Data & Information – Test 2: Solutions

25 May 2018, 13:45–15:15

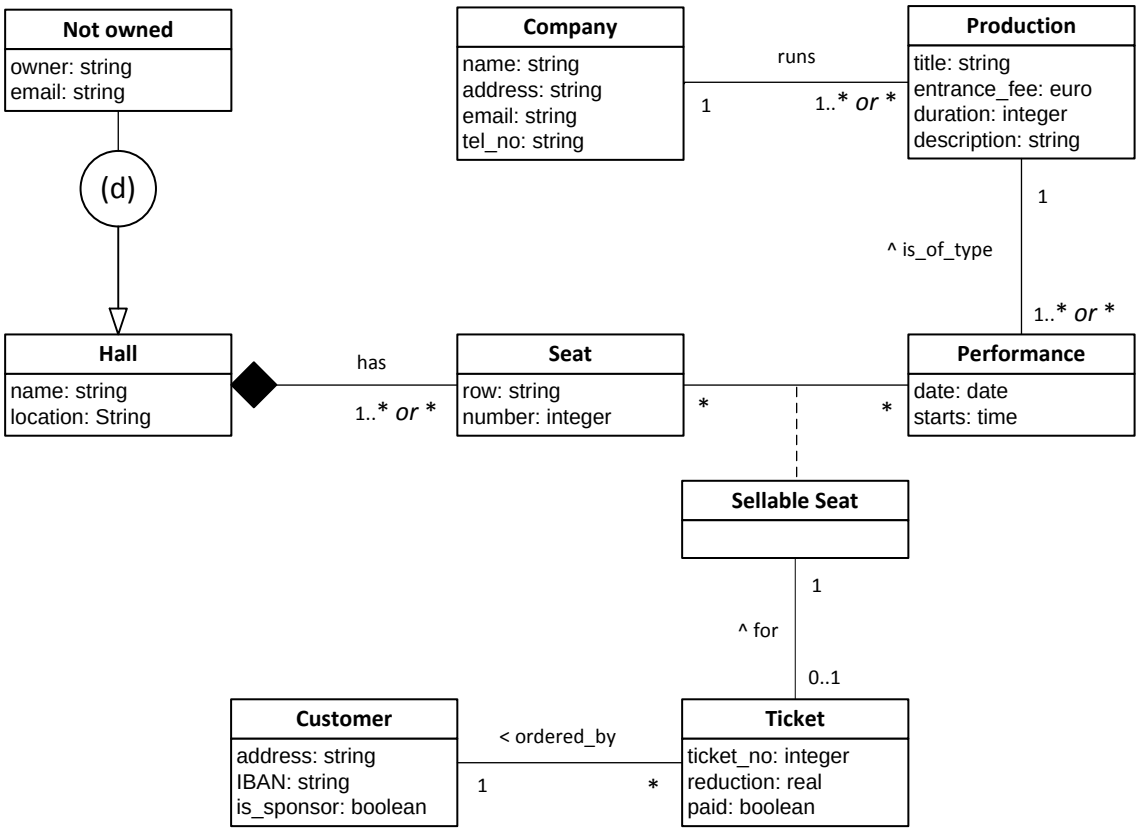
Question 1 (Database Schema)

1a)

```
Customer(cid, address, IBAN, is_sponsor,
      PK(cid));

Sponsor(sid, company_name, yearly_amount, ends,
                free_tickets, contact_person NOT NULL,
      PK(sid),
      FK(sid) REF Customer(cid),
      FK(contact_person) REF Person(pid));

Person(pid, first_name, last_name,
      PK(pid),
      FK(pid) REF Customer(cid),
      CHECK (pid IN (SELECT pid FROM Telephone_no)));

Telephone_no(number,pid NOT NULL,
      PK(number),
      FK(pid) REF Person);

Ticket(ticket_no, reduction, paid, cid NOT NULL,
      PK(ticket_no),
      FK(cid) REF Customer);
```

1b) Alternative I: **only tables for the subclasses**
"Ticket" needs an association "ordered_by" with both subclasses. The difficulty then is to check that there is exactly one customer (either a sponsor or a person) for a ticket.
Alternative II: **only a table for the superclass**
The association "contact_person" would be an association from "Customer" with itself. The difficulty then would be to check that each sponsor has one conctact person, and no sponsors are associated with sponsors or persons with persons.

Question 2 (Class Diagram)



Remarks:

- Not explicitly mentioned in the text is the class ``Performance''. It was stated, however, that a production is performed multiple times. A performance is the execution of a production at a specific time – just like module execution vs. module in the lab exercise, and car vs. car type in the lecture slides.
- Sellable seat is an association class, because it is the combination of a particular seat (in some given hall) and a particular performance that defines it.
- It is possible to add an assocation indicating that a performance “takes place in” a hall. What matters in this class diagram is which seats can be sold for a performance. These seats are by definition in a particular hall, so no information is lost when we leave out the association.
- In a few cases it does not matter whether you indicate * or 1..* as multiplicity. For example: it would be strange if there is a production with 0 performances. But no harm is done in allowing it, so there is no need to add an extra constraint to the database to check that. Also, it could happen that at some point in time the production has been defined but the performances not yet scheduled. This is different from the case that a each customer should have a telephone number. That was explicity demanded, therefore it must be modelled that way.
- A generalization with one subclass (here: “Not owned” as a subclass of “Hall”) is disjoint by definition. Whether or not you indicate it as “d” does not matter.
- The seats (meaning *seat numbers*) as part of a hall are a natural case of composition.
  If the hall is dismantled you could reuse the physical seats, but they would get a new number in their new environment.

Question 3a (Functional dependencies)

a) $T \longrightarrow H$ **Yes** from 6, 2 and 1 we have $T \longrightarrow S$, $S \longrightarrow R$, $R \longrightarrow H$, thus $T \longrightarrow H$

b) $T \longrightarrow N$ **No** A ticket is for a visitor, but s/he may have multiple phone numbers

c) $N \longrightarrow T$ **No** A telephone number is for a unique visitor, but s/he may have multiple tickets

d) $HT \longrightarrow C$ **Yes** $T \longrightarrow C$ holds by 7 and 8, then $HT \longrightarrow C$ holds a fortiori

e) $SP \longrightarrow HC$ **Yes** From 2,1 we have $S \longrightarrow H$, from 8 $P \longrightarrow C$, then also $SP \longrightarrow HC$

f) $V \longrightarrow PC$ **No** A visitor can visit multiple productions of multiple companies

g) $P \longrightarrow VC$ **No** A production has more than one visitor

h) $C \longrightarrow PV$ **No** A company has multiple productions (with multiple visitors)

i) $T \twoheadrightarrow CPV$ **Yes** From 7, 8, 5, we find $T \longrightarrow CPV$, which implies $T \twoheadrightarrow CPV$

Also graded as correct (applying the standard logic about MVDs):

No This would be a MVD if *CPV* is completely unrelated with *HRSN*, which is contradicted by, e.g., $N \longrightarrow V$ (3)

j) $T \twoheadrightarrow HCP$ **Yes** From 6, 1, 2, 7, 8 we find $T \longrightarrow HCP$, which implies $T \twoheadrightarrow HCP$

Also graded as correct (applying the standard logic about MVDs):

No This would be a MVD if *HCP* is completely unrelated with *RSVN*, which is contradicted by, e.g., $R \longrightarrow H$ (1)

Question 3b (Normal forms)

1) From $D \longrightarrow C$, $C \longrightarrow B$ we find $D \longrightarrow B$, from $ACE \longrightarrow D$, $D \longrightarrow B$ we find $ACE \longrightarrow B$.
So

$$\mathcal{F}^+ = \{ ACE \longrightarrow BD,\ C \longrightarrow B,\ D \longrightarrow BC \}$$

Schema R has two candidate keys: ACE, ADE.

FDs which violate the BCNF condition are $C \longrightarrow B$ and $D \longrightarrow BC$ (or $D \longrightarrow C$ from the original $\mathcal{F}$)

2) Start with functional dependency $C \longrightarrow B$.

$C^+ = BC$. Splitting on C we get

$R_1(B,C)$, with $\mathcal{F}_1 = \{ C \longrightarrow B \}$
$R_2(A,C,D,E)$, with $\mathcal{F}_2 = \{ ACE \longrightarrow D,\ D \longrightarrow C \}$
R_1 has candidate key C and is in BCNF
R_2 has candidate keys ACE, ADE and is not in BCNF because of $D \longrightarrow C$.

$D^+ = CD$. Splitting R_2 on D we get

$R_{21}(C,D)$, with $\mathcal{F}_{21} = \{ D \longrightarrow C \}$
$R_{22}(A,D,E)$, with $\mathcal{F}_{22} = \{\ \}$
R_{21} has candidate key D and is in BCNF
R_{22} has candidate key ADE and is in BCNF

Alternatively, start with functional dependency $D \longrightarrow BC$.

$D^+ = BCD$. Splitting on D we get

$R_1(B,C,D)$, with $\mathcal{F}_1 = \{ C \longrightarrow B,\ D \longrightarrow BC \}$
$R_2(A,D,E)$, with $\mathcal{F}_2 = \{\ \}$
R_1 has candidate key D and is not in BCNF, because of $C \longrightarrow B$.
R_2 has candidate key ADE and is in BCNF.

$C^+ = BC$. Splitting R_1 on C we get

$R_{11}(B,C)$, with $\mathcal{F}_{11} = \{ C \longrightarrow B \}$
$R_{12}(C,D)$, with $\mathcal{F}_{12} = \{ D \longrightarrow C \}$
R_{21} has candidate key C and is in BCNF
R_{22} has candidate key D and is in BCNF

3) From the original functional dependencies, $ACE \longrightarrow D$ was lost in the second (resp. first) step. The other FDs in $\mathcal{F}$ still exist in $\mathcal{F}_1 \cup \mathcal{F}_{21} \cup \mathcal{F}_{22}$ (resp. $\mathcal{F}_{11} \cup \mathcal{F}_{12} \cup \mathcal{F}_2$).