WolfVilla Hotels
Database Management System

CSC 440 Database Management Systems

Project Report #1

Andrew Poe, Felix Kim, John Hutcherson, Guoyang Di

1) Assumptions and Problem Statement

Assumptions

1. A presidential suite has dedicated room service and catering staff as required, but other room types can also have them if the customer opts in at check in.
2. “Payment info” only consists of: payment method, card umber, billing address, and SSN of payer.
3. The phone, laundry, and restaurant bills are treated as tabs, which the customer can accrue expenses in over their visit, while room service and catering are treated as services that are either provided for the entire visit, or not, and as such are opted into at check in time.
4. A hotel has at most one manager, and a manager manages exactly one hotel.
5. The hotel name is not necessarily unique among different hotels.
6. Catering and room service are two different services.
7. The “staff ID” is unique for every staff member through the entire hotel chain, assigned by a global authority for the hotel chain.
8. Similarly, each customer has a unique ID throughout the entire chain.
9. A customer is able to check into as many hotel rooms as he/she wants. At check-in, the customer says how many people they wish to check into a room (but a room has at most one listed customer). This gives the database the ability to handle situations where a customer has more than 4 people in their party.
10. Pricing is static year round.

Problem Statement

The task is to design a database management system for Wolf Villas, a hotel chain with hotels in various cities. It will have information about hotels, rooms, staff, customers, billing, and check-ins accessible to its numerous users from hotel management to regular hotel staff to customers for efficient queries and updates.

A database would be a good idea for the task for numerous reasons. For example, it holds all of the information in a centralized manner, and is much simpler to query than a set of files. A database is quicker than using text files when accessing data since it does not have to go through many lines to find the data. Without a database, all tasks have to be done manually, which would take much more time. A DBMS also reduces redundancy of information which saves data storage space and reduces costs for the hotel chain. At the same time, the supported storage of a DBMS is so large that it can serve a huge amount of data for the hotels. Moreover, that data can persist over long periods of time and the DBMS is durable in response to failures that will inevitably occur with its frequent usage in the day-to-day operations of the hotels. Furthermore, the hotel DBMS allows its multiple users to access and change data in the database concurrently with some kinds of constraints to those users to safely and gracefully handle inevitable collisions in their actions.

2) User Classes

- **Customers**: Guest of the hotel, and stays in a room. Accrues laundry, phone, and restaurant bills, and purchases the services of the room service and catering staff.

- **Managers**: Sees overall hotel information, including detailed personnel information, along with information related to how full the hotel is and how much revenue the hotel is bringing in.

- **Front-desk representatives**: Registers a customer, and processes check ins and check outs. Also manages customer billing.

- **Room service staff**: These staff function as a butler for the customers they are assigned. They are on call for the customers they are assigned during the hours they work, and handle just about any request that the guest has.

- **Catering staff**: These staff provide food to the guests they are assigned, and function similarly to an on demand personal chef.

3) The Main Entities

- **Room information**: room number, room rate, max occupancy, room category
- **Hotel information**: hotel ID, hotel name, phone number, address, manager ID
- **Staff information**: staff ID, name, SSN, gender, age, phone number, address, job title
- **Customer information**: customer ID, name, address, gender, phone number, email address
- **Billing information**: SSN of payer, name, billing address, payment method, card number
- **Check in information**: room number, number of guests, start date, start time, end date, end time

4) Usage Situations

- A customer wants to change their room. To do this, the front desk person will need to check to see if the desired room is empty, assign them to that room, update information of the customer with the new room and the former room the customer had, and release the former room.
- Generating a billing account. To do this, the billing staff must check the room to ensure it is in the hotel and the information of the customer(s) who previously lived in that room for validation, then the billing account can be created.

5) Application Program Interface

Staff Info Operations

newHotelStaff(ssn, name, jobTitle, gender, phoneNumber, address, age)

Return staffID or NULL for error

If NULL for ssn, name, or jobTitle then do not add new hotel staff

updateHotelStaff(staffID, name, ssn, gender, phoneNumber, jobTitle, address, age)

Return confirmation

If NULL for staffID then do not update hotel staff

deleteHotelStaff(staffID)

Return confirmation

If NULL for staffID then do not delete hotel staff

Room Info Operations

newRoom(hotelID, roomNumber, roomLimit, costPerNight, availability, roomCategory, roomSpecialServices)

Return confirmation

If NULL for any of these fields then do not create new room

updateRoom(hotelID, roomNumber, roomLimit, costPerNight, availability, roomCategory, roomSpecialServices)

Return confirmation

If NULL for hotelID or roomNumber then do not update room

deleteRoom(hotelID, roomNumber)

Return confirmation

If NULL for hotelID or roomNumber then do not delete room

checkRoomAvailability(hotelID, roomNumber)
    Return availability
    If NULL for hotelID or roomNumber then return error message

assignRoom(hotelID, roomNumber, customerID)
    Return confirmation
    If NULL for any of these fields then do not assign room

releaseRoom(hotelID, roomNumber)
    Return confirmation
    If NULL for hotelID or roomNumber then do not release room

Customer Info Operations

newCustomer(name, address, phoneNumber, emailAddress, gender)
    Return customerID
    If NULL for name or address then do not create new customer

updateCustomer(customerID, name, address, phoneNumber, emailAddress, gender)
    Return confirmation
    If NULL for customerID then do not update customer

deleteCustomer(customerID)
    Return confirmation
    If NULL for customerID then do not delete customer

Service/Billing Info Operations

newService(hotelID, roomNumber, hasRoomService, hasCatering, restaurantBill, laundryBill, phoneBill)
    Return confirmation
    If NULL for hotelID, roomNumber, hasRoomService, or hasCatering then do not create new service.

updateService(hotelID, roomNumber, restaurantBill, laundryBill, phoneBill, hasRoomService, hasCatering)
    Return confirmation
    If NULL for hotelID or roomNumber then do not update service

deleteService(hotelID, roomNumber)
    Return confirmation
    If NULL for hotelID or roomNumber then do not delete service

newBillingAccount(hotelID, roomNumber, customerID, ssn, name, billingAddress, paymentMethod, cardNumber)

Return confirmation

If NULL for hotelID, roomNumber, customerID, ssn, paymentMethod do not create new.

Note: uses checkRoomAvail() before new billing account generated for a customer

updateBillingAccount(hotelID, roomNumber, customerID, ssn, name, billingAddress, paymentMethod, cardNumber)

Return confirmation

If NULL for hotelID, roomNumber, customerID then do not update billing account.

deleteBillingAccount(hotelID, roomNumber, customerID)

Return confirmation

If NULL for hotelID, roomNumber, customerID then do not delete billing account.

Reporting Operations

reportHotelOccupancy(roomType, beginDate, endDate, hotelID)

Return list of room occupancy values (roomNumber, hotelNumber, occupancyNumber) for relevant rooms.

reportHotelOccupants()

Return list of occupants of hotel (customerID, name, gender, phone, address, email, numberGuests).

reportHotelOccupancyPercent()

Return percentage of rooms occupied in hotel.

reportStaffInfo(jobTitle)

Return list of staff members with the specific job title (all attributes of HotelStaff relation is used)

reportCustomers(staffID)

Return list of customers assigned to the particular staff member (all attributes of Customer relation is used)

6) Data views

Customer: The customer sees their personal information, room number, room category, check-in information, and billing information.

Manager: The manager can see all of the rooms, check in information of the hotel, along with limited customer information. The manager does not need to see the in-depth details of customer information, just enough to generate reports. He/she can also see how catering and room service staff are assigned.

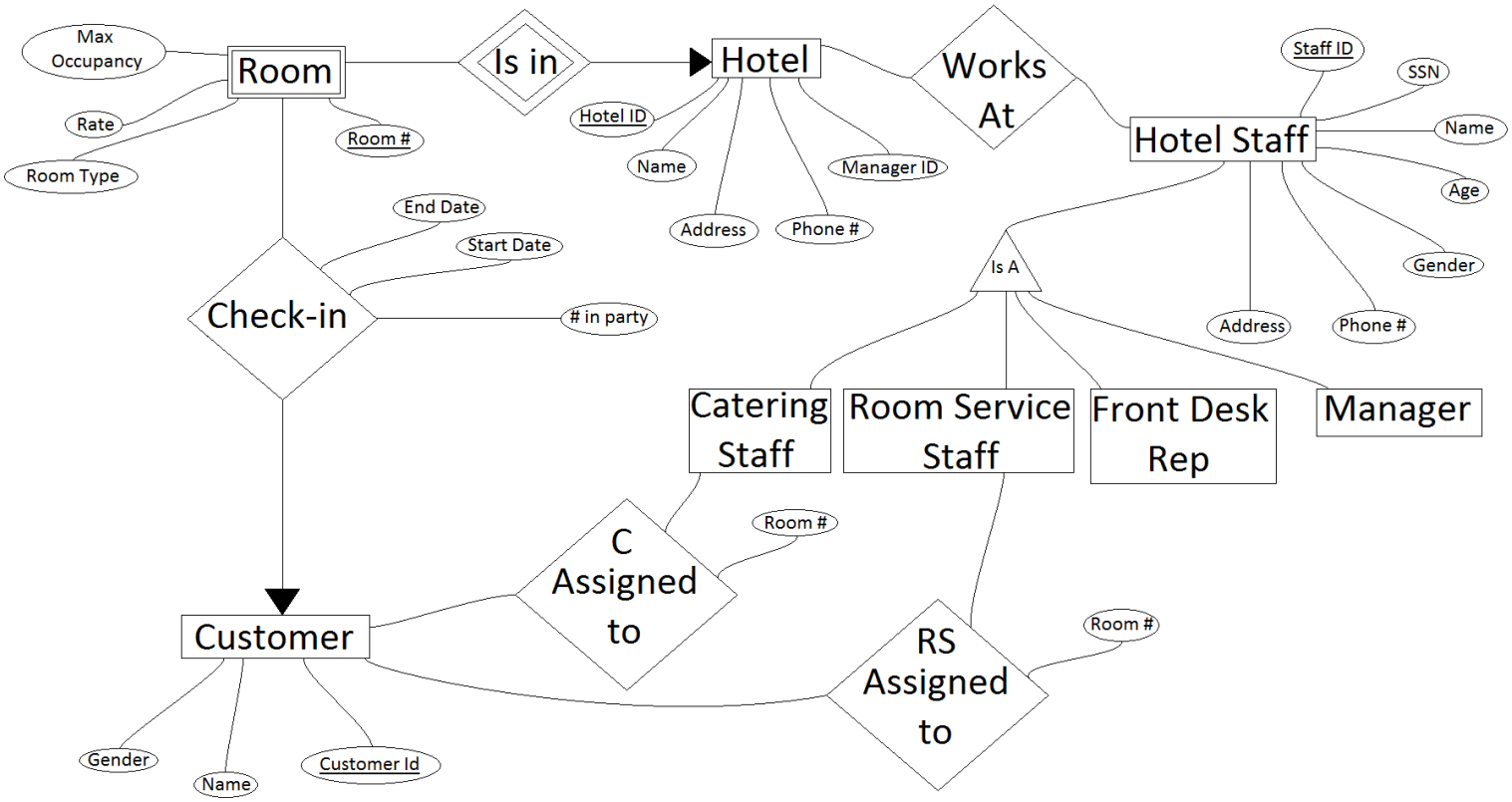
Caterer: The caterer can see a list of customers to which they are assigned, and the information necessary to address them properly.

Room Service staff: The room service staff can see a list of customers to which they are assigned, and the information necessary to address them properly.

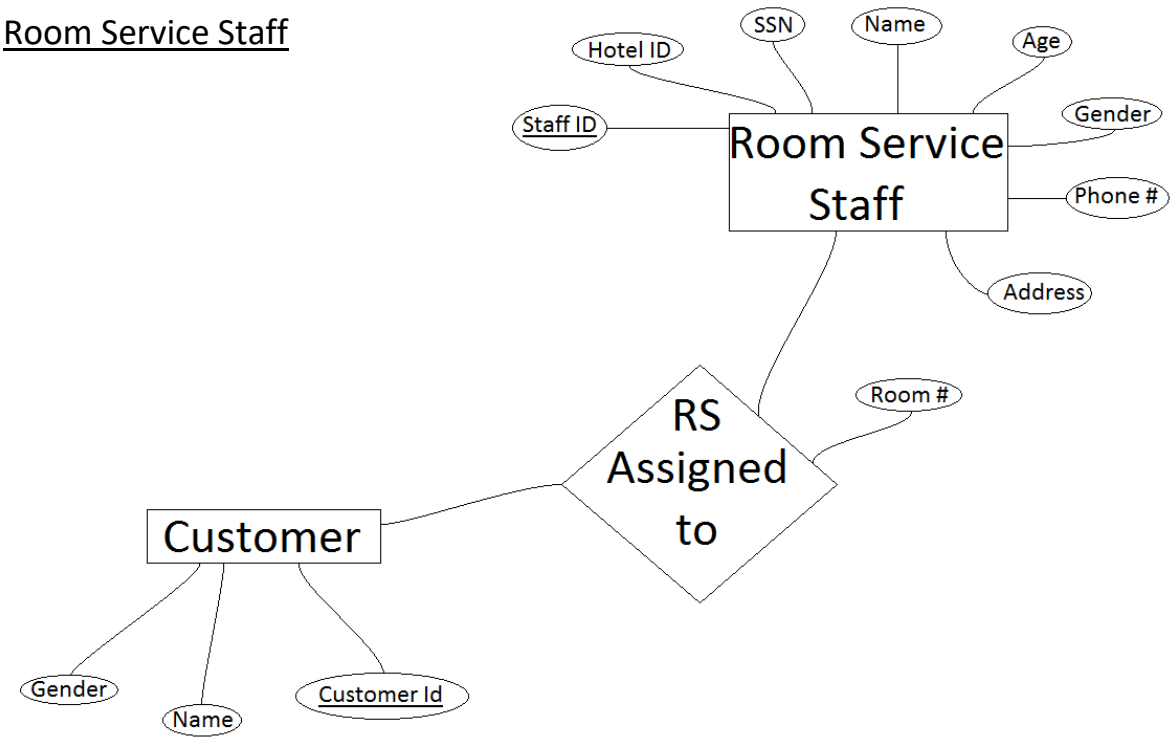
Front desk representative: Sees a list of rooms, and customers which are assigned to each room, along with all of the information about the customers. In addition, sees all check in and billing information about the customers. Also sees how catering / room service staff members are allocated to the customers.

7) Local E/R Diagrams

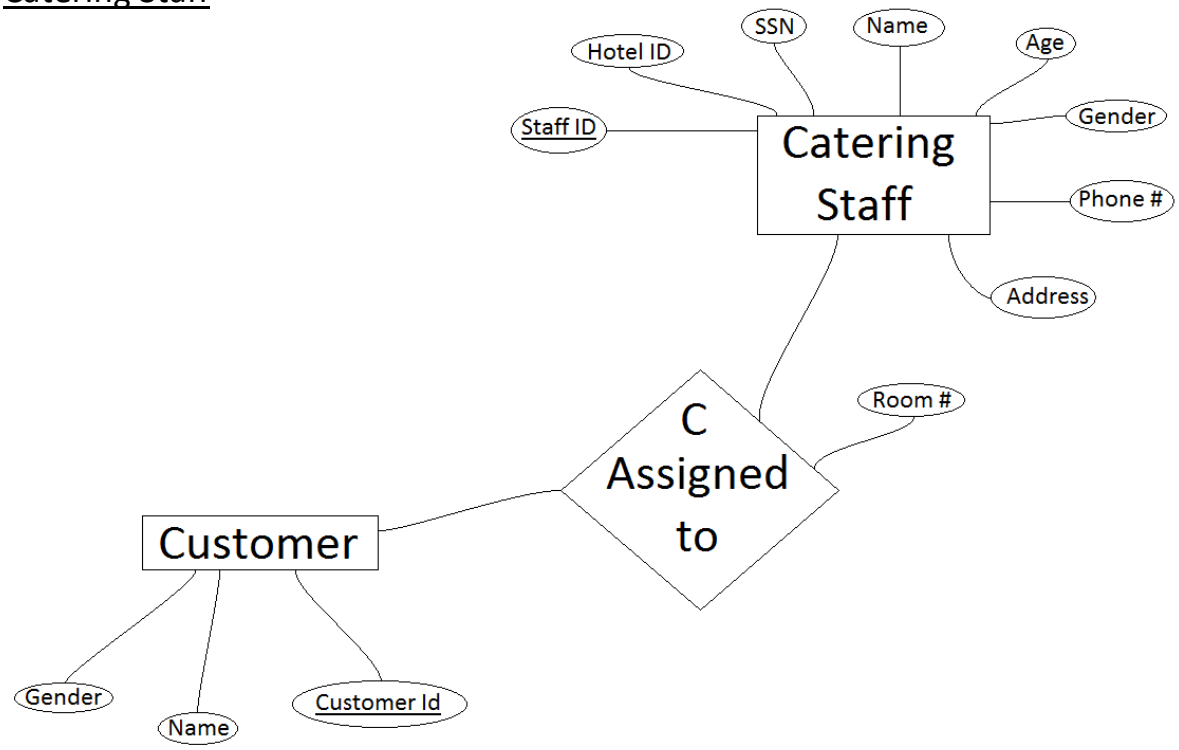
Manager



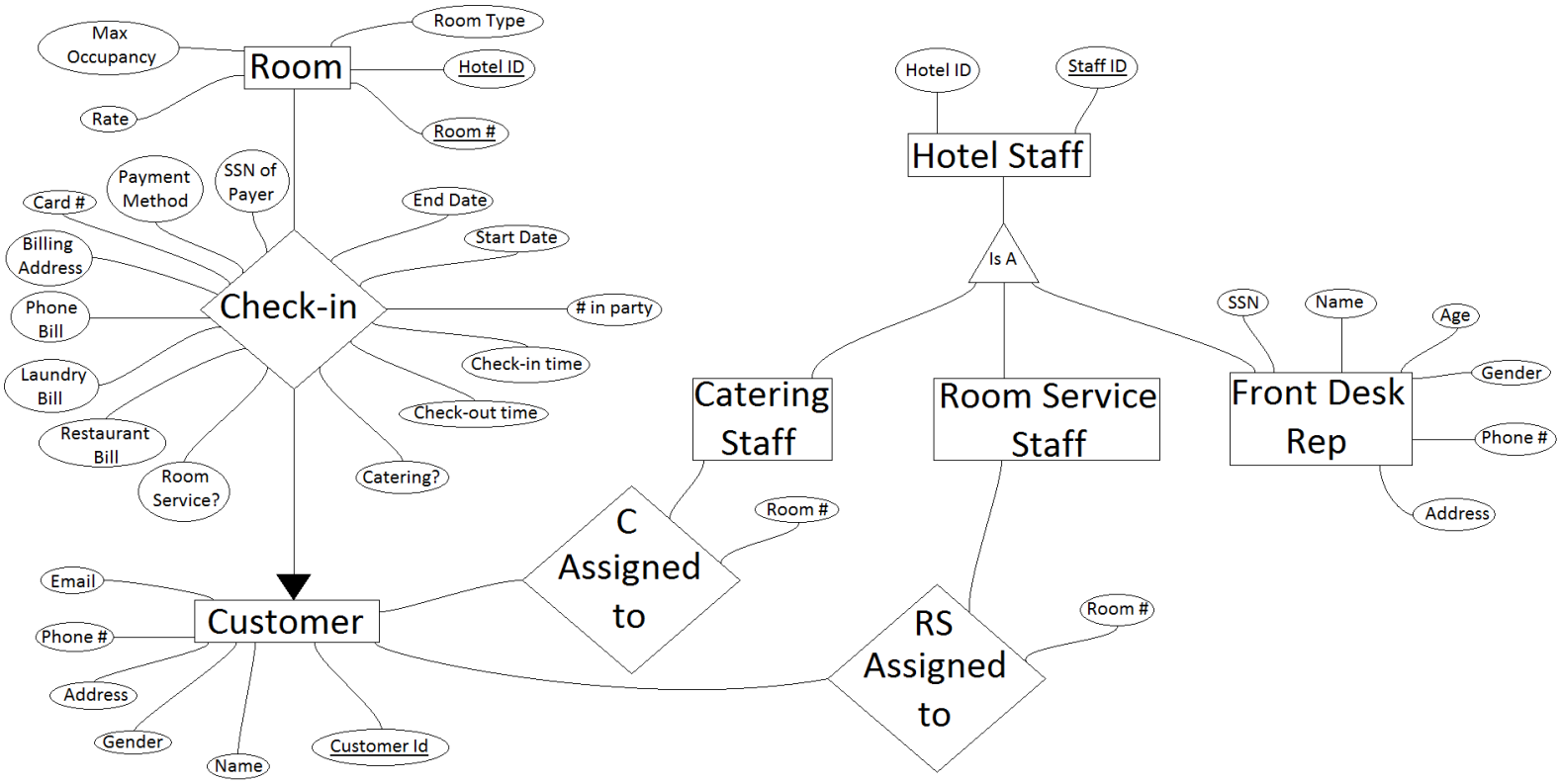
Room Service Staff



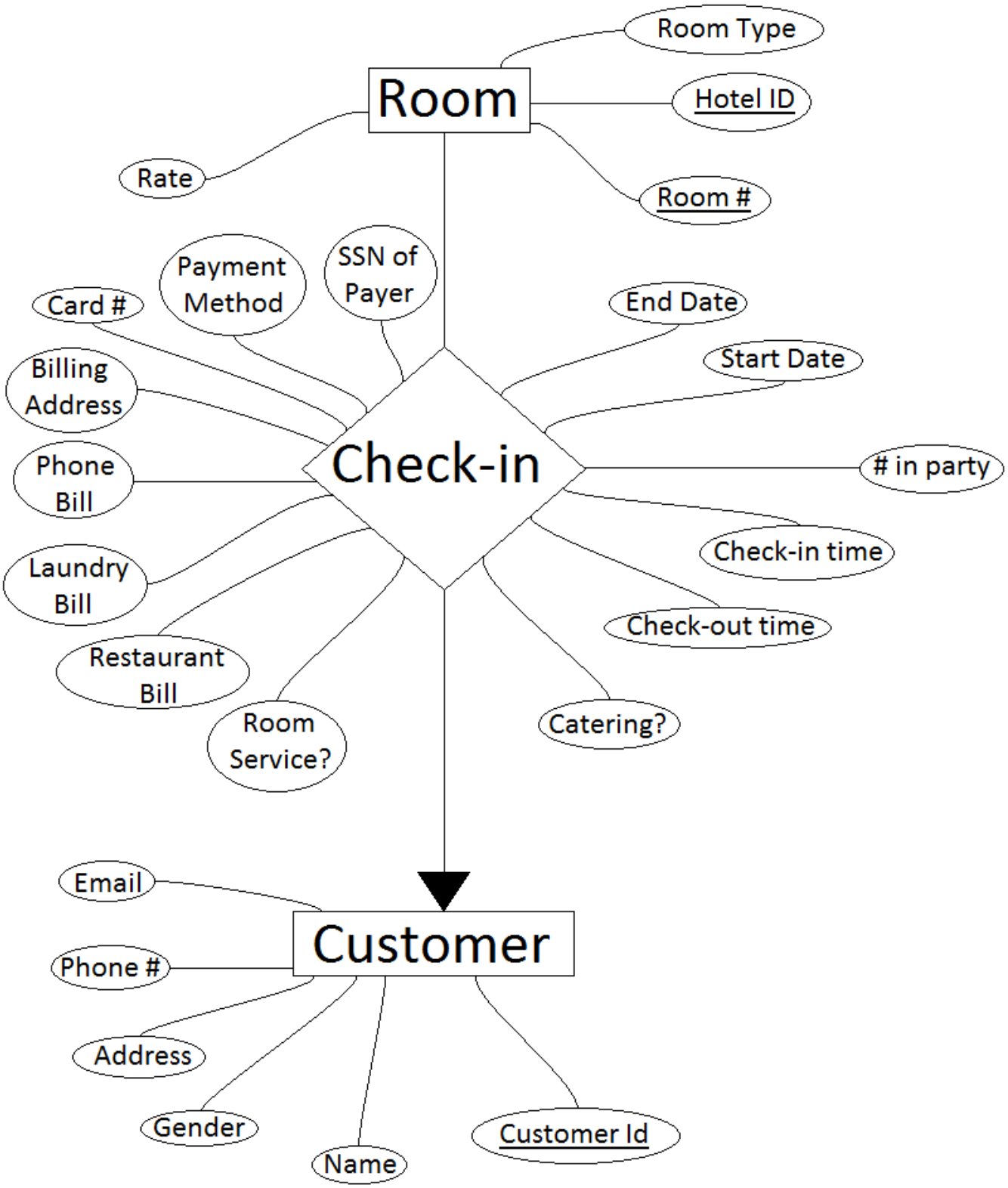
Catering Staff



Front Desk Representative



Customer


Room Type
Room
Hotel ID
Rate
Room #
Payment Method
SSN of Payer
Card #
End Date
Start Date
Billing Address
Check-in
Phone Bill
in party
Check-in time
Laundry Bill
Check-out time
Restaurant Bill
Room Service?
Catering?
Email
Customer
Phone #
Address
Gender
Name
Customer Id

8) Local E/R Documentation

Customer

- Entities:
  - Customer: The customer needs to be able to see all the information about themselves to verify that it is correct.
  - Room: The customer needs to be able to see which hotel room they are staying in, and how much it costs.
- Relationships:
  - Checks in: The customer should be able to see all of the information related to each of their check-in's. A customer is allowed to check into multiple different rooms, because they might have more than 4 people in their party, or for any other reason they want. Each check in will have its own billing information associated with it, which gives the customer the freedom to pay for each room they check into in a different way. If the customer checks into a presidential suite, a room service staff and catering staff will be automatically assigned to them, but they can also opt to have these staff members assigned to them for any of the other room types.

Manager

- Entities:
  - Hotel: The managers needs to see the information about the hotel to which they are assigned. If anything needs to be changed, then they are able to make changes as appropriate.
  - Staff: The managers should be able to see the personal information of all of the staff, and make changes as needed. The staff are separated into subsets because the catering and room service staff need to be able to be assigned to more than one room.
  - Customer: The manager only needs to see the name/gender on the customer to help them know how to address them. The more in depth customer information is left up to the front desk staff to maintain, the manager does not really care about customer information at that level of granularity.
  - Room: The manager needs to see all the information on each room in the hotel, so that they can generate full reports on occupancy and revenue information. Room is a weak entity set of hotel because hotels share the same room numbers, so hotel id is needed to determine a specific hotel room.
- Relationships:
  - Checks into: This gives the manager information on what rooms have customers checked into them, the date ranges those rooms will be occupied, and how many people will be staying in each room.
  - Works at: This just associates the staff with a particular hotel.
  - Assigned to: This lets the manager generate the report about which customers the room service and catering staff are assigned to.

Caterer

- Entities:
  - Catering staff: The catering staff needs to see all his/her own necessary information listed as the attributes in the diagram to ensure the validation of their identities.
  - Customer: The catering staff needs to see his/her customers name and gender to know how to address them.
- Relationships:
  - Assigned-To: This relationship makes the catering staff able to see the list of his/her customer information.

Service staff

- Entities:
  - Room service staff: The room service staff needs to see all his/her own necessary information listed as the attributes in the diagram to ensure the validation of their identities.
  - Customer: The room service staff needs to see his/her customers name and gender to know how to address them.
- Relationships:
  - Assigned-To: This relationship makes the room service staff able to see the list of his/her customer information.

Front Desk

- Entities:
  - Hotel Staff: This entity set is broken down into three different subsets. The front desk subset gives the front desk staff the ability to see their own personal information, without being able to see everyone else's. The catering and room service subsets gives the front desk staff the ability to see what those subsets are assigned to, and make assignments when customers check in.
  - Customer: The front desk staff needs to be able to see all of the personal information for each customer, since they're the ones who have to enter it into the system.
  - Room: The front desk staff needs to be able to see all of the rooms at their hotel so that they can see which ones are empty, and know when rooms will be coming available. The room type lets them know whether or not catering and room service staff MUST be assigned. A non-presidential room can still have staff assigned to it, it just isn't required.
- Relationships:
  - Assigned to: This lets the front desk staff see which customers/rooms the catering and room service staff are assigned to, and make assignments as necessary.

- Check in: This relation contains all of the information related to checking in a customer, and all of the billing information associated with that specific check-in. The number of people that the customer wants to check into a room is associated with check in, and not customer, because a customer could have more than 4 people in their party, in which case they would need multiple hotel rooms. The laundry/phone/restaurant bills are all treated like opening a tab, where each use adds on to the total bill, whereas the catering and room service fields are just Boolean, since the customer can chose to have a catering and/or room service staff assigned to them at check in if they aren't already staying into a presidential suite.

9) Local Relational Schemas

Front Desk Representative staff view
Room(Room #, Hotel ID, Rate, Max Occupancy, Room Type)
Check-in(Room #, Hotel ID, Customer Id, Start Date, End Date, Check-in time, Check-out time, # in party, Room Service?, Catering?, Restaurant Bill, Phone Bill, Laundry Bill, Payment Method, SSN of Payer, Card #, Billing Address)
Customer(Customer Id, Name, Gender, Address, Phone #, Email)
HotelStaff(Staff ID, Hotel ID, SSN, Name, Age, Gender, Phone #, Address)
CAssignedTo(Staff ID, Customer Id, Room #)
RSAssignedTo(Staff ID, Customer Id, Room #)

Room Service Staff View
Customer(Customer Id, Name, Gender)
AssignedTo(Customer Id, Staff ID, Room #)
RoomServiceStaff(Staff ID, Hotel ID, SSN, Name, Age, Gender, Phone #, Address)

Catering Staff View
CateringStaff(Staff ID, Hotel ID, SSN, Name, Age, Gender, Phone #, Address)
Customer(Customer Id, Name, Gender)
AssignedTo(Customer Id, Staff ID, Room #)

Customer View
Room(Hotel ID, Room #, Room Type, Rate)
Customer(Customer Id, Name, Gender, Address, Phone #, Email)
Check-in(Hotel ID, Room #, Customer Id, SSN of Payer, Payment Method, Phone Bill, Laundry Bill, Restaurant Bill, Catering?, Room Service?, Card #, Billing Address, Start Date, End Date, Check-in time, Check-out time, # in party)

Manager View
Room(Room #, Hotel ID, Rate, Room Type, Max Occupancy)
Hotel(Hotel ID, Name, Manager ID, Address, Phone #)
WorksAt(Staff ID, Hotel ID)
HotelStaff(Staff ID, SSN, Name, Age, Gender, Phone #, Address, Job Title)
CAssignedTo(Staff ID, Room #, Customer Id)
RSAssignedTo(Staff ID, Room #, Customer Id)

Customer(Customer Id, Name, Gender)
Check-in(Room #, Hotel ID, Customer Id, Start Date, End Date, # in party)

10) Local Schema Documentation

Front Desk Representative Staff View
The front desk representative can see the room attributes and the attributes for the customers since he/she will need to perform matching between them. The Check-in relationship captures the following information for a particular customer's stay for a particular date range in a particular room: billing information by service category, payment information (method, payer, card number, billing address), check-in time, check-out time, and number of people in the party. The front desk representative needs such info both to assign customers to rooms and to generate bills for customers. The HotelStaff entity set is treated as a superclass with the following subclasses: FrontDeskRep (the front desk representative is the user of the view and should have access to his/her own information), RoomServiceStaff, and CateringStaff. The latter two are allocated by the front desk representative to customers and moreover, each have a special AssignedTo relationship with the customers (unlike the FrontDeskRep) which justifies their information being included. Now in terms of the actual implementation of the HotelStaff "is-a" hierarchy, a single relation called HotelStaff was used with the "Job Title" attribute to distinguish the subclasses (for is-a relationships, this is called the "Use nulls" approach). The reasoning for this choice was that there were few enough nulls and therefore the single relation would save space.

Room Service Staff View
The Customer entity set is represented with the attributes relevant to the room service staff member. The RoomServiceStaff entity set includes all the attributes for the room service staff member since the room service staff member is the user of the view. The "Assigned to" relationship between Customer and RoomServiceStaff shows a room service staff member which customers he/she is assigned to and includes the additional attribute "Room #" so that he/she can locate the rooms that the customers are in.

Catering Staff View
The Customer entity set is represented with the attributes relevant to the catering service staff member. The CateringStaff entity set includes all the attributes for the catering service staff member since the catering service staff member is the user of the view. The "Assigned to" relationship between Customer and CateringStaff shows a catering staff member which customers he/she is assigned to and includes the additional attribute "Room #" so that he/she can locate the rooms that the customers are in.

Customer View
Room and Customer are entity sets that interact with each other. Each room has at most one customer (customer is simply the contact person) associated with it. We refer to the relation schema table that connects the Room entity set to the Customer entity set as Check-in to

represent the relationship. Check-in contains the unique keys of both the Room and the Customer schema, which is Hotel ID, Room #, and Customer Id. While it may be possible to combine the Room and Check-in relations into a single relation (because Check-in is a many-to-one relationship), it was decided to keep them separate because the relation for the Check-in relationship itself has many attributes.

Manager Staff View

Room is a weak entity set that is connected to the entity set Hotel by the supporting relationship "Is in." For that reason, the Room relational schema includes the "Hotel ID" key attribute from Hotel and the "Is in" relationship does not have a relation at all. Hotel Staff has a "Is A" relationship with Manager, Front Desk Rep, Catering Staff, and Room Service Staff. We translated this inheritance relationship to a relational schema by using the "Null Values" approach. This approach was chosen since the "children" staff entity sets have little differences compared to the parent staff entity set. Staff and Hotel are in a many-to-many relationship, and the relationship schema connecting them contains the unique values of both the Staff (Staff ID) and the Hotel (Hotel ID.)