This is an exercise from the Merode course at KU Leuven.
The solution is given as an EDG (part of a Merode model)
Some of the mistakes from student solutions are specific to Merode, others are universal

Feedback Home for the Elderly

Considerations to arrive at template solution (EDG)

Older Student solutions

Home for the Elderly

- A Home for the elderly is divided in departments. Each department consists of a number of rooms and each room has a number of beds. Each inhabitant is assigned a bed for his/her stay. A bed belongs to a particular category. Each category has a price. The home offers different types of stays (=category): rest home (RH), rest & nursing home (RNH), service flat (SF). An inhabitant can do more than one stay at a time: if an inhabitant of a service flat has to stay in rest&nursing for a while, the service flat stay goes on.

Candidate Object Types

- Home
  - There is only one home, so this is the context. If you are managing several homes, then this is a valid Object Type
- Department
  - OK
- Room
  - OK

Candidate Object Types

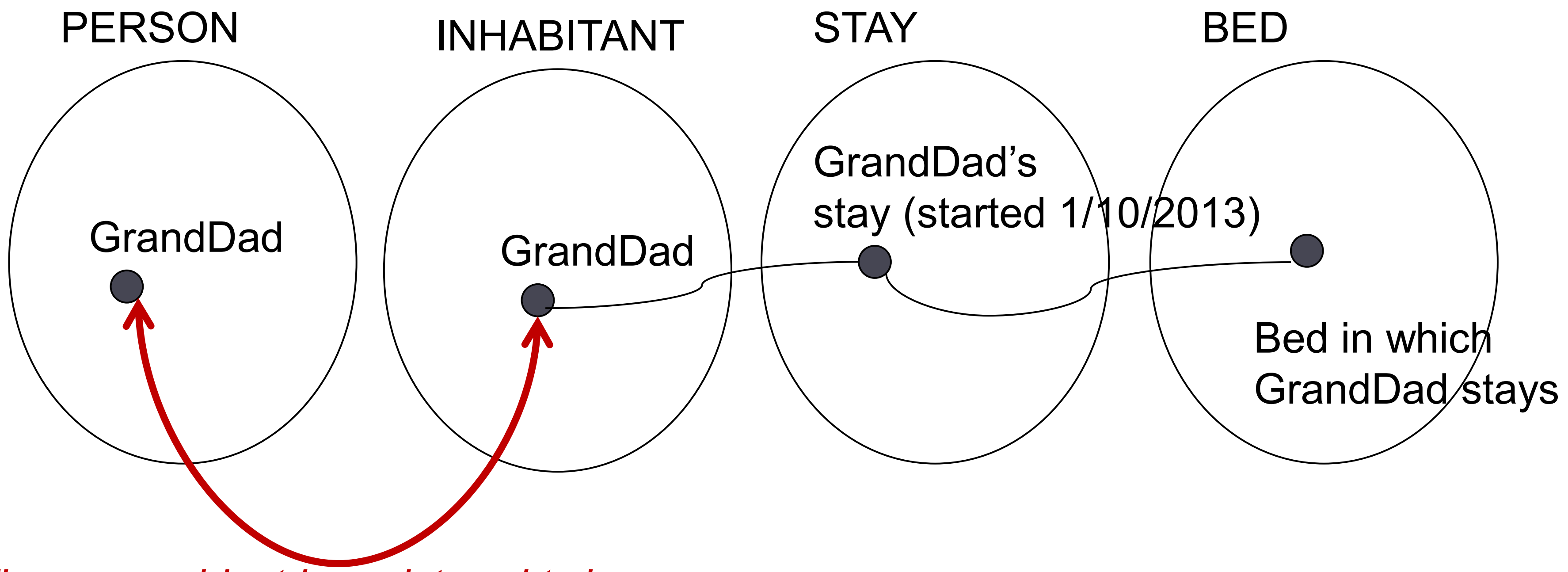
- Bed
  - A bed can be a physical bed, made from wood or steel. But it can also be interpreted as the abstract concept of a “place” in a room.
  - These two different interpretations are important when considering the relationship with Room. A physical bed exists independently from a room and can be moved from one room to another.
  - A “place” on the contrary, can be considered as thight to a room: another room is another place. If one wants to model resource allocation, then a place will make use of a bed to lay people at rest.

Candidate Object Types (ctd.)

- Inhabitant ➔ Person
  - An inhabitant or resident is a person that resides in the home. One needs to carefully investigate when a person enters and leaves the universe of the rest home. Is a person known once the person has been assigned a place in a room or is a person known and registered as soon as this person leaves his/her personal details to be contacted as soon as there is a place available. In other words: is managing the people on e.g. a waiting list in or out scope ?
- Possible Extension ➔ Waiting List
  - Since we are speaking of a waiting list, is waiting list then a candidate object type? No, because the waiting list is a subset of the class “Person”, namely those that are in the state “waiting”, whereas the “Inhabitants” are the people in the class “Person” that are in the state “place assigned”

Person – Inhabitant – registration - assessment

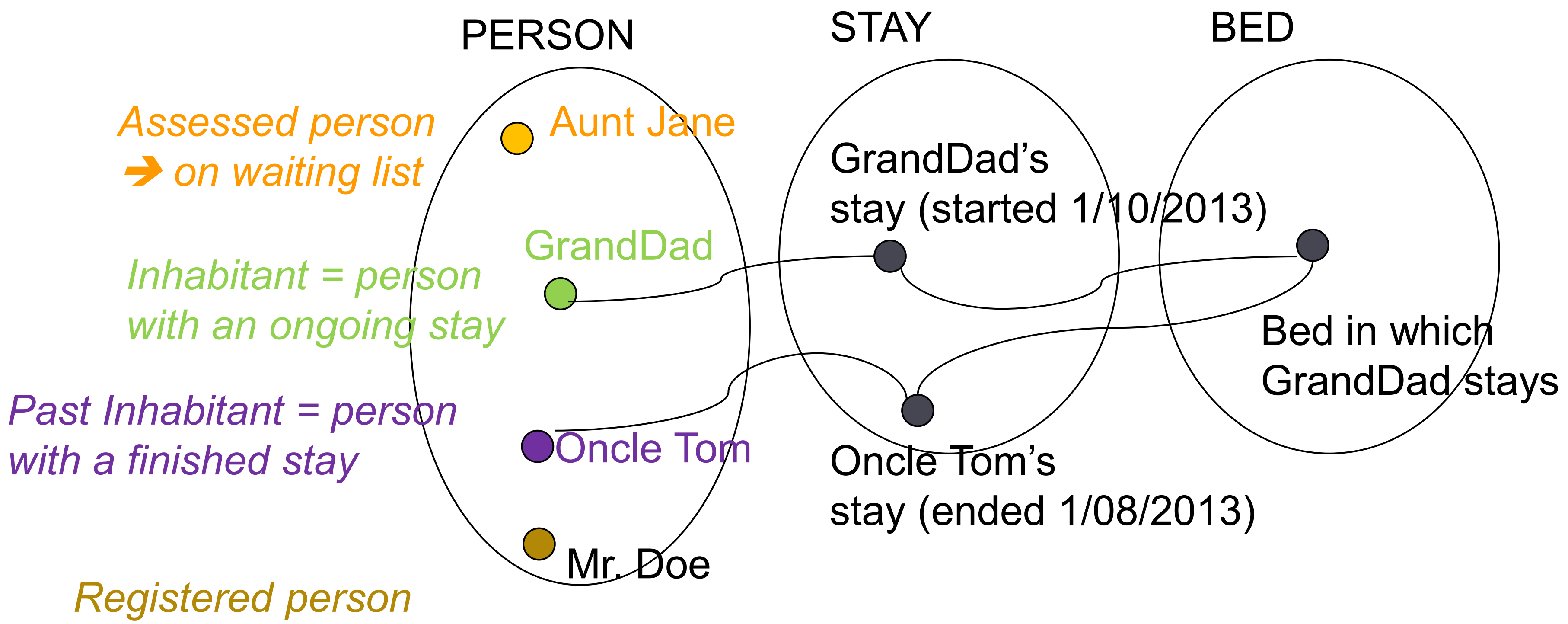
- Avoid classes that have the same objects
- Avoid objects to move from one class to another


PERSON
INHABITANT
STAY
BED
GrandDad
GrandDad
GrandDad's stay (started 1/10/2013)
Bed in which GrandDad stays

The same object is registered twice
Or, the object moves from one class to the next

Person – Inhabitant – registration - assessment

- Avoid classes that have the same objects
- Avoid objects to move from one class to another


PERSON
STAY
BED
Assessed person
➔ on waiting list
Aunt Jane
GrandDad
Inhabitant = person with an ongoing stay
Past Inhabitant = person with a finished stay
Oncle Tom
Mr. Doe
Registered person
GrandDad's stay (started 1/10/2013)
Oncle Tom's stay (ended 1/08/2013)
Bed in which GrandDad stays

Candidate Object Types (ctd.)

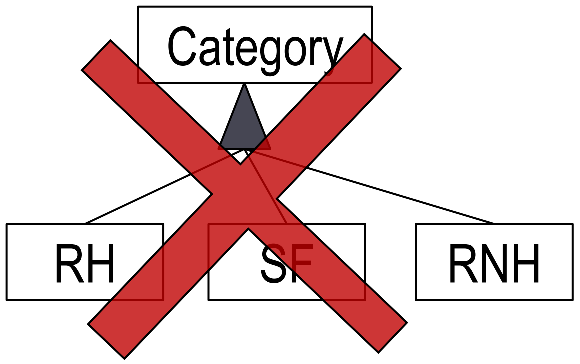
- Stay
  - Stay will be required to model the time period that a person has something in common with a bed/place.

- Category, types of stay (=category)
  - Required to capture the different kinds of services that are offered to residents.
  - A category comes with a price, so price can be modelled as attribute of category.

Candidate Object Types (ctd.)

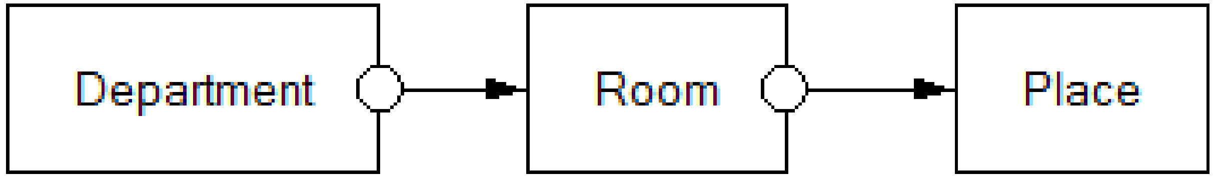
- rest home (RH), rest & nursing home (RNH), service flat (SF)
  - These three candidate object types can be considered as **instances** of the class “category”
  - An alternative would be not to model category, but to identify three subtypes of stays.
  - ➔ see chapter 8 and the type pattern.

- Never mix class and instance:



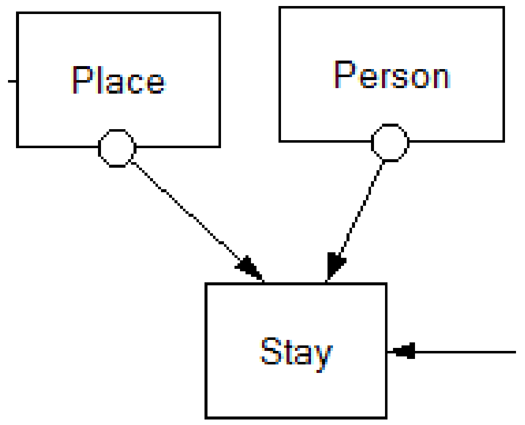
Relationships

- Department – Room
  - A Room is modelled as ED from Department. This means that a room can not be reassigned to another department. This choice simplifies the model, but then we need to accept the implied constraints. To allow for more flexibility, one can model a modifiable relationship, which will require an intermediate “assignment” Object Type.
  - A department can exist (temporarily) without having a room and it can have many rooms. Each Room belongs to exactly one and always the same department.
- Room – Place
  - A Place is ED from a Room: another room is another place.
  - A Room can exist (temporarily) without having a place and it can have many Places. Each Place belongs to exactly one and always the same Room.



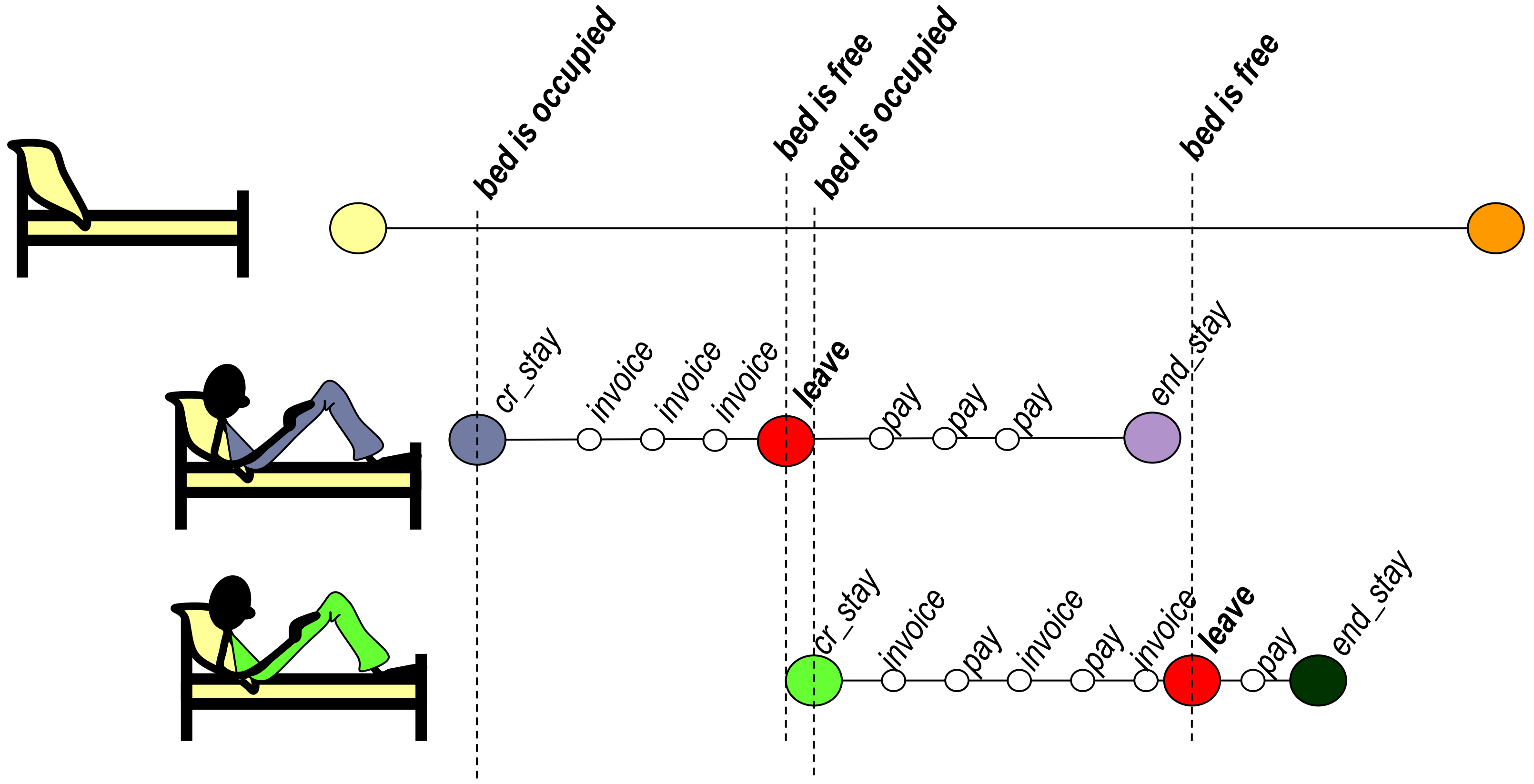
Relationships

- Place – Stay - Person
  - A stay exists in the context of exactly one and always the same place: a change of place means a new stay is started. So Stay is ED from Place
  - A place can be temporarily empty, so a place needs not to have a stay. How many stays can a place have at one point in time ? As we did not consider behaviour, one would logically think one.
  - When considering behaviour (see extended assignment & next slide), we arrive at the conclusion that a place can have several ongoing stays.
  - A stay is done by exactly one and always the same person: another person means another stay. So stay is ED from Person
  - Since the service flat stay goes on while a person resides in rest&nursing, one person can have more than one simultaneous stays. Let us assume a person can be on the waiting list, so then it is not mandatory for a person to have at least one stay.



When does a stay end ?

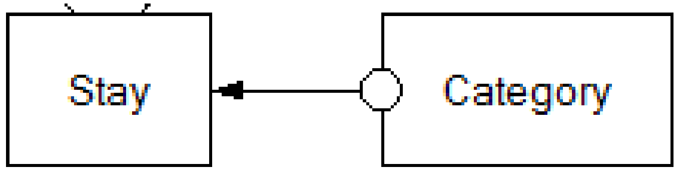
- Consider the life cycles of Place/Bed versus Stay to determine the correct cardinalities.



- ➔ One place has optional many ongoing stays.

Relationships

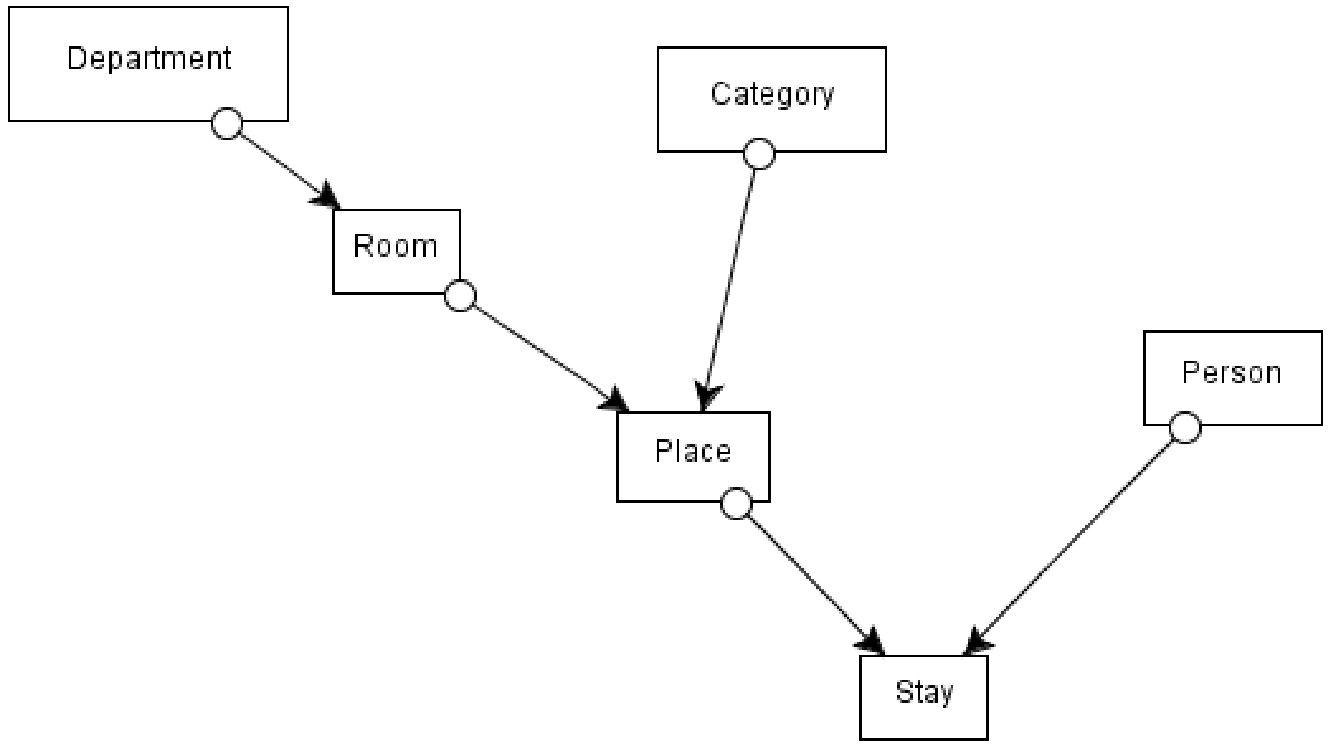
- Relate Category to Department, Room, Place or Category to Stay ?
  - A category could be related to a department, a room, a bed/place or to a stay. Relating the category to the stay allows for more flexibility, as a place can then be used for different types of stays. If you relate category to place, then a place will always serve for the same type of stay.
  - If a category is related to a room, then all places in that room belong to the same category. Analogous for department.
  - If a stay is ED from category, then each stay has exactly one and always the same category : changing the category means that a new stay is started.



  - If a place is ED from category, then each place has exactly one and always the same category : changing the category means that a new place is created.

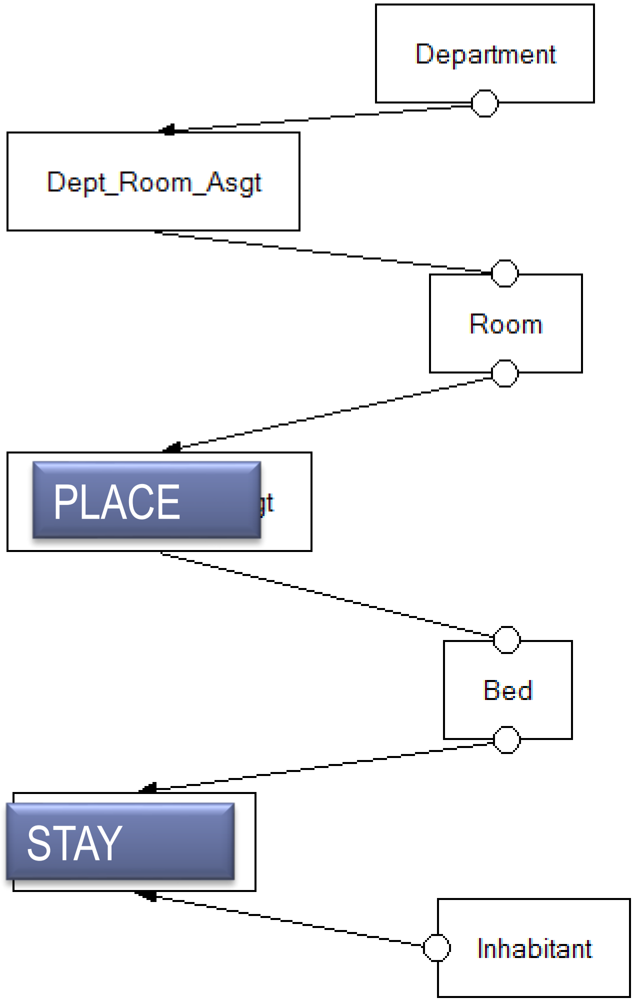
Basic Model Solution

- Taking into account that the text says that “A bed belongs to a particular category” relating category to place would be more in line with to the case description than relating category to stay.



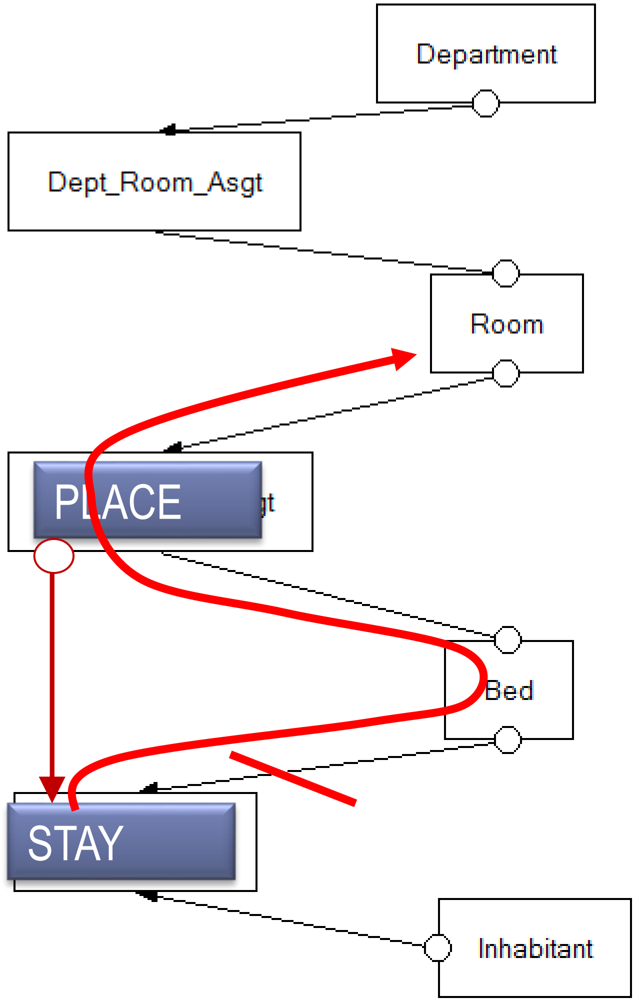
Additional Considerations

- Naming of assignment object types
  - Assigning a bed to a room corresponds with the notion of “Place”
  - Assigning an inhabitant to a bed, corresponds to the notion of “Stay”.



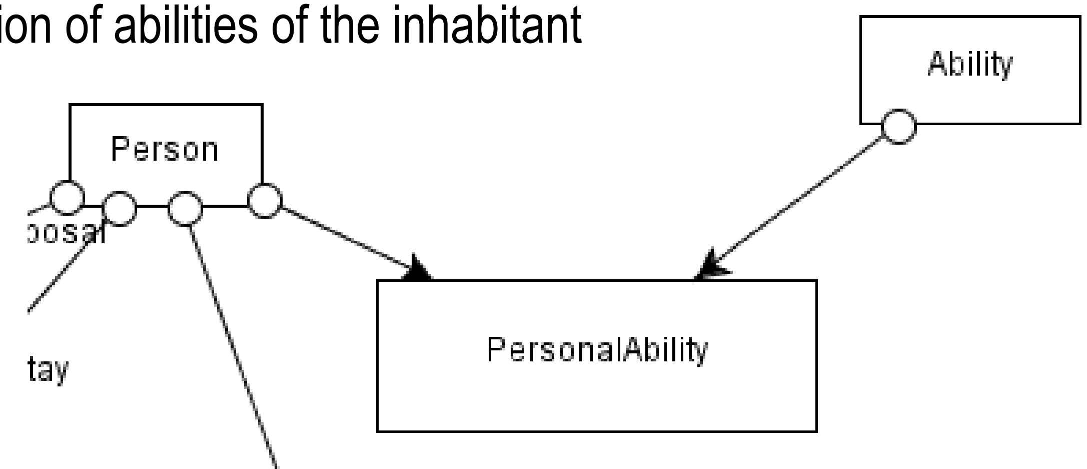
Additional Considerations

- Reconsider connections:
  - now a Stay is connected to a bed, which is in turn connected to a room via place. What happens when a bed is moved to another room ? This destroys the old and creates a new “place” object.
  - You could consider to connecting the stay to “place” instead of to bed. By making stay ED of Place, you obtain that a bed cannot be moved to another room while there is a stay going on
- Check out the domain modelling patterns, in particular “Nested association pattern” *(to see in Chapter 8)*



Assessment

- Could be considered as
  - filling in a number of attributes of “PERSON” ➔ it’s a task requiring an input service
  - Assessing a person along a set of predefined abilities
    - Requires defining “Ability” as a class
    - Assessment is a task during which “Personal Abilities” are registered (via Input Service)
    - This more extended model allows to re-assess a person and keep a trace of the evolution of abilities of the inhabitant



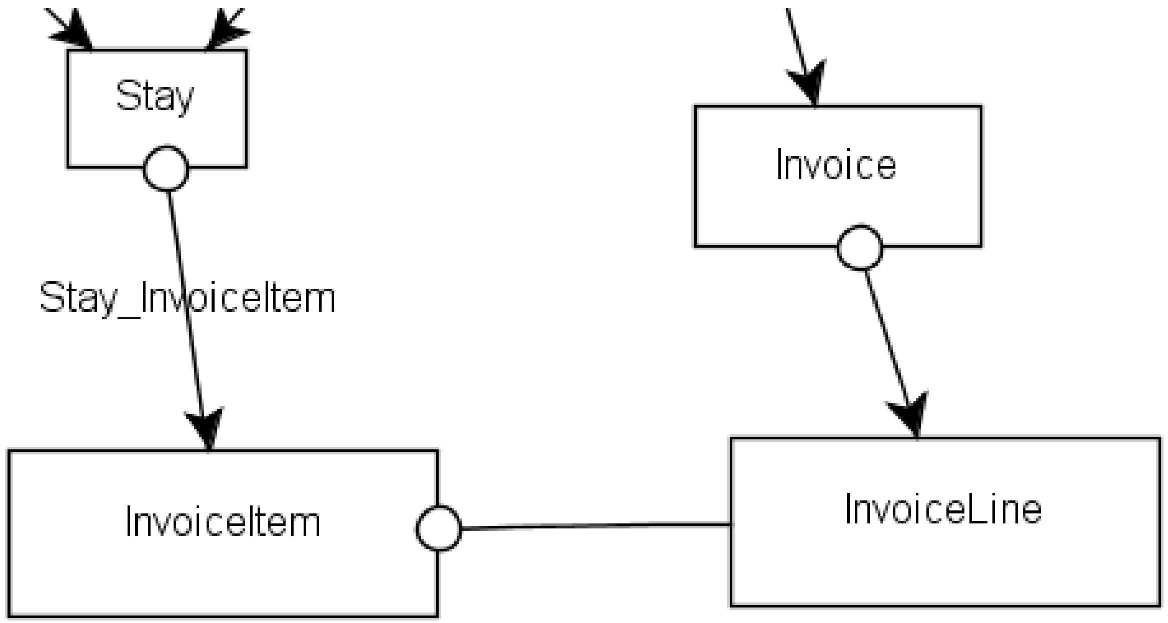
Proposal

- Could be a paper based process ➔ no corresponding item in the model, except maybe “rank” of a person on the waiting list.
- Could be a registered object that is kept for management purposes:
    - When was it made
    - What is its state (accepted or not)
    - ….
  - ➔ can be a separate object of stay because not all proposals result in a stay.

Invoice (=how to make money ☺)

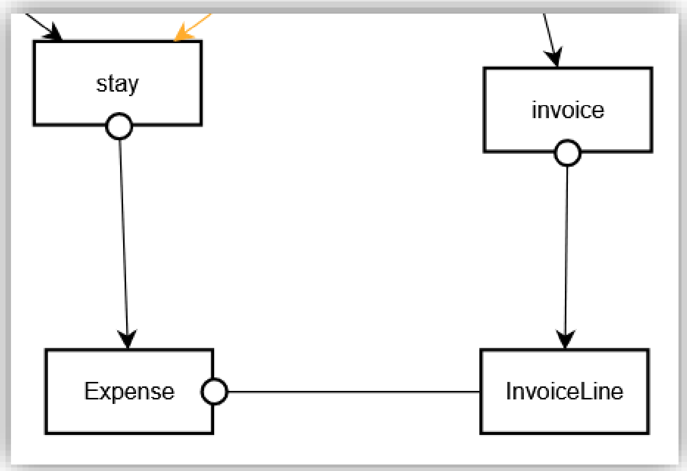
- Make sure you know who the invoice is sent to (relate to person)
- Make sure you know what is invoiced (relate to stay)
- Make sure you know who did an expense (relate to stay or at least to person)
- Make sure you can send multiple invoices: you must be able to send a next invoice, even if the previous one has not been paid yet.
- Consider different levels of granularity
  - invoice items: e.g. a doctor's visit, an extra meal for a visitor, the price for participating to organised activities, …
  - Invoice= group of items invoiced at the end of the month
  - Since invoice items can exist before being put on an invoice, add intermediate association class to connect invoice item to invoice.

Invoicing

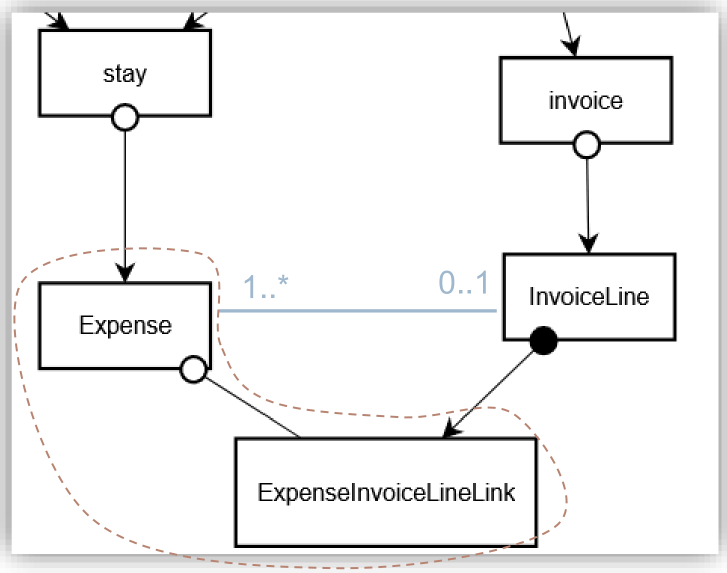


Invoicing - valid alternative

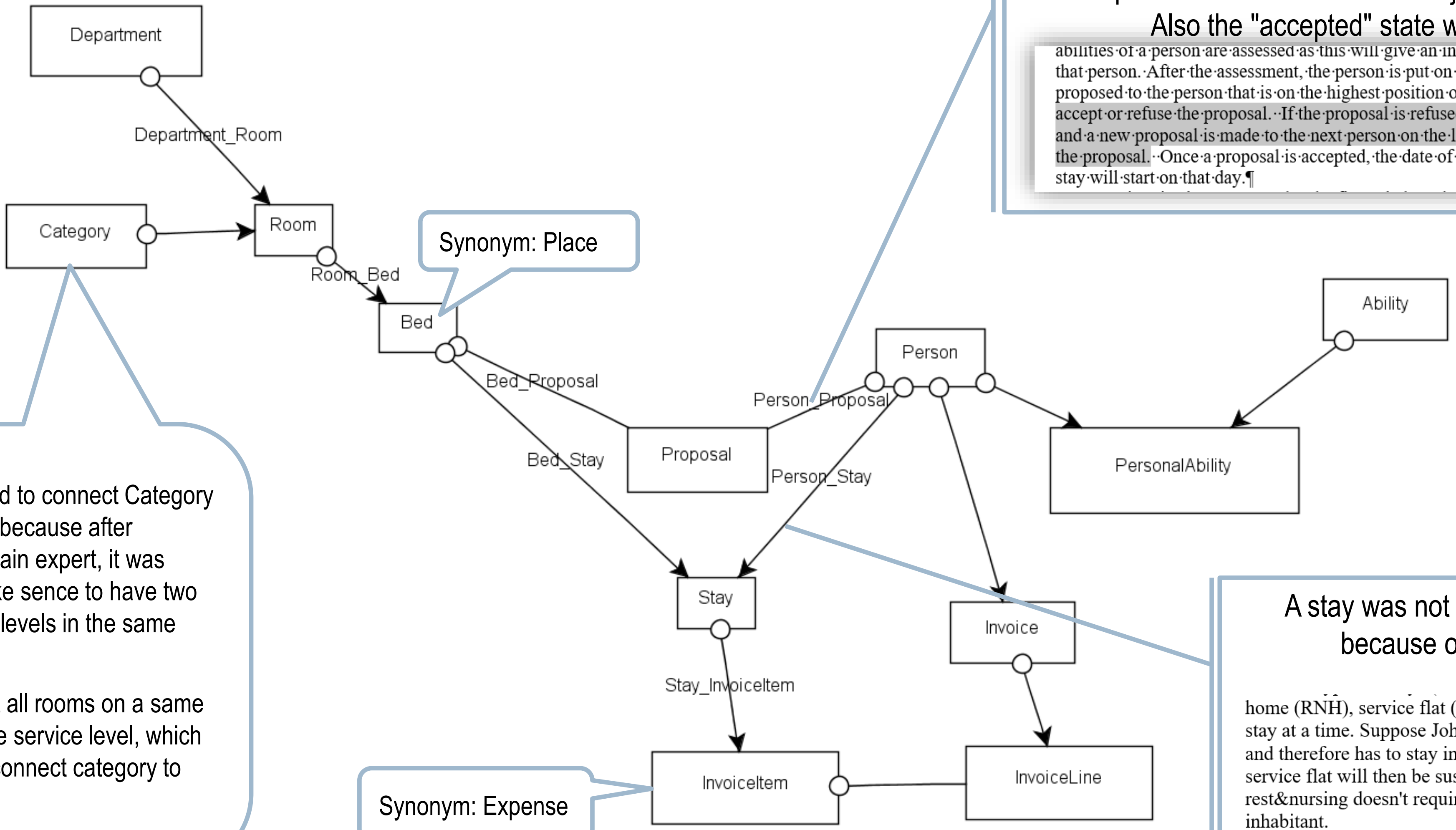
- With this model, each expense ends up as a separate line on the invoice. E.g. if Grandpa drinks a coffee every sunday sunday in the cafetaria, there will be 4 lines on the invoice by the end of the month.



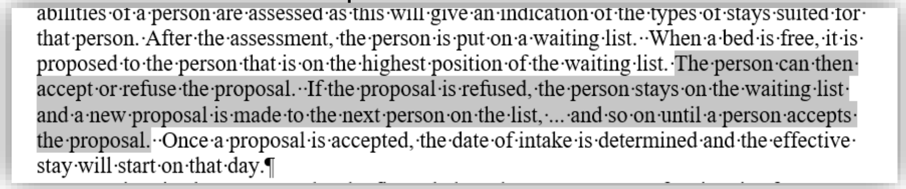
- If you want to be able to group these expenses in one line, then an additional intermediate object is needed. The blue association shows the UML equivalent: initially, and expense has no corresponding invoice line, and later it has at most one. The invoice line will refer to at least one expense, but may refer to many (alle the coffees in one line). Because of the 0..1, there is no ED, and so an intermediate class is needed. As will be seen in chapter 10, such additional class can be merged into the Expense class through a model-to-model transformation when generating the database schema.



Complex solution



A person receives max. one proposal, see this part of the description. We assume that the "rejected" state is a final state. Also the "accepted" state will be a final state.


abilities of a person are assessed as this will give an indication of the types of stays suited for that person. After the assessment, the person is put on a waiting list. When a bed is free, it is proposed to the person that is on the highest position of the waiting list. The person can then accept or refuse the proposal. If the proposal is refused, the person stays on the waiting list and a new proposal is made to the next person on the list, … and so on until a person accepts the proposal. Once a proposal is accepted, the date of intake is determined and the effective stay will start on that day.¶

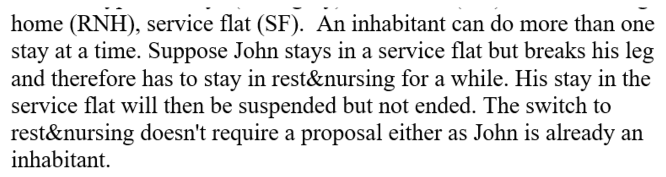
Synonym: Place

In this solution, it was opted to connect Category to Room instead of Place, because after dicussion with the the domain expert, it was decided that it doesn't make sence to have two beds with different service levels in the same room.

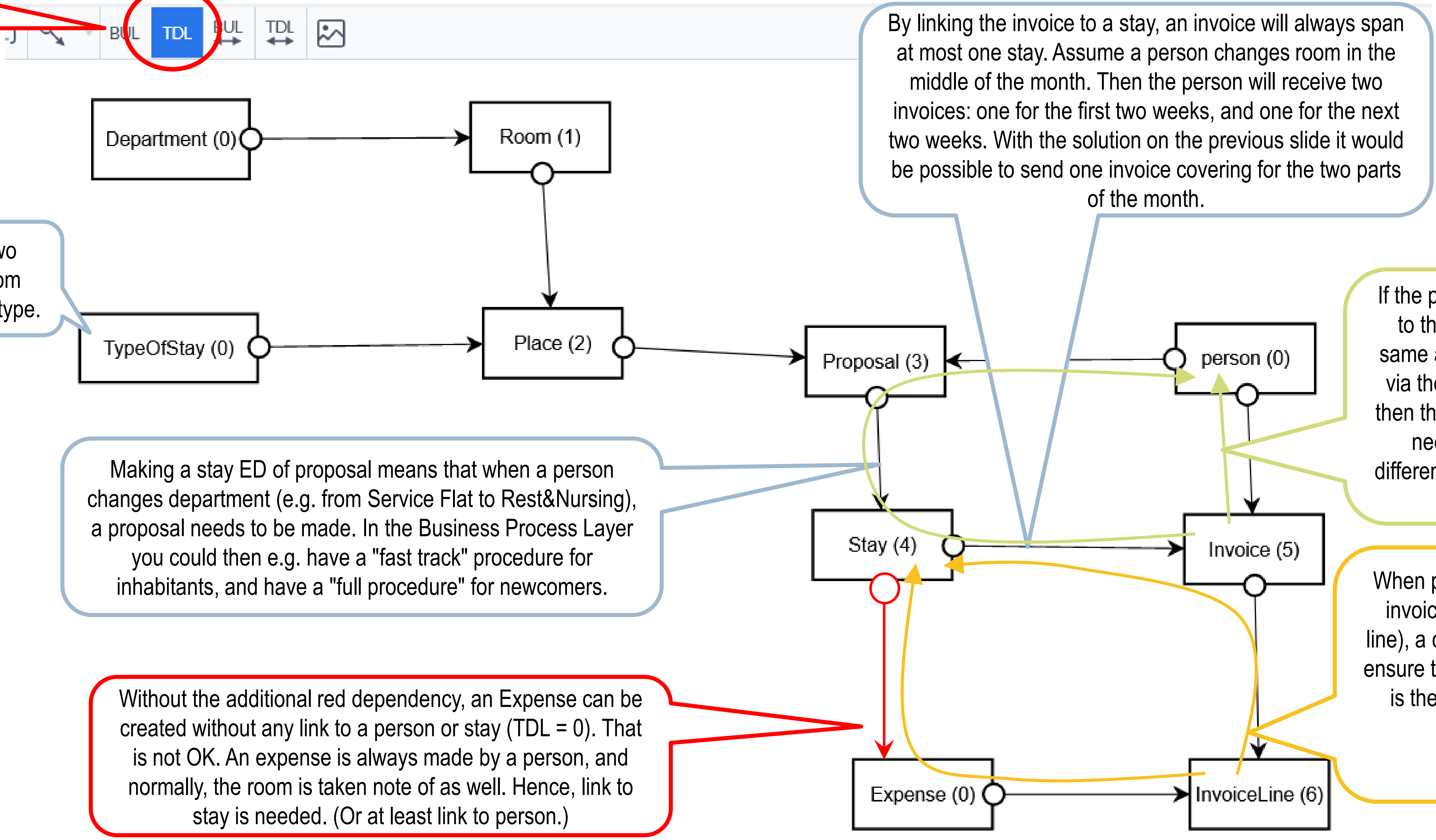
One could even argue that all rooms on a same level should have the same service level, which would be an arugment to connect category to Department

Synonym: Expense

A stay was not made ED of the proposal because of this part of the text


home (RNH), service flat (SF). An inhabitant can do more than one stay at a time. Suppose John stays in a service flat but breaks his leg and therefore has to stay in rest&nursing for a while. His stay in the service flat will then be suspended but not ended. The switch to rest&nursing doesn't require a proposal either as John is already an inhabitant.

Valid Alternatives



Click to visualise order in which objects will be created

With this solution, two places in a same room could be of a different type.

By linking the invoice to a stay, an invoice will always span at most one stay. Assume a person changes room in the middle of the month. Then the person will receive two invoices: one for the first two weeks, and one for the next two weeks. With the solution on the previous slide it would be possible to send one invoice covering for the two parts of the month.

If the person directly connected to the invoice is always the same as the person connected via the stay and the proposal, then the direct connection is not needed. If these can be different, then this association is needed.

Making a stay ED of proposal means that when a person changes department (e.g. from Service Flat to Rest&Nursing), a proposal needs to be made. In the Business Process Layer you could then e.g. have a "fast track" procedure for inhabitants, and have a "full procedure" for newcomers.

When putting and expense on an invoice (by creating an inboice line), a constraint will be needed to ensure that the stay of the expense is the same as the stay of the invoice.

(see Chapter 7)

Without the additional red dependency, an Expense can be created without any link to a person or stay (TDL = 0). That is not OK. An expense is always made by a person, and normally, the room is taken note of as well. Hence, link to stay is needed. (Or at least link to person.)

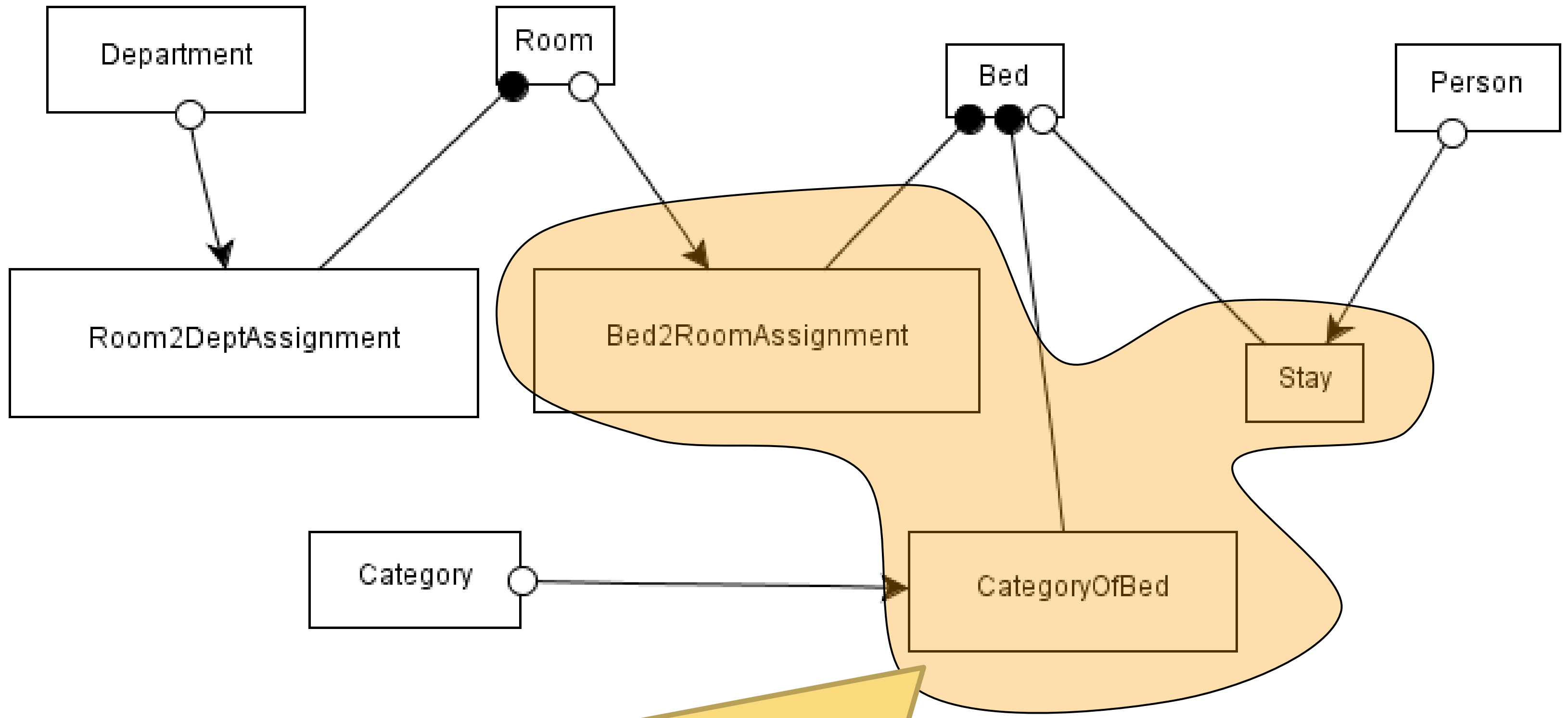
Student solutions with feedback

Look at the following proposed solutions.

Find out what you think is wrong and why.

Look at the feedback.

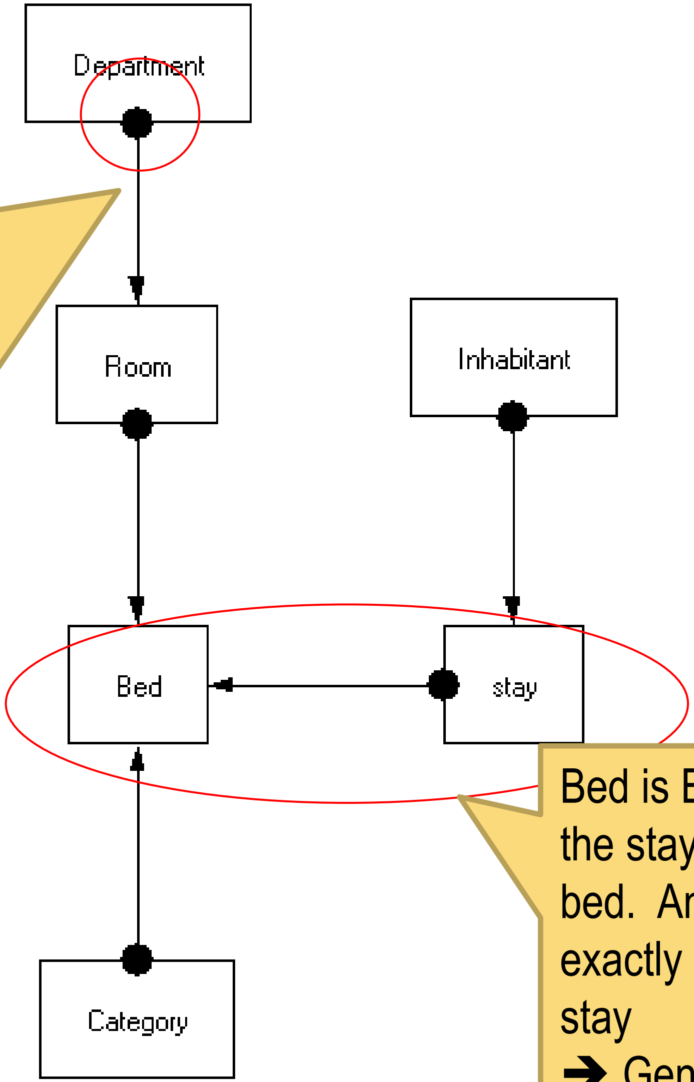
Home for the Elderly



Do these three objects evolve completely independently ? Check out the domain modelling patterns, in particular “Nested association pattern” (seen in Chapter 8).

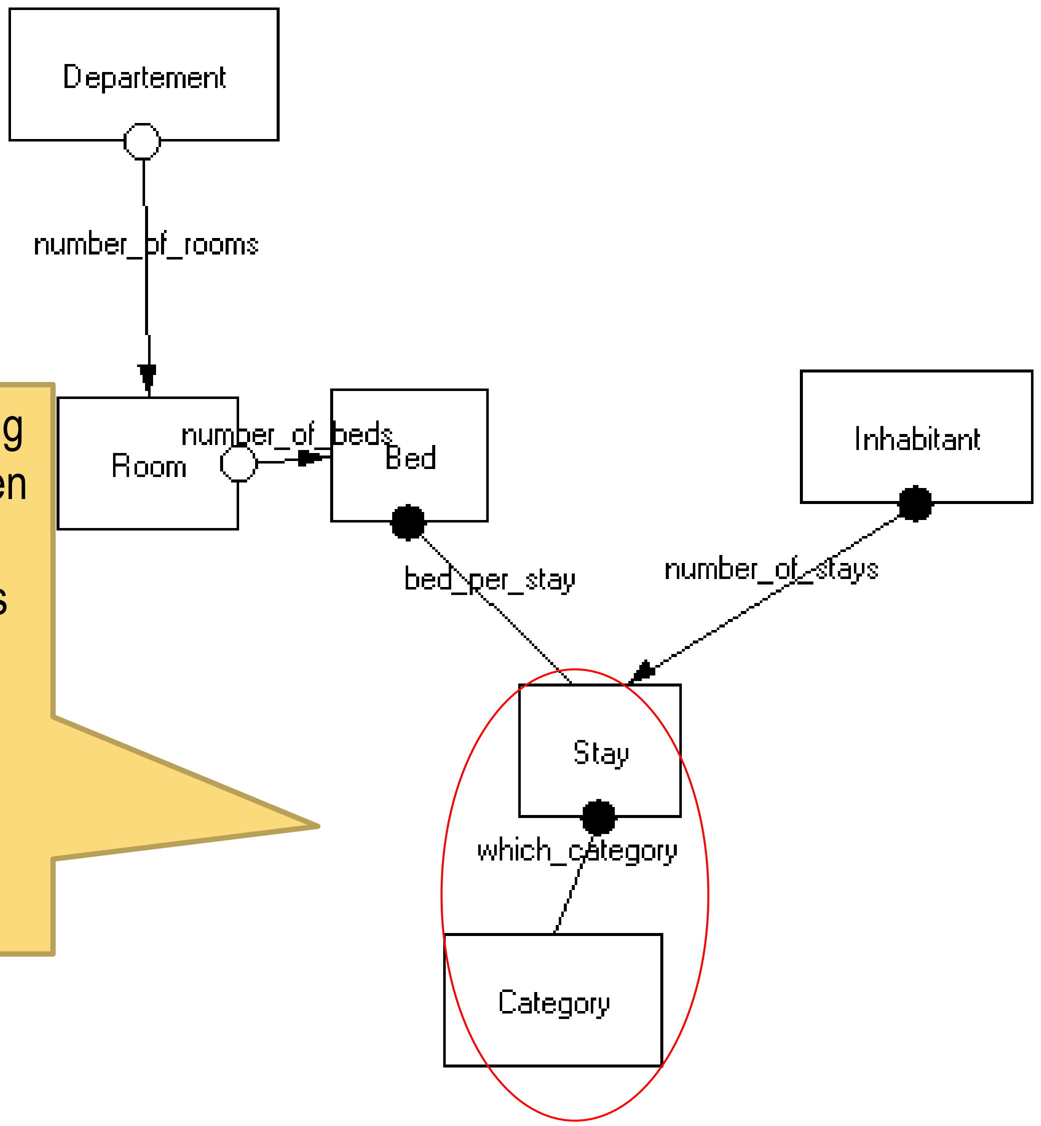
Home for the Elderly

In this schema, all associations are mandatory. This implies a lot of constraints for the users of the future information systems based on this model:
You cannot create a Department, without creating at least one room in that department.
You cannot create a room without creating at least one bed.
As Bed is ED on stay, you cannot create a bed, without creating a stay, which in turns requires an inhabitant.



Bed is ED on stay, meaning that the stay exists first, and then the bed. And the bed refers to exactly one and always the same stay
➔ Generate and read the learning report !

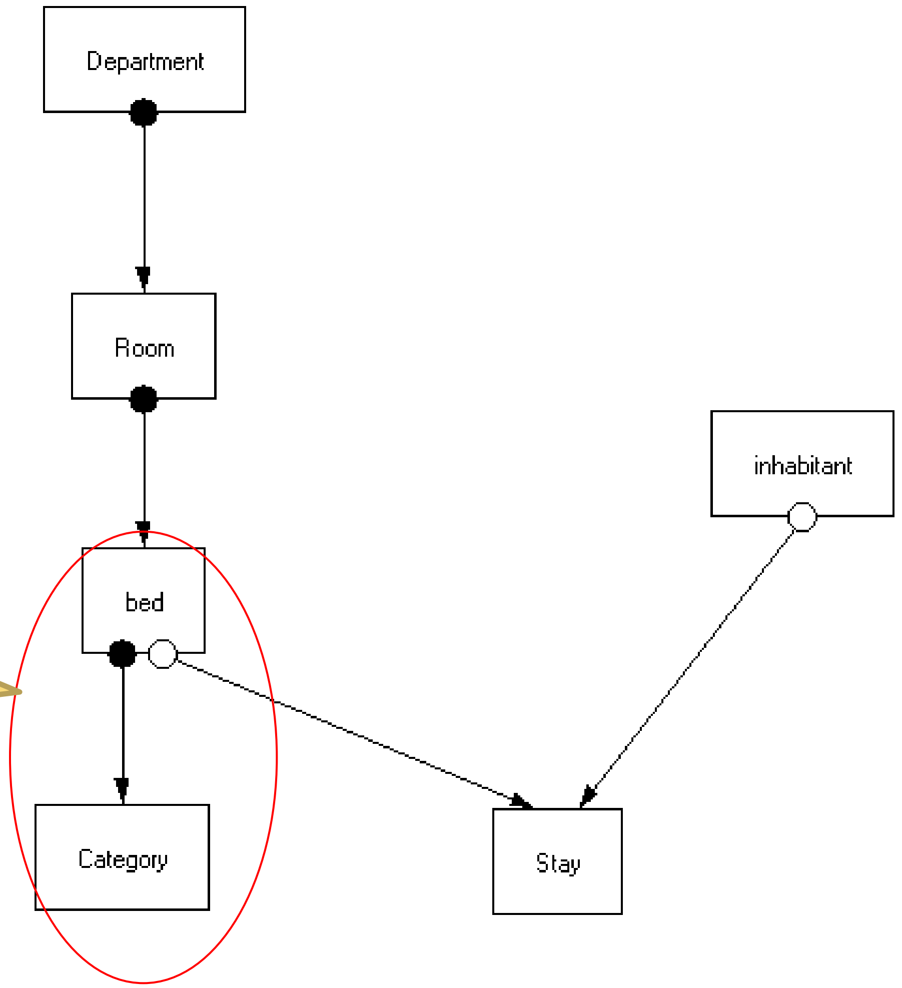
Home for the Elderly



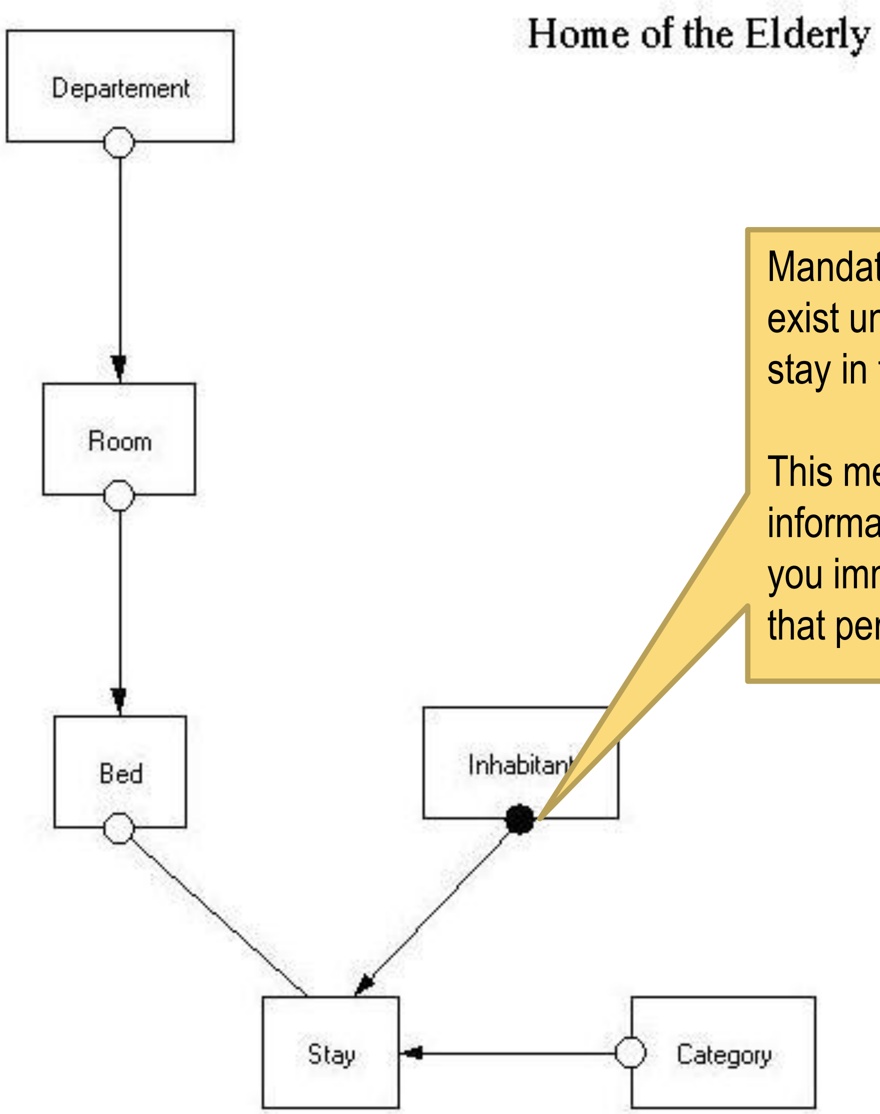
Category is ED on Stay, meaning that the Stay exists first, and then the category. And the category refers to exactly one and always the same stay.
➔ switch to UML notation and read the learning report !

Home for the Elderly

Category is ED on Bed, meaning that the bed exists first, and then the category. And the category refers to exactly one and always the same bed. Moreover, a bed can have simultaneously multiple categories.
➔ switch to UML notation and read the learning report !



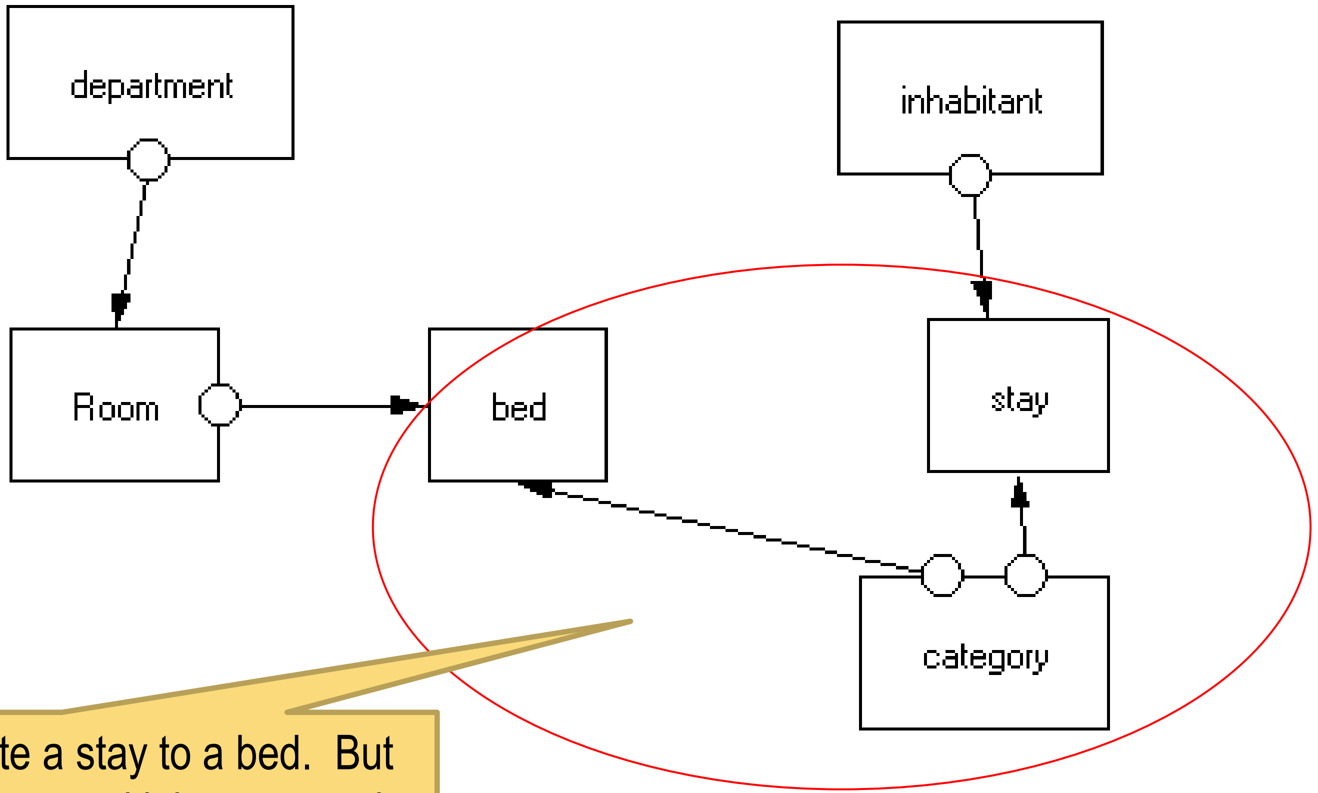
Home for the Elderly



Mandatory ➔ An inhabitant cannot exist unless (s)he has an ongoing stay in the Home.

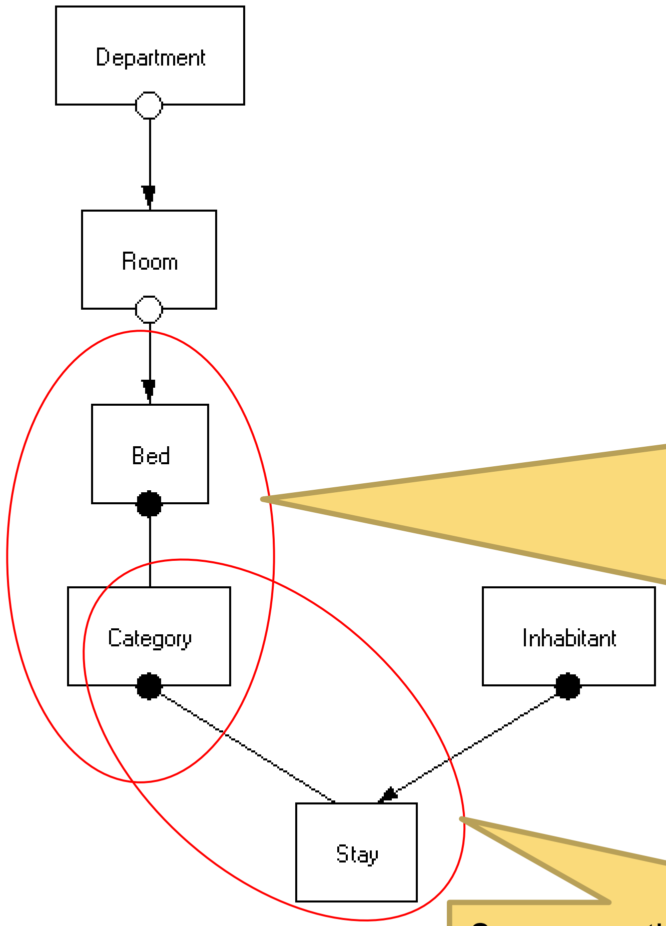
This means you can never register information about a person, unless you immediately register a stay for that person as well.

Home for the Elderly



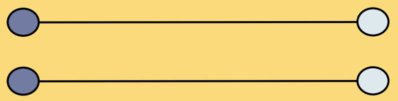
A category is used to relate a stay to a bed. But since one category can have multiple stays and multiple beds, we cannot know which stay relates to which bed.

Home for the Elderly

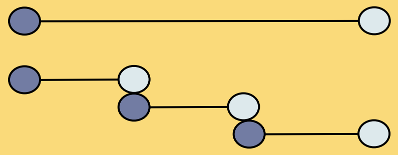


Category is ED on Bed, meaning that the bed exists first, and then the category. And the category refers to exactly one and always the same bed. ➔ switch to UML notation and read the learning report !
What is the difference between the life cycle of a bed and of a category ?

Like this ? Bed
Category



or like this ? Bed
Category



Neither of the solutions seems right.

Same question for the relationship between a category and a stay. There can be only one stay per category.
How do the life cycles of category and stay relate to each other ?