# 201300180 Data & Information – Test 2 (1.5 hours)

**22 May 2015, 13:45 – 15:15**

Please note:

- ***Please answer questions 1, 2, and 3 each on a separate sheet of paper***
  (Not on the back side of the previous question, the questions will be distributed to different person for grading).
- You can give your answers in Dutch or English.
- Reference materials are given in the appendices, therefore you are not allowed to bring any study materials to the test

Grade = #points/10

## Question 1 (Database Schema) (30 points)

The Polderland Symphony Orchestra (PSO) faces declining government subsidies. In order to make up for this, the orchestra management has come up with a plan to raise money from the many regular visitors to their concerts.
People who support the orchestra can become 'Friend of the PSO'. In exchange for their financial support they get extra services and privileges: first choice for seats for concerts, meet-and-greets with musicians, the possibility to attend some rehearsals, etc.
A friend must pay a yearly minimum contribution, but it is expected that true fans of the orchestra (many of whom are well-to-do) will contribute a lot more than the minimum.
A friend can invite up to 4 persons to be come 'Friend of a friend'. These persons pay only a small contribution (or it is paid for them by the inviting friend) and get access to most of the services for friends. To keep the terminology for the database design clear and unabiguous, we call regulars friends *primary friends* and friends of friends *secondary friends* (though it is not intended that the marketing department will address them in these terms). In Figure 1 these are denoted by *Friend1* and *Friend2*, respectively.

- The orchestra likes to know the type of relationship between primary and secondary friends (children / family / friends / ...). Different categories could be targeted in a different way to become primary friends themselves. Not in all cases the relationship is known, however.
- Primary friends can become "*Maecenas*" (called after a legendary sponsor of the arts in ancient Rome). A maecenas pays a substantially higher contribution, but they get more privileges, and their names will be mentioned as Maecenas in various brochures.
- All friends can indicate whether they particularly like certain instruments (e.g. violin, flute, trumpet, ...). This can be taken into account for inviting friends to meet-and-greets. For each type of instrument, one of the musicians in the orchestra will be appointed *contact person* for friends.

A class diagram for information about friends of the PSO is shown in Figure 1.

**a)** The specific characteristics of the generalization of friends allow for different database schemas. Explain in 2–4 lines what the different options are to define tables for friends (not taking into account the various associations in the class diagram).

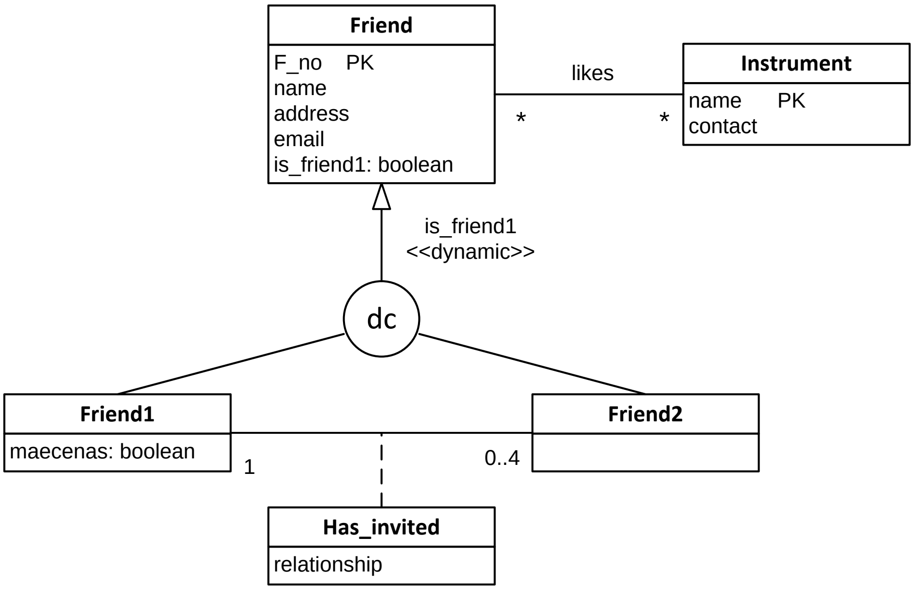



*Figure 1: class diagram for 'Friends of the PSO'*

**b)** Define a database schema for the option under a) that you find easiest to extend with the other information represented in Figure 1. (*Don't be bothered with considerations about query efficiency*). For the information represented by the associations, do not introduce more tables than needed.

For the class *Instrument* the following table already has been defined (referring to an table `Musician` that has been defined elsewhere).

```
Instrument(name, contact,
   PK (name),
   FK (contact) REF Musician(m_id));
```

Give definitions for the remaining tables.

## Question 2 (Class Diagram) (35 points)

Extend the class diagram in Figure 1 with the information given below.
The final diagram should incorporate all classes and associations in Figure 1. It is possible, however, that some attributes will belong to a different class in the final diagram.

The class diagram should cover the following information concerning friends, visitors, and concerts of the Polderland Symphony Orchestra.

- For all customers (friends as well as other customers) it is known which concerts they visited. More precisely: For every customer it is known for which concerts s/he has purchased tickets as well as the number of tickets per concert. (It is not known who the other persons are that accompanied the customer to the concert.)

- A concert consists of one or more works. For example a concert with works of Beethoven could consist of: Ouverture Fidelio, Piano Concerto no. 3, Symphony no. 5.
- Each concert has a date and a location.
- Each concert has a conductor.
- Some concerts (but not all) have a soloist, who is not a member of the orchestra but plays a prominent role in the concert. Occasionally there is more than one soloist (e.g. Beethoven's Triple Concerto needs soloists for piano, violin and cello). A soloist is always associated with a particular instrument. There are no soloists who play different instruments. (They may do so in private, perhaps, but at concerts a soloist always plays the same instrument.)
- A concert can be performed more than once. The typical concert is performed two or three times in the span of one week. Always they have the same conductor and soloist(s), who stay for these concerts and then move on to perform concerts elsewhere in the world.
- A work is identified by the title of the work and the name of the composer. The same work can be part of different concerts.
- Last but not least, all contributions from friends are recorded. For each contribution the amount and the date is known, as well as the friend whose contribution it is.

## Question 3 (35 points)

### 3a) ***Printed on page 4*** *(so that all relevant info is on a single page)*

### 3b) (Normal forms) (20 points)

Consider the relational schema $R(A,B,C,D,E,F,G)$ with functional dependencies $\mathfrak{F}$, defined by

$$\mathfrak{F} = \{ A \rightarrow BC,\ B \rightarrow A,\ DE \rightarrow AF,\ FG \rightarrow EB \}.$$

**1)** Which functional dependencies violate the BCNF condition? Why?

**2)** Apply the algorithm in Appendix 3 to decompose $R$ into a set of relational schemas that are all in BCNF. Please describe how you executed the steps of the decomposition algorithm.

**3)** Which of the functional dependencies of $R$ were lost in the decomposition?

**3a) (Functional dependencies)** (15 points)

For the administration of a car rental service, a relation *R*(*T,C,D,A,R,F,L,E,P*) is defined. For the attributes of *R*, the following holds

1. *T* is a *car type*, e.g. a particular model of a specific manufacturer;
2. *C* is car of a specific type *T*;
3. *D* is a driver (i.e. a customer) who has one or more rentals;
4. *A* is the address of the driver;
5. *R* is a rental by a driver *D* of a car *C*;
6. *F* is the first day (start date) of a rental *R*;
7. *L* is the last day (end date) of a rental *R*;
8. *E* is an employee of the car rental service;
9. *P* is a phone number of an employee *E*.

In addition, the following facts are given:

a. A rental is always for one specific driver.
b. A rental is always for one specific car.
c. If a driver has more rentals, these do not overlap in time, i.e., the first day of the second rental is always after the last day of the first rental.
d. The first and last day of a rental can be the same (for a one-day rental), but (as a consequence of c.) one cannot make two one-day rentals for the same day.
e. A driver has a single address.
f. It is possible, however, that different drives live at the same address.
g. An employee has one or more phone numbers
h. Different employees never have the same phone number.

For each of the follwing – potential – fucntional dependencies (FDs) *i)...viii)* and multivalued dependencies (MVDs) *ix)...x)*, please indicate whether the dependency holds ("yes") or not ("no"). Give a brief motivation for each answer, if possible referring to statements 1...9 and a...e above. Having the right motivation has the same weight as having the right answer.

*i)* $E \longrightarrow P$

*ii)* $L \longrightarrow C$

*iii)* $R \longrightarrow T$

*iv)* $R \longrightarrow AT$

*v)* $DF \longrightarrow R$

*vi)* $L \longrightarrow T$

*vii)* $CD \longrightarrow F$

*viii)* $DFE \longrightarrow T$

*ix)* $R \twoheadrightarrow EP$

*x)* $C \twoheadrightarrow RL$

**3b)** *See page 3.*

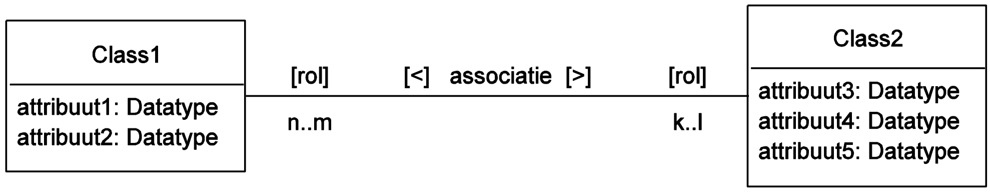
## Appendix 1: Notations for class diagrams

*meta-notation:*

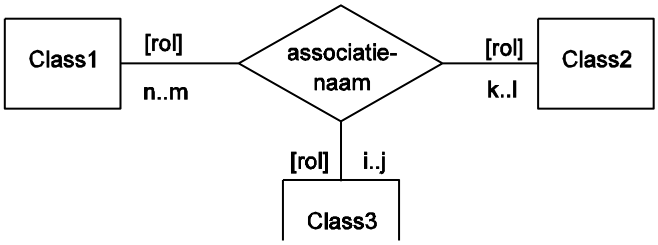
[ ... ]   Optional (can be deleted)

.. | ..   Choice:  one of the given alternatives

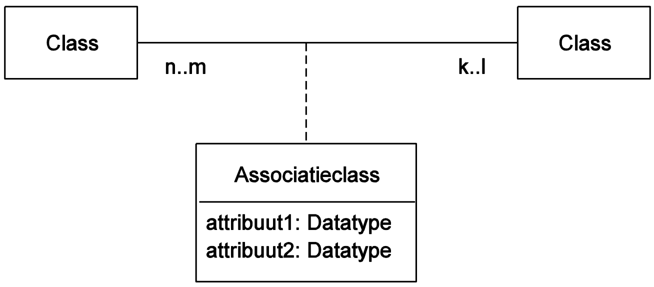
### Class and Association



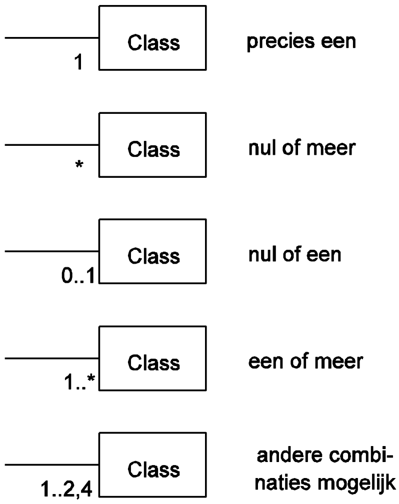

### Ternary (or *n*-ary) association



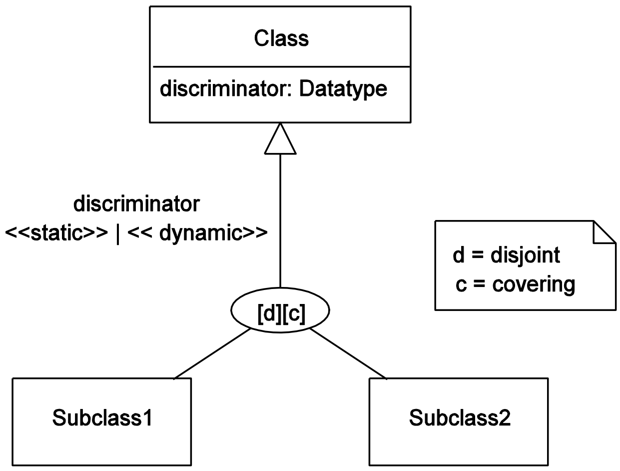

### Association class



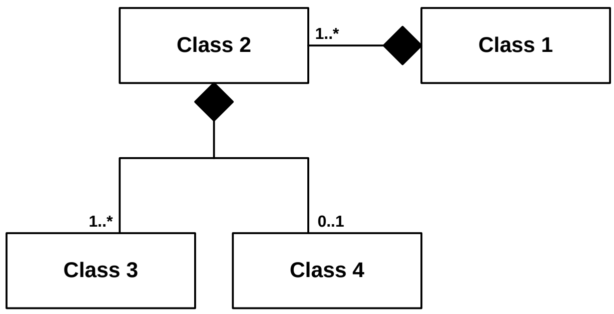

### Multiplicity




### Generalization




### Composition



## Appendix 2: Informal syntax for database schema

Informal syntax ('|' for choice and '[ ]' for optional):

```
CREATE TABLE table (
  column [NOT NULL] [UNIQUE] [PRIMARY KEY]
  [ , column ... ]
  [ , PRIMARY KEY (column, ... ) ]
  [ , FOREIGN KEY (column, ... ) REFERENCES table(column, ...)
    [ , FOREIGN KEY ...] ]
  [ , CHECK ( condition ) ]
);
```

Examples of condition:

```
column = value [ (OR | AND) [NOT] column <> value ] |
column IS [NOT] NULL |
column [NOT] IN (value, ...) |
...
```

## Appendix 3: Losless BCNF decomposition algorithm

Definition of BCNF:
A relational schema is in BCNF if for every nontrivial functional dependency the left-hand side is a superkey.

Decomposition algorithm:
Let $R$ be a relational schema with a set of functional dependencies $\mathfrak{F}$.
Let $X \rightarrow Y$ be a functional dependency in $\mathfrak{F}$ which violates the BCNF constraint.

- Decompose R into
  - $R_1(X^+)$
  - $R_2(Z)$ with $Z = \{X\} \cup \{$attributes of $R$ not in $X^+\}$.
- For $i = 1,2$:
  - determine $\mathfrak{F}_i$ for $R_i$ by restricting $\mathfrak{F}^+$ to functional dependencies within $R_i$
  - if $R_i$ is not in BCNF, recursively apply the algorithm