



Database System Concepts for Non-Computer Scientist – WiSe 22/23

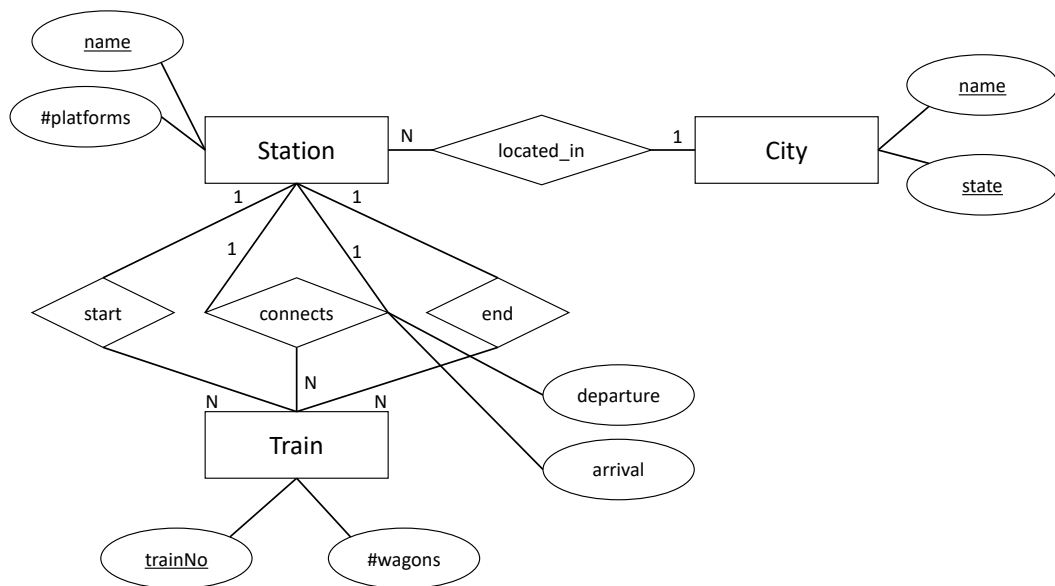
Alice Rey (rey@in.tum.de)

<http://db.in.tum.de/teaching/ws2223/DBSandere/?lang=en>

Sheet 05

Exercise 1

Consider the entity relationship diagram from exercise sheet 3:



Refine the relational schema that you created in sheet 3 from the ER-Diagram. Underline keys and find appropriate data types. As a reminder, here is the un-refined schema:

- City : {[name : string, state : string]} (1)
- Station : {[name : string, #platforms : integer]} (2)
- Train : {[trainNo : integer, #wagons : integer]} (3)

For the relationships in the model, we create the following relations:

- located_in : {[stationName : string, cityName : string, cityState : string]} (4)
- start : {[trainNo : integer, stationName : string]} (5)
- end : {[trainNo : integer, stationName : string]} (6)
- connects : {[fromStationName : string, toStationName : string, trainNo : integer, departure : date, arrival : date]} (7)

Solution:

During refinement, we merge relations for binary relationships into relations for entities, if the relations have the same key and it was a 1:N, N:1 or 1:1 relationship in the ER-model. Note: A binary 1:N relationship can be merged into the entity with the N next to it.

Doing so we can merge the (4) relation into (2). (5) gets merged into (3). And same for the *end* relation, which also gets merged into *train*.

$$(4) \mapsto (2), (5) \mapsto (3), (6) \mapsto (3)$$

Thus, we end up with the following schema:

```

City : {[name : string, state : string]}
Station : {[name : string, #platforms : integer,
            cityName : string, state : string]}
Train : {[trainNo : integer, #wagons : integer,
          startStationName : string, endStationName : string]}
connects : {[fromStationName : string, toStationName : string,
            trainNo : integer, departure : date, arrival : date]}

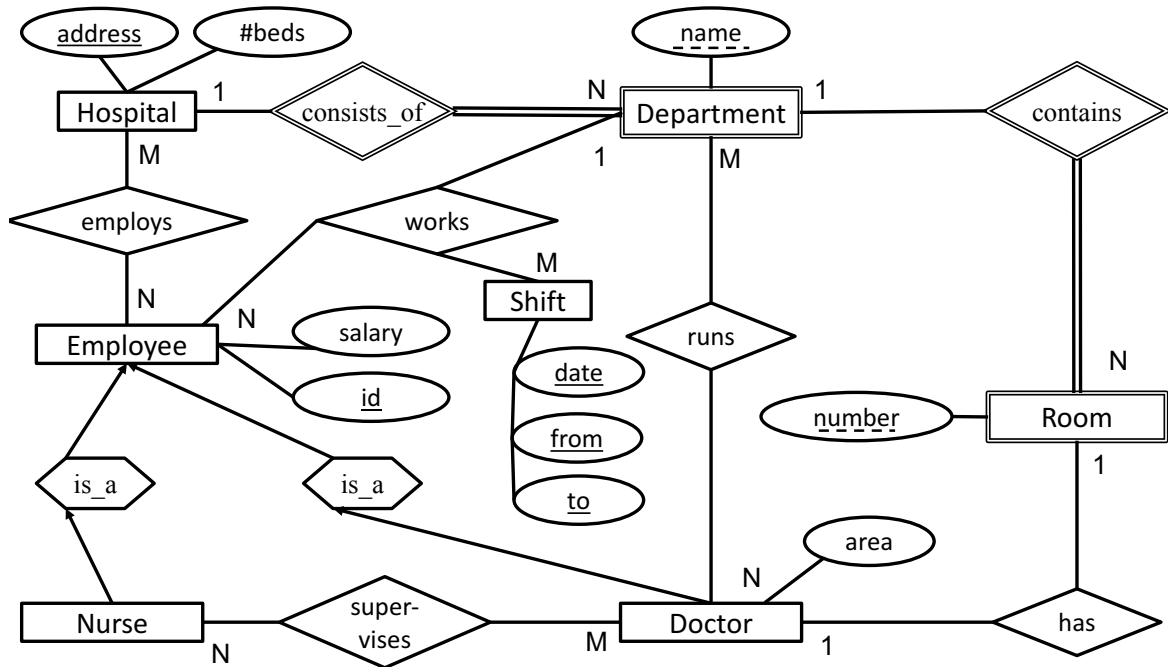
```

In our model the train number is uniquely identifying a connection between two cities (possibly involving several stations). An ICE starting in Munich (*startStationName*) and going to Berlin (*endStationName*) has a unique train number. When the train returns it has a different train number. Therefore, in the *connects* relation, the (*trainNo*, *fromStationName*)-pair and the (*trainNo*, *toStationName*)-pair are both valid keys (as they are both uniquely identifying a tuple in the relation).

Exercise 2

For additional practice, consider the hospital example, again. This time take the entity relationship diagram and transform it into a relational schema. Then, optimize it by eliminating relations.

This is obviously a large example but practice is very helpful. However, if you want to save time, you could focus on the difficult parts: *employs*, *works*, *consists_of*, *Doctors + has*



Solution:

a) Create a relational schema

The un-refined translation yields the following relations for the entities in the model:

- Hospital : {[address : string, #beds : int]} (8)
- Department : {[address : string, name : string]} (9)
- Room : {[address : string, name : string, roomNo : int]} (10)
- Employee : {[id : int, salary : int]} (11)
- Nurse : {[id : int]} (12)
- Doctor : {[id : int, area : string]} (13)
- Shift : {[date : date, from : time, to : time]} (14)

For the relationships in the model, we create the following relations:

$$\text{consists_of} : \{\{\underline{\text{address}} : \text{string}, \text{departmentName} : \text{string}\}\} \quad (15)$$

$$\text{contains} : \{\{\underline{\text{address}} : \text{string}, \text{departmentName} : \text{string}, \underline{\text{roomNo}} : \text{int}\}\} \quad (16)$$

$$\text{employs} : \{\{\underline{\text{address}} : \text{string}, \text{id} : \text{int}\}\} \quad (17)$$

$$\text{supervises} : \{\{\underline{\text{nurseId}} : \text{int}, \underline{\text{doctorId}} : \text{int}\}\} \quad (18)$$

$$\text{doctor_has} : \{\{\underline{\text{doctorId}} : \text{int}, \text{address} : \text{string}, \text{departmentName} : \text{string}, \underline{\text{roomNo}} : \text{int}\}\} \quad (19)$$

$$\text{runs} : \{\{\underline{\text{doctorId}} : \text{int}, \text{address} : \text{string}, \text{name} : \text{string}\}\} \quad (20)$$

$$\text{works} : \{\{\underline{\text{employeeId}} : \text{int}, \underline{\text{date}} : \text{date}, \underline{\text{from}} : \text{time}, \underline{\text{to}} : \text{time}, \text{address} : \text{string}, \text{name} : \text{string}\}\} \quad (21)$$

There are several alternative translation options:

1. The *is_a* relationship could have also been translated by merging the attributes of the *Employee* into the *Nurse* and *Doctor* relation:

$$\text{Nurse} : \{\{\underline{\text{id}} : \text{int}, \text{salary} : \text{int}\}\}$$

$$\text{Doctor} : \{\{\underline{\text{id}} : \text{int}, \text{area} : \text{string}, \text{salary} : \text{int}\}\}$$

2. In the 1:1 relation *has* between *Doctor* and *Room* we could have also chosen the key of the *Room* as a key.

b) Refine the relational schema

Next, we refine the relational schema by combining relations.

All binary relations with 1:1, 1:N, N:1 can be refined in the following way:

First, we can eliminate all relations that originate from weak relationships in the ER-model. In this case we do not have to add additional keys to the entity we merge them into because they already have this key because they are weak entities:

$$(15) \mapsto (9), (16) \mapsto (10)$$

Next, we take care of the *has* relation between *Doctor* and *Room*. This is a 1:1 relation and can therefore be merged into *Doctor* or *Room*. We choose to merge it into *Room*, as this requires us to only add one attribute to *Room* instead of four to *Doctor*:

$$(19) \mapsto (10)$$

Now, there is no binary relation left with a 1:1, 1:N or N:1 functionality. Therefore, we are done and end up with the following relational schema:

$$\begin{aligned} \text{Hospital} & : \{\{\underline{\text{address}} : \text{string}, \text{\#beds} : \text{int}\}\} \\ \text{Department} & : \{\{\underline{\text{address}} : \text{string}, \text{name} : \text{string}\}\} \\ \text{Room} & : \{\{\underline{\text{address}} : \text{string}, \text{name} : \text{string}, \underline{\text{roomNo}} : \text{int}, \text{doctorId} : \text{int}\}\} \\ \text{Employee} & : \{\{\underline{\text{id}} : \text{int}, \text{salary} : \text{int}\}\} \\ \text{Nurse} & : \{\{\underline{\text{id}} : \text{int}\}\} \\ \text{Doctor} & : \{\{\underline{\text{id}} : \text{int}, \text{area} : \text{string}\}\} \\ \text{Shift} & : \{\{\underline{\text{date}} : \text{date}, \underline{\text{from}} : \text{time}, \underline{\text{to}} : \text{time}\}\} \end{aligned}$$

For the relationships in the model, we create the following relations:

employs : {[address : string, id : int]}

supervises : {[nurseId : int, doctorId : int]}

runs : {[doctorId : int, address : string, name : string]}

works : {[employeeId : int, date : date, from : time, to : time,
address : string, name : string]}