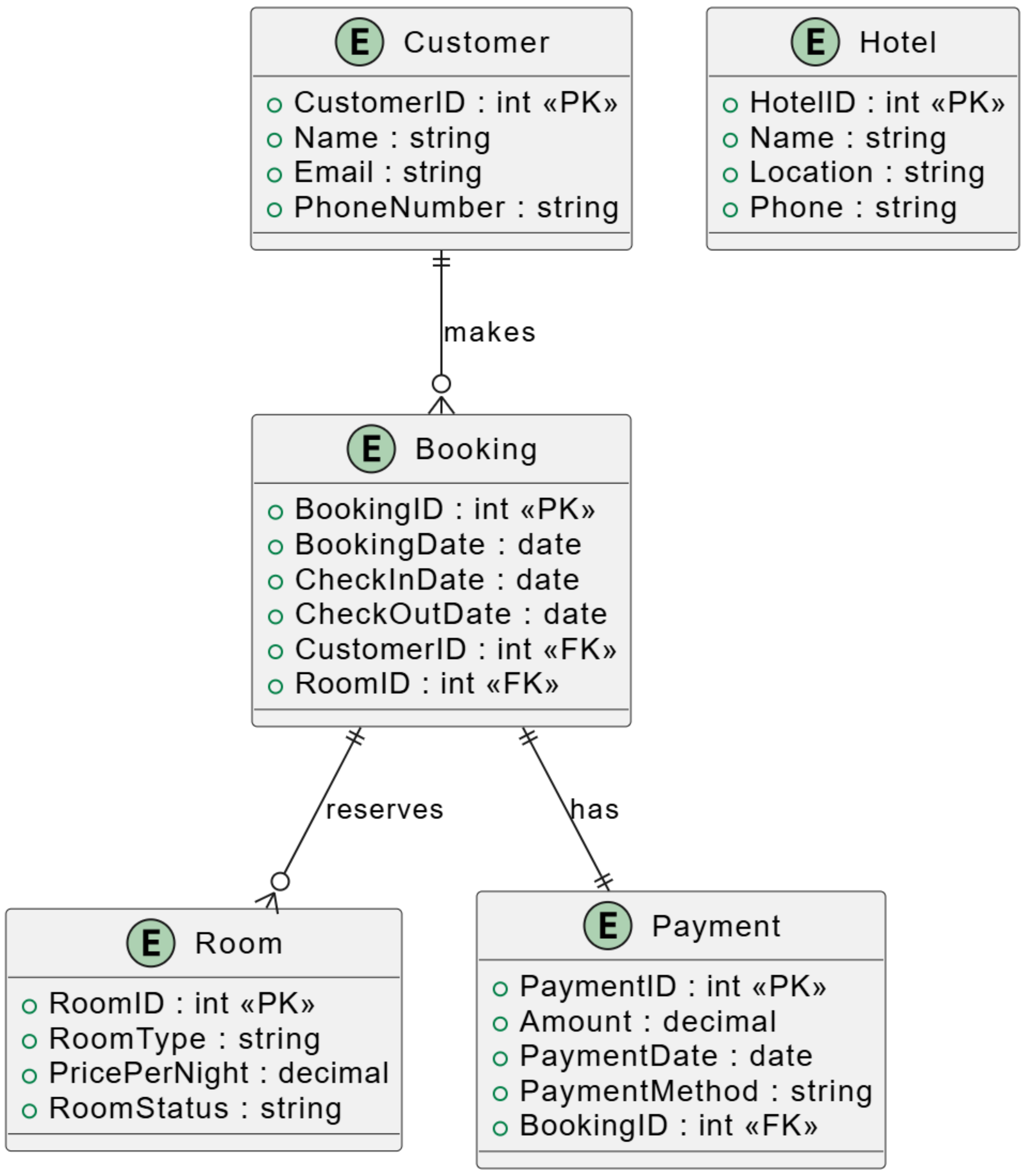
Entity Relationship Diagrams



An Entity Relationship Diagram (ERD) is a tool used in database design to visually represent the relationships between data entities in a system. ERDs are crucial in planning and organising the structure of a database before the development begins, ensuring that data is accurately stored and accessible for efficient use.

What is an Entity Relationship Diagram?

An **Entity Relationship Diagram (ERD)** is a diagrammatic representation that outlines the key entities within a database and their relationships. ERDs provide a structured approach to database design by allowing developers to see how data is interrelated, helping prevent redundancy and ensuring proper linkage of data. ERDs are composed of three main components:

1. **Entities**: These represent objects or concepts that have a distinct existence, like "Customer" or "Room." In an ERD, entities are typically represented by rectangles.
2. **Attributes**: Each entity has characteristics or properties called attributes. For example, a "Customer" entity may have attributes like "CustomerID," "Name," and "Email."
3. **Relationships**: These define how entities relate to one another, such as "Customer books Room." Relationships are represented by diamonds or simple lines connecting entities.

Steps to Develop an ERD for a Hotel Booking System

When creating an ERD for a hotel booking system, the goal is to outline all the essential components and how they interact to support the booking process.

Step 1: Identify the Key Entities

Start by listing all the entities required for a hotel booking system. Some primary entities might include:

- **Customer**: Represents individuals who book hotel rooms.
- **Room**: Represents the different rooms available for booking.
- **Booking**: Tracks each room booking made by a customer.
- **Payment**: Captures payment details associated with bookings.
- **Hotel**: Represents the hotel itself, especially if it has multiple branches.

Each of these entities will play a significant role in the system, so ensure you thoroughly understand what data each one will need to store.

Step 2: Define the Attributes for Each Entity

After identifying the entities, determine what information needs to be stored within each one. Here's an example breakdown:

- **Customer**:

    - CustomerID (Primary Key)
    - Name
    - Email
    - PhoneNumber
- **Room**:
    - RoomID (Primary Key)
    - RoomType (e.g., Single, Double, Suite)
    - PricePerNight
    - RoomStatus (e.g., Available, Booked)
- **Booking**:
    - BookingID (Primary Key)
    - BookingDate
    - CheckInDate
    - CheckOutDate
    - CustomerID (Foreign Key linking to Customer)
    - RoomID (Foreign Key linking to Room)
- **Payment**:
    - PaymentID (Primary Key)
    - Amount
    - PaymentDate
    - PaymentMethod (e.g., Credit Card, Cash)
    - BookingID (Foreign Key linking to Booking)
- **Hotel**:
    - HotelID (Primary Key)
    - Name
    - Location
    - Phone

Step 3: Establish Relationships Between Entities

Once entities and attributes are defined, determine the relationships. For a hotel booking system:

- **Customer – Booking**: A customer can have multiple bookings over time, but each booking belongs to only one customer. This relationship is one-to-many.
- **Booking – Room**: Each booking is linked to a specific room, but a room may have multiple bookings across different dates. This relationship is also one-to-many, with constraints ensuring availability on specific dates.
- **Booking – Payment**: Each booking has one payment associated with it, but payments directly link to one booking. This is a one-to-one relationship.

Step 4: Draw the ERD

Using an ERD tool or diagram software:

1. Draw rectangles for each entity.
2. Add ovals for each attribute, linking them to their respective entities.
3. Draw diamonds (or labelled lines) between entities to indicate relationships, marking them as one-to-one, one-to-many, or many-to-many where relevant.
4. Connect entities with primary and foreign keys to establish linkage. For example, CustomerID should appear as a foreign key in the Booking entity.

Here's a rough outline of what the ERD could look like for the hotel booking system:

- **Customer** connects to **Booking** with a one-to-many line, meaning each customer can make multiple bookings.
- **Booking** connects to **Room** with a one-to-many line, allowing each room to be booked many times but only once per booking.
- **Booking** connects to **Payment** in a one-to-one relationship, where each booking has a corresponding payment.

Step 5: Review and Refine

After creating the initial ERD, review it for completeness. Check if:

- All required entities and relationships are included.
- Primary and foreign keys are correctly designated.
- The attributes are specific enough to support system functionality.

Benefits of an ERD in a Hotel Booking System

Developing an ERD provides several advantages, including:

- **Clear Data Organisation**: Ensures that data storage is logically organised.
- **Improved Data Consistency**: Reduces redundancy and supports data integrity.
- **Efficient Database Development**: Provides a blueprint for developers to implement the database structure, reducing development time.

Conclusion

Creating an ERD is an essential step in designing an effective hotel booking system. By identifying entities, defining attributes, and establishing relationships, an ERD ensures a well-structured database capable of supporting complex data interactions. For a hotel booking system, an ERD lays the foundation for efficiently managing customers, rooms, bookings, and payments, providing an organised structure for the backend of a booking website.