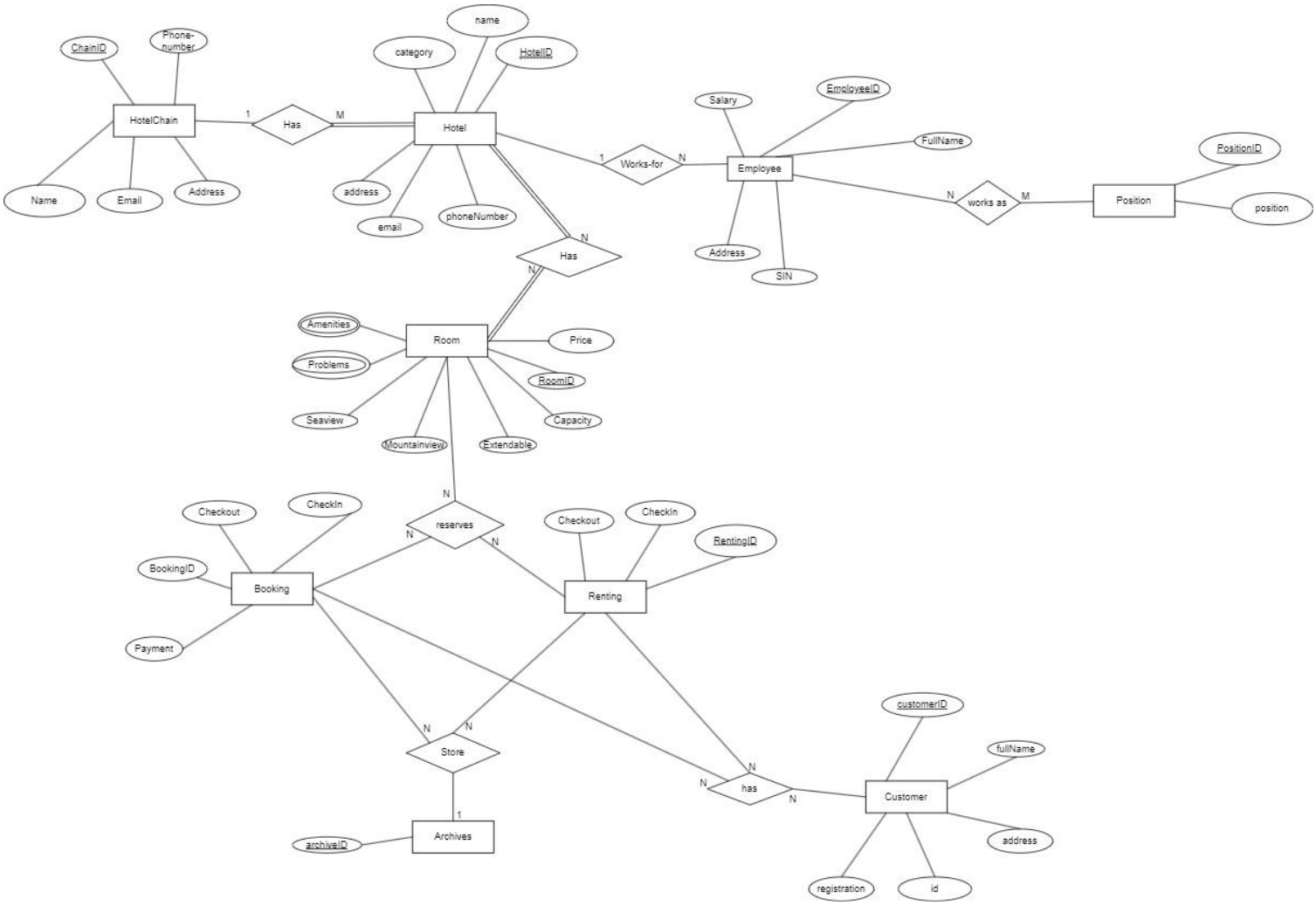
1 a) ER-Diagram



Justification:

- Amenities and Problem attributes are multi-valued, as they can have different kinds of values, so they were assigned as multi-valued with the double rings to account for this. For example, there can be multiple kinds of problems, so we wanted to account for this.
- The cardinality between Employee and position has the many-to-many relation as employees can have multiple positions, so we wanted to make sure this case is covered in our database.

- Customer can have many Rentings and Booking, so this is accounted for in the relation between customer and the entities “Renting” and “Booking”.
- The relation between the entities Room and Hotel is a total participation relation, because in order for a Hotel to exist, they must have Rooms.
- Hotel has a total participation relationship with HotelChain because a hotel has to be associated with a HotelChain.
- The relationship between Booking and Rooms is that one Room can have multiple bookings, as long as it’s not at the same time as another. This same principle applies to renting.
- The relation between Customer and Bookings is that many Customers can have many bookings, which is why it is a many-to-many relation.
- The Booking and Renting entities have a many-to-one relation with the Archive as we are storing multiple Bookings and Renting’s in a single archive.

1 b) Relational Schema

Primary Keys are underlined.

Entity Sets:

- Hotel Chain (ChainID, Name, CentralAddress, number-of-hotel, Email, Phone-number)
- Hotel (HotelID, ChainID, Name, HotelAddress, Email, Phone-Number, Category)
- Room(Room ID, HotelID, Price, Seaview, Mountainview, Capacity, Extendable)
- Room_Amenitites(RoomID, Amenities)
- Room_Problems(RoomID, Problems)
- Booking(BookingID, RoomID, CustomerID, Checkout, CheckIn, Payment)
- Renting(RentingID, RoomID, CustomerID, Checkout, CheckIn)
- Customer(CustomerID, FullName, Address, Registration, ID)
- Employee(EmployeeID, HotelID, Fullname, Salary, SIN, Address)
- Position(PositionID, EmployeeID, JobTitle)
- Archives(Archived, CustomerID, RentingID, BookingID)

Justification:

Entity sets are all the information of users and requirements of the system and the attributes of the entity sets will be how the necessary information is retrieved when a user goes to book or rent a hotel room.

Foreign Keys (Justification is list below)

- Hotel.ChainID to HotelChain.ChainID
    - To show what hotel chain the hotel belongs to.
- Room.HotelID to Hotel.HotelID
    - To show what room the hotel belongs to.
- Room_Amenities.RoomID to Room.RoomID
    - To show what amenities are associated with the room.
- Room_Problems.RoomID to Room.RoomID
    - To show what problems are associated with the room.
- Booking.RoomID to Room.RoomID
    - To show what room is being booked by the customer.
- Booking.CustomerID to Customer.CustomerID
    - To show what customer has the booking.
- Renting.RoomID to Room.RoomID
    - To show what room is being rented out.
- Renting.CustomerID to Customer.CustomerID
    - To show what room the customer is renting.
- Archives.CustomerID to Customer.CustomerID
    - To store the customers' information that had the room.
- Archives.RentingID to Renting.RentingID
    - To store the rental information.
- Renting.BookingID to Booking.BookingID
    - To store the past Booking Information.
- Position.EmployeeID to Employee.EmployeeID
    - To know what Employee is associated with the specific position.

Relationship sets: -

Attributes of theses sets come from the primary keys of the entities belonging to the relationship set to show the connection between Keys.

Has(ChainID, HotelID)

Has.ChainID to HotelChain.ChainID

Has.HotelID to Hotel.hotelID

Works-For(HotelID, EmployeeID)

Works-For.HotelID to Hotel.HotelID

Works-For.EmployeeID to Employee.EmployeeID

Has(HotelID, RoomsID)

Has.HotelID to Hotel.ID

Has.RoomsID to Rooms.RoomsID

Works-as(EmployeeID, PositionID)

Works-as.EmployeeID to Employee.EmployeeID

Works-as.PositionID to positionID

Has(CustomerID, RentingID, BookingID)

Has.CustomerID to Customer.CustomerID

Has.RentingID to Renting.RentingID

Has.BookingID to Booking.BookingID

Store(ArchiveID, RentingID, BookingID, CustomerID)

Store.archive ID to Archive.ArchiveID

Store.RentingID to renting.RentingID

Store.BookingID to Booking.bookingID

Store.CustomerID to Customer.CustomerID

Reserves(RoomID, BookingID, RentingID)

Reserves.RoomID to Room.RoomID

Reserves.BookingID to Booking.BookingID

Reserves.RentingID to Renting.RentingID

1 c) Integrity constrants

PRIMARY KEYS

chainID is the primary key in HotelChain

hotelID is the primary key in Hotel

roomID is the primary key in Room

bookingID is the primary key in Booking

rentingID is the primary key in Renting

customerID is the primary key in Customer

employeeID is the primary key in Employee

positionID is the primary key in Position

archiveID is the primary key in archive

These values cannot be null

==========

FOREIGN KEYS (Referential Integrity)

hotelID in Hotel -> hotelID in HotelChain

hotelID in Room -> hotelID in Hotel

roomID in Booking -> roomID in Room

customerID in Booking -> customerID in Customer

roomID in Renting -> roomID in Room

customerID in Renting -> customerID in Customer

hotelID in Employee -> hotelID in Hotel

employeeID in Position -> employeeID in Employee

==========

Domain Constraints

checkInDate and checkOutDate in Booking and Renting should be valid – ensures that the dates provided by the user for check in and check out are valid and within range

Renting should be valid dates – ensures that the renting date should be valid to maintain consistency

price in bookingID should be positive – ensures that the price associated with bookingID is a valid financial transaction

hotel always has a manager – ensures that there is always a manager for the hotel for decision making and proper management

Assume payment isn't stored – ensures that the users private info isn't stored to the database

The roomID can only have one booking in a certain check-in and check-out date (room cannot have more than 1 booking at a time).

==========

Attribute Constraints

Hotel category is between 1 and 5 stars – a standardized system for rating. Ensures that the rating system is consistent with other brands

SIN must be 9 digits - ensures a specific format for SIN

==========

User-Defined Constraints

checkInDate in Booking is before checkOutDate – ensures consistent logic with dates

length of stay is nonnegative – prevents unrealistic scenarios

Sample tables showing how the database will look

Sample tables

HotelChain Table:

chainID	name	officeAddress	numberOfHotels	email	phoneNumber
1	Chain A	Address A	20	chaina@email.com	123-456-7890
2	Chain B	Address B	15	chainb@email.com	987-654-3210

Hotel Table:

hotelID	chainID	name	category	address	email	phoneNumber
101	1	Hotel 1A	4-star	Address 1A	hotel1a@email.com	111-111-1111
102	1	Hotel 1B	3-star	Address 1B	hotel1b@email.com	222-222-2222
201	2	Hotel 2A	5-star	Address 2A	hotel2a@email.com	333-333-3333

Room Table:

roomID	*hotelID*	price	capacity	seaView	mountainView	extendable	Amenities	problems
1001	101	150	Double	Yes	No	Yes	TV	None
1002	101	120	Single	No	Yes	No	Fridge	Broken TV
2001	201	300	Suite	Yes	Yes	Yes	Air conditioning	None

Booking Table:

bookingID	customerID	roomID	checkInDate	checkOutDate	payment
8001	20001	1001	2024-03-01	2024-03-05	400
8002	20002	2001	2024-04-10	2024-04-15	800

Customer Table:

customerID	fullName	address	idType	registrationDate
20001	John Doe	Address X	SSN	2023-12-01
20002	Jane Smith	Address Y	Driving License	2024-01-15

Employee Table:

employeeID	hotelID	fullName	address	salary	sin
30001	101	Manager A	Manager's Addr	50000	111-222-333

30002	201	Manager B	Manager's Addr	90000	444-555-666

Position Table:

positionID	employeeID	position
60001	30001	Manager
60002	30002	Manager

Renting Table:

rentingID	customerID	roomID	checkInDate	checkOutDate
90001	20003	1002	2024-05-20	2024-05-25
90002	20004	2001	2024-06-15	2024-06-20

Archives Table:

archivesID	customerID	bookingID	rentingID
10001	20003	8001	90001
10002	20004	8002	90002