entity.

Final Conceptual Schema

- **Student(Student_ID, First_Name, Last_Name, Age, Address, Grade, Department_ID (FK))**

- **Department(Department_ID, Department_Name, Instructor_ID (FK))**
- **Course(Course_ID, Course_Name, Duration, Description)**
- **Instructor(Instructor_ID, Name, Hiring_Date, Address, Salary, Hourly_Rate)**
- **Topic(Topic_ID, Topic_Name, Course_ID (FK))**
- **Take(Student_ID (FK), Course_ID (FK), Enrollment_Date)**
- **Teach(Instructor_ID (FK), Course_ID (FK), Evaluation_Score)**