Course: CS 631
Section: 002

Database Management System Design

Group 10
John Losito
Liu Yang

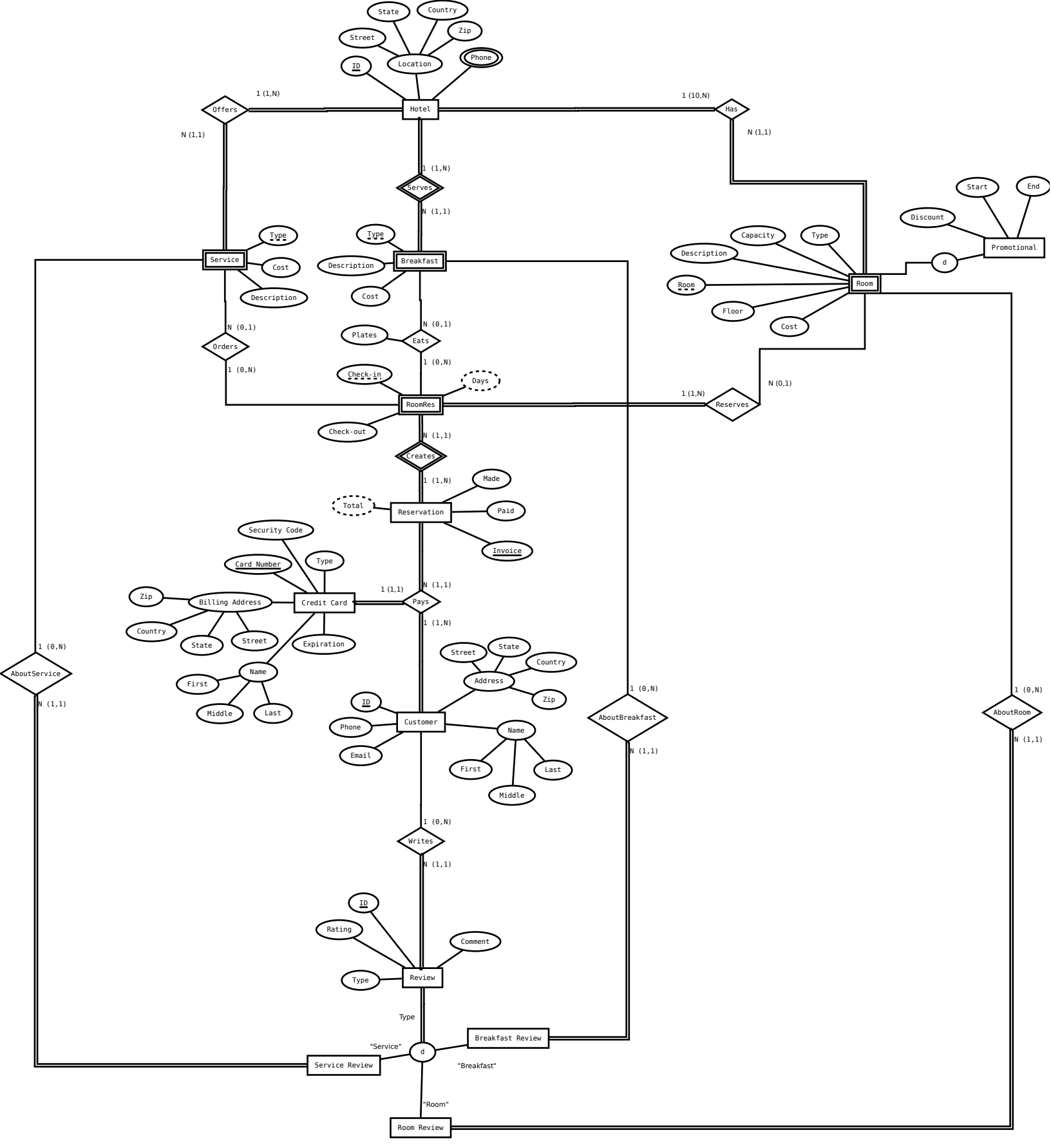
Deliverable 1

1. Outline the goals of this phase.

The purpose of phase one is to derive a conceptual model of a database. The database is meant to represent a hotel chain that has an on-line application that allows customers to make reservations for its rooms that are located around the world. The customer should be able to view information on this application about the rooms, different services that are provided by the hotel, and other various details such as other customer reviews. Thus, in this stage our group will have to analyze key aspects of the specification page that we have been given to determine how to best design and implement the database.

While reading the specification page, words to look out for are nouns which will probably represent entities, characteristics that describe these entities, and verbs that tie different entities together.

One final goal of this phase is to familiarize ourselves with software that is used for entity-relationship modeling. In this case, our group first created a rough draft on paper and then converted it into an acceptable representation using a program called Dia. More importantly It is also meant to get us familiar with designing entity relationship models, its notation, benefits, and flaws.


State
Country
Zip
Street
Location
Phone
ID
Hotel
Offers
1 (1,N)
N (1,1)
1 (10,N)
Has
N (1,1)
1 (1,N)
Serves
N (1,1)
Start
End
Discount
Promotional
Capacity
Type
Description
d
Room
Floor
Cost
Type
Service
Cost
Description
Type
Breakfast
Description
Cost
N (0,1)
Orders
1 (0,N)
N (0,1)
Plates
Eats
1 (0,N)
Check-in
Days
RoomRes
Check-out
1 (1,N)
Reserves
N (0,1)
N (1,1)
Creates
1 (1,N)
Made
Total
Reservation
Paid
Invoice
Security Code
Card Number
Type
Zip
Billing Address
Credit Card
1 (1,1)
N (1,1)
Pays
Country
State
Street
Expiration
1 (1,N)
1 (0,N)
AboutService
N (1,1)
Name
First
Middle
Last
Street
State
Country
Address
Zip
ID
Customer
Phone
Email
Name
First
Middle
Last
1 (0,N)
AboutBreakfast
N (1,1)
1 (0,N)
AboutRoom
N (1,1)
1 (0,N)
Writes
N (1,1)
ID
Rating
Comment
Type
Review
Type
"Service"
d
"Breakfast"
"Room"
Service Review
Breakfast Review
Room Review

3. Mention any assumptions you made in doing the above that go beyond what is given in the project description. In particular, mention all assumptions that led you in determining structural constraints.

Reading the description of how customers make reservations and how each customer can use a credit card that is not in their name led us to believe that one credit could be used by two separate customers. This means that one credit card can pay for many reservations. In order to properly show the constraints of all three being required during a transaction, we decided to combine the customer, reservation, and credit card into a tertiary relation. This enables us to show that all three are required simultaneously for a transaction.

We also assumed from the description that the room, service, and breakfast entities are all weak. This is due to each entity relying on the hotel that hosts each entity. In other words, if there is no hotel, there does not exist a room. If there is no hotel, there does not exist a service. If there is no hotel, there does not exist a breakfast. Taking this approach also allows us to show which room is associated with a specific hotel because the room entity will require the primary key of the parenting entity.

The hotel room also provided us some difficulty. We were unsure of how to represent the difference between a normally priced room and one that has a promotion. Our final conclusion was to represent it with a generalization or specialization hierarchy. This also posed other problems though. A promotion room is essential a room with some extra attributes, but a normally priced room does not have any additional information. The later would then only add redundant information to our diagram. It would also not provide us any useful information when translating the entity relationship model into a relational database.

A similar specialization and generalization hierarchy could have been made for services and breakfast, but again this would not add any additional information to our diagram.

One instance we did find that was different was the customer's review of their reservation. In this case, a specialization and generalization hierarchy did provide us additional information even though there were no additional attributes associated with the specializations. A review can either be a review about a room, service, or breakfast. Thus we made these into their own entity. Doing so allowed us to show that there exists some constraint on each. A room review is only allowed to review a room. A service review is only allowed to review a service. A breakfast review is only allowed to review a breakfast. All of which the customer had to make doing their reservation, meaning that a customer cannot review some other service they did not partake in or order. We found that this approach provided us the most information.

4. Make a list of constraints that apply over and above what you can show in the diagram. In particular make a list of additional keys for entity types (if there are any).

The most difficult aspects of the requirements page to map into an entity-relationship model are derived attributes, referential data, and domain constraints.

As for derived attributes, an example is the total cost of a reservation. The cost of a reservation is derived by all of the subcategory costs. A subcategory cost would be the services and breakfast costs that the customer purchased such as laundry or parking and American pancakes and eggs, so not until all of these subcategories are summed will one get the total cost of a reservation. In concept this may seem trivial, but how can one represent this in an entity-relationship model is not clear. We found that this could not be done in our instance. That led us to the conclusion that these instances are one of the shortcomings of the entity relationship model.

Domain constraints also pose difficultly. Something as simple as valid telephone numbers for a customer cannot be represented in an entity relation model. One can show that a customer has attributes such as an email and telephone number, but one cannot show what are valid inputs for these attributes.

In one's head, one imagine what are valid phone numbers or what an email address looks like, but what happens for more complicated situations? Another simple constraint is the total amount paid for a reservation should never exceed the total cost of the reservation. Those instances of showing domain constraints are critical, but once again, we came to the conclusion that this is another downfall of this type of model.

5. Besides the above, comment on the difficulties you faced in doing this conceptual design task.

Taking the description from the specification page and translating it into a conceptual model for the database is difficult because there are cases where information that is provided is extremely explicit and at other times vague. For example, there is specific instruction in section one of the requirements page to store information about the hotel's location. That location is comprised of a street address, country, state, and zip code. Therefore, one can come to the conclusion that a location is a composite attribute of a hotel, and that location is comprised of several other attributes. Then there is section six of the requirements page that describes information about the customer. One of those descriptions are of the customer's address. All that is stated in the requirements page is that we are required to store the address of the customer, but we could infer that the customer's address is composite attribute similar to the hotel's. What makes this simple problem difficult is that we are unsure if the requirements page is inferring this information since this scenario is already stated or that it does want the customer's location to be represented as an atomic value. The situation of vagueness is also difficult because there is no way to determine how this will impact the system in the future.

Getting information out of order is frustrating. Information may come in such a way that may not make sense at the moment. Only until more information is provided will that previous information make sense. What makes this situation even more frustrating is that there is no guarantee that we can realize that there exists a relation between two different pieces of information. For instance, there is a description in the requirements page of how customers make reservations and what information is needed for each reservation. Then there is a description of how customers use credit cards and how each credit card does not necessarily need to be in the name of the customer using it. What may not be clear at the moment is how the three entities relate to each other. The first being the customer; the second being the reservation; the third being the credit card. Not until all the information is provided may the relation become clear. We found ourselves having to take an iterative approach where we made an original draft and then refined it until it made more logical sense.

Figuring out unimportant information is difficult. What the hotel may think is important may not be important in an entity-relationship model. Therefore, it difficult for us to determine something such as domain constraints or how derived attributes are created and if that information has any bearing on our model. For instance, there is a total cost that is associated with each reservation. The total cost is composed of sub total costs such as services and breakfast. It is unclear if this relation has any bearing on our entity-relationship model and how to represent it. There is also the case of the lack of generalization and specialization hierarchies. A hotel representative may think that think this information is critical to our diagram. They may want to show how a breakfast can be of five different types. What we found is that even if we did add this additional feature into our diagram, it did not provide any useful information.

All of these difficulties that we faced such as vagueness, unimportant information, and conflicting data all reduce to one. It comes down to how we interpret the requirements page opposed to how the hotel interprets it. There are many way one can conclude how the entity relationship model should look like. All could be correct. That is what makes this project most difficult. We found that no one right answer makes discussing situations difficult because the way one person may interpret the description may be different than the way another person interprets the same information.