Course: Databases

Lab: 3 (Group 2)

Date: 07.06.2025

Students:

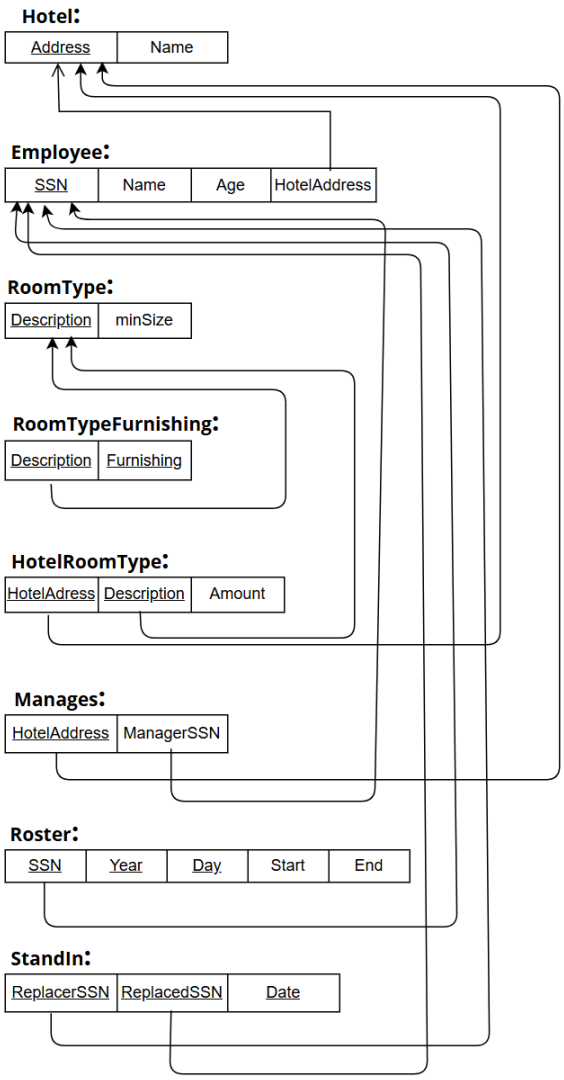
Name: Thilakraj Soundararajan
Matriculation No: 2705370

Name: Mir Md Redwon Sagor
Matriculation No: 2613747

Team No: 12

Assignment 1:

Relation Schema:



Assignment 2:

π lastName, dateOfBirth (σ dateOfBirth < '2000-01-01' (PLAYER))

π firstName, lastName (PLAYER ⋈ PLAYER.PID = GAME.winnerID (GAME))

Assignment 3:

1. **Full functional dependencies:**

   OrderID → CustomerID, Product, Quantity, Price

   CustomerID → Name, Address

2. In the table, we can see that only OrderID is unique for each order, and it determines all other attributes. Therefore, OrderID is a candidate key.
   Since there is only one candidate key, we select it as the primary key.

   Candidate Key: OrderID
   Primary Key: OrderID

3. From 1. we see that Name and Address depend on CustomerID, not directly on OrderID. This is a transitive dependency, which violates 3NF.
   So, we split the relation into two tables to remove this issue.

   Customer Table:
   Customer(CustomerID, Name, Address)
   PK: CustomerID

CustomerID	**Name**	**Address**
16	Petra Wagner	Lindenallee 7, Munich
18	Klaus Schmidt	Hauptstraße 4, Hamburg
24	Maria Müller	Wegstraße 12b, Berlin

Order Table:

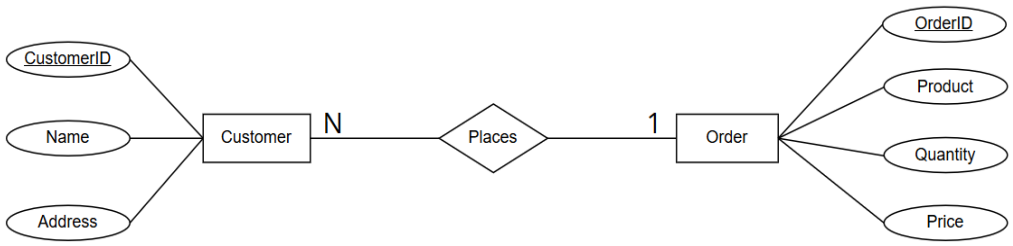
Order(OrderID, CustomerID, Product, Quantity, Price)

PK: OrderID

FK: CustomerID → Customer(CustomerID)

OrderID	Product	Quantity	Price	CustomerID(FK)
101	Table	2	150.0	24
102	Chair	5	80.0	24
103	Table	1	130.0	18
104	Sofa	2	200.2	16

4. Entity Relationship(ER) diagram of the 3NF Schema:



ER Diagram

4. Assignment 4: Functional dependencies and normalization of a Furniture Database Version 2

A furniture company maintains a database that records information about orders, customers, and products. Note: In version 2, each order may contain several products. The sample database relation ORDER is as follows:

CustomerID	Name	Address	OrderID	Product	Quantity	Price
24	Maria Müller	Wegstraße 12b, Berlin	101	Table	2	150.0
24	Maria Müller	Wegstraße 12b, Berlin	102	Chair	5	80.0
18	Klaus Schmidt	Hauptstraße 4, Hamburg	103	Table	1	130.0
16	Petra Wagner	Lindenallee 7, Munich	104	Sofa	2	200.0

1. Determine the **(full) functional dependencies**. Keep in mind that functional dependencies are determined by the model, not just by the actual data in the database relations.

   Ans:

   - **CustomerID → Name, Address**
     (A customer with a name and an address is uniquely identified by their ID.)

   - **OrderID → CustomerID**
     (Each order is placed by one customer.)

   - **OrderID, Product → Quantity, Price**
     (Each product in an order has a specific quantity and price.)

2. Determine potential candidate keys and a primary key for the given relation ORDER. Elaborate on your answer.

   Ans:

   - **OrderID** alone cannot be a key because a single order can contain multiple products.
   - **Product** alone cannot be a key since multiple orders can contain the same product.
   - But the combination **OrderID + Product** uniquely identifies each tuple.
   - So, the **Candidate Key** = {OrderID, Product}

     We can also select this as our **Primary Key**.

3. Transform the relational schema to 3NF. Your relation(s) should indicate PKs & FKs and contain all the data

   Ans:

   Customer(CustomerID PK, Name, Address)

   Order(OrderID PK, CustomerID FK)

   Product(Product PK)

   OrderDetails(OrderID PK, Product PK, Quantity, Price,
   FK OrderID REFERENCES Order,
   FK Product REFERENCES Product)

4. Implement the schema in your database and insert sample data.

Query | Query History

```sql
-- Table: Customer
CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(100),
    Address VARCHAR(200)
);

-- Table: Product
CREATE TABLE Product (
    Product VARCHAR(50) PRIMARY KEY
);

-- Table: Order
CREATE TABLE OrderTable (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
);

-- Table: OrderDetails
CREATE TABLE OrderDetails (
    OrderID INT,
    Product VARCHAR(50),
    Quantity INT,
    Price DECIMAL(10,2),
    PRIMARY KEY (OrderID, Product),
    FOREIGN KEY (OrderID) REFERENCES OrderTable(OrderID),
    FOREIGN KEY (Product) REFERENCES Product(Product)
);
```

```sql
-- Insert Customers
INSERT INTO Customer (CustomerID, Name, Address) VALUES
(24, 'Maria Müller', 'Wegstraße 12b, Berlin'),
(18, 'Klaus Schmidt', 'Hauptstraße 4, Hamburg'),
(16, 'Petra Wagner', 'Lindenallee 7, Munich');

-- Insert Products
INSERT INTO Product (Product) VALUES
('Table'),
('Chair'),
('Sofa');

-- Insert Orders
INSERT INTO OrderTable (OrderID, CustomerID) VALUES
(101, 24),
(102, 24),
(103, 18),
(104, 16);

-- Insert Order Details
INSERT INTO OrderDetails (OrderID, Product, Quantity, Price) VALUES
(101, 'Table', 2, 150.00),
(102, 'Chair', 5, 80.00),
(103, 'Table', 1, 130.00),
(104, 'Sofa', 2, 200.00);
```

5. Assignment 5: Relational Algebra vs. SQL query for a Weather Database

The following excerpt from a database schema models a database about a weather station. The following assignments are to be answered in the form of relational algebra.

- **WEATHER_DATA** (DataID, CityID (FK), Date, Temperature, Humidity, Precipitation, Wind-Speed)
- **CITY** (CityID, Name, Country)

1. Which cities have an average daily temperature above 25°C in August 2023?

Ans: Filtered the weather data for August 2023 and selected cities where the temperature exceeded 25°C. Joined with the CITY table to display city names.

AUGUST_DATA $\longleftarrow \sigma_{Date}$ = '2023-08-01' ∧ Date ≤ '2023-08-31'(WEATHER_DATA)

HOT_CITIES $\longleftarrow \sigma_{Temperaure > 25}$ (AUGUST_DATA)

RESULT $\longleftarrow \pi_{Name}$(HOT_CITIES ⋈ CITY)

2. Which cities experienced no precipitation on any day in July 2023?

   Ans: Identified cities with no rainfall during July 2023 by subtracting cities that had any precipitation from the full list. Joined the result with CITY to show names.

$$\text{JULY\_DATA} \leftarrow \sigma_{Date} \geq \text{'2023-07-01'} \wedge \text{Date} \leq \text{'2023-07-31'}(\text{WEATHER\_DATA})$$

$$\text{RAINY\_CITIES} \leftarrow \pi_{CityID}\ (\sigma_{Precipitation > 0}\ (\text{JULY\_DATA}))$$

$$\text{ALL\_CITIES} \leftarrow \pi_{CityID}(\text{JULY\_DATA})$$

$$\text{NO\_RAIN\_CITIES} \leftarrow \text{ALL\_CITIES} - \text{RAINY\_CITIES}$$

$$\text{RESULT2} \leftarrow \pi_{Name}\ (\text{NO\_RAIN\_CITIES} \bowtie \text{CITY})$$

3. On which particular day did the cities have the highest wind speed?

   Ans: Compared all wind speed records to find dates with no higher values in the dataset. These are the dates with the highest recorded wind speed.

$$\text{RESULT3} \leftarrow \pi_{Date}\ (\text{WEATHER\_DATA}) - \pi_{Date}\ (\sigma_{\text{WEATHER\_DATA.WindSpeed} < \text{HIGH\_WIND.WindSpeed}}(\text{WEATHER\_DATA} \times \rho_{\text{HIGH\_WIND}}\ (\text{WEATHER\_DATA})))$$

4. Which cities recorded the highest temperature?

   Ans: Detected cities that recorded the highest temperature by removing those with any lower temperature. The result shows only the cities with maximum temperature.

$$\text{RESULT4} \leftarrow \pi_{Name}\ (\text{WEATHER\_DATA} \bowtie \text{CITY}) - \pi_{Name}\ (\sigma \text{WEATHER\_DATA.Temperature} < \text{HIGH\_TEMP.Temperature}\ ((\text{WEATHER\_DATA} \bowtie \text{CITY}) \times \rho_{\text{HIGH\_WIND}}\ (\text{WEATHER\_DATA} \bowtie \text{CITY})))$$

5. Which cities had the lowest humidity in May 2023?

Ans: Filtered records for May 2023, compared humidity values, and excluded cities with higher humidity. The result includes cities with the lowest humidity for that month

MAY_DATA $\longleftarrow \sigma_{Date}$ ≥ '2023-05-01' ∧ Date ≤ '2023-05-31'(WEATHER_DATA)

RESULT5 $\longleftarrow \pi_{Name}$ (MAY_DATA ⋈ CITY) − π_{Name} ($\sigma_{MAY\_DATA.\text{Humidity} > LOW\_HUM.Humidity}$

((MAY_DATA ⋈ CITY) × ρ_{LOW_HUM}(MAY_DATA ⋈ CITY)))