

# Introduction to Database Systems

Radim Bača

Department of Computer Science, FEECS

radim.baca@vsb.cz

dbedu.cs.vsb.cz

# Content

- Introduction
- Conceptual model
  - Tools for conceptual modeling
  - E-R model
  - UML model

# Information sysems

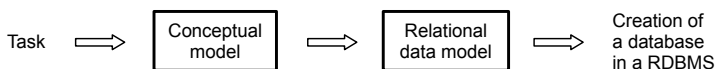
- An information system (IS) is a special type of a software work
- When developing an IS, we use the recommended software development techniques
- However, we focus only on a part of an IS development that concerns **databases** in this subject

# Information sysems

- An information system (IS) is a special type of a software work
- When developing an IS, we use the recommended software development techniques
- However, we focus only on a part of an IS development that concerns **databases** in this subject

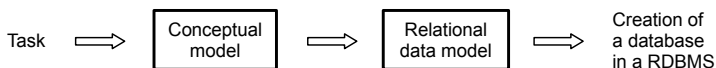
# Data analysis

- Three steps for a database development:
  - **Task** - written specification of the task
  - **Conceptual modeling** (conceptual model) - a logical description of a database
  - **Database scheme design** (relational data model) - a description of a database defined for a concrete database system
  - **Physical design** - a concrete implementation of data files (CREATE TABLE ..., CREATE INDEX ...)



# Data analysis

- Three steps for a database development:
  - **Task** - written specification of the task
  - **Conceptual modeling** (conceptual model) - a logical description of a database
  - **Database scheme design** (relational data model) - a description of a database defined for a concrete database system
  - **Physical design** - a concrete implementation of data files (CREATE TABLE ..., CREATE INDEX ...)



# Task

- Task formulate the requirements
- Features:
  - Description in a text form
  - Ambiguous, incomplete, inaccurate
  - Understandable to both the client and the developer

# Task Specification

Task specification constructively answers the following questions:

- WHY? The motivation behind the IS creation
- FOR WHAT? Problems solved by the IS
- WHO? Roles of users that will use the IS
- INPUTS? The entities that we want to store in our system
- OUTPUTS? The most important views
- FUNCTIONS? Non trivial functions



# Task Specification

Task specification constructively answers the following questions:

- **WHY?** The motivation behind the IS creation
- **FOR WHAT?** Problems solved by the IS
- **WHO?** Roles of users that will use the IS
- **INPUTS?** The entities that we want to store in our system
- **OUTPUTS?** The most important views
- **FUNCTIONS?** Non trivial functions

# WHO?

- Describe roles of users in the company that will be using your IS
- Those people have specific job/role in the company, describe it
- Do not write roles USER and ADMIN! ADMIN is not a role in a company!

## Example - WHY?

- We need a room reservation system in our building
- There is no central authority and we often argue about the rooms between the colleagues
- Some colleagues even take personally if the room is not empty for their meeting

## Example - FOR WHAT?

- The software will provide a simple way to organize reservations in a building
- The system will not only provide a way to reserve a meeting room, but it will also offer a possibility to define a single authority for a meeting room
- The authority will have an authorization to cancel a reservation in that room and the software will implement a notification system as well that will send an email in such case

## Example - WHO?

- Company employee – a person that work in a company and have to organize a meeting time to time. This is the person that will have a possibility to create a reservation.
- Company manager – a person that manage a larger group of people and need a meeting room for them and for himself quite often.
- Codebook keeper – a person that will keep the codebooks in the database up-to-date

## Example - INPUTS?

- **Person** information such as name, email, office, phone and flag indicating whether it is a office manager or not (Company employee, Company manager).
- For each **room** we store its building, floor, room number and room authority (Codebook keeper).
- Information about a **reservation** such as reservation date, meeting interval, room and person who created the reservation (Company employee, Company manager).
- Each person can create many reservations.

## Example - OUTPUTS?

- List of an employee future reservations that are sorted according to the meeting time ascending order. Employee may also list the history of his reservations according to a specified time interval. (Company employee)
- List of a future room reservations. The system should offer also a week schedule for a room. (Company employee, Company manager)
- List of free rooms for a specific time. (Company employee)

## Example - FUNCTIONS?

- Possibility to create a reservation in a meeting room even though there are already reservations. In such a case, the system will propose an empty meeting room in a similar time for each overlapping reservation.
- The system will notify by email each user whose reservation was shifted.



# The Project Topic

When selecting the topic of the project consider the following:

- The topic **have to be familiar** to you
  - Do not select bank accounts or bank loans if you never worked in a bank
  - Do not select food delivery if you have no experience with it
- It should **not be easily replaceable** by a content management system
  - Database of battles and persons in a World War II
  - Database of space objects

# Conceptual Modeling

- is a process of a development of a system description that is used to design and implement a database application
- is independent of database
- defines restrictions put on data

# Linear Notation & Basic Concepts

- Linear notation:

Student (stID, name, birth\_year)

Subject (suID, name, study\_year)

# Linear Notation & Basic Concepts

- Linear notation:

Student (stID, name, birth\_year)

Subject (suID, name, study\_year)

- **Entity type**

# Linear Notation & Basic Concepts

- Linear notation:

Student (stID, name, birth\_year)

Subject (suID, name, study\_year)

- **Entity type**
- **Attribute**

# Linear Notation & Basic Concepts

- Linear notation:

Student (stID, name, birth\_year)

Subject (suID, name, study\_year)

- **Entity type**
- **Attribute**
- **Key**

# Linear Notation & Basic Concepts

- Linear notation:

Student (stID, name, birth\_year)

Subject (suID, name, study\_year)

- **Entity type**
- **Attribute**
- **Key**
- **Entity** – object of a reality (one instance of entity type)

# Relationship

- **Relationship** - describes a relationship among entity types
  - Linear notation:  
`RELATIONSHIP (EntityType1, ... , EntityTypen)`
  - Two or more entity types can be in a relationship
- **Relationship with attributes** - a relationship containing also attributes specifying properties of the relationship

- *Example of a relationship (the relationship Studies)*

`STUDIES (Student, Subject)`

- *Example of a relationship with attributes (the relationship Studies)*

`STUDIES (Student, Subject, gained_points)`



# Relationship

- **Relationship** - describes a relationship among entity types
  - Linear notation:  
`RELATIONSHIP (EntityType1, ... , EntityTypen)`
  - Two or more entity types can be in a relationship
- **Relationship with attributes** - a relationship containing also attributes specifying properties of the relationship
- *Example of a relationship (the relationship Studies)*  
`STUDIES (Student, Subject)`
- *Example of a relationship with attributes (the relationship Studies)*  
`STUDIES (Student, Subject, gained_points)`

## Relationship, Example 1

- *We would like to model a situation where one student coordinate another student. What is the proper way using linear notation?*

# Relationship, Example 1

- *We would like to model a situation where one student coordinate another student. What is the proper way using linear notation?*

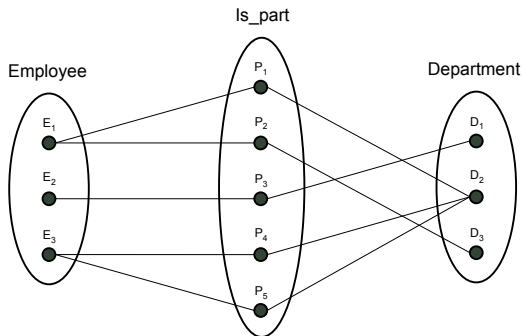
`STUDIES (Student, Student)`

## Cardinality of a Relationship

- We distinguish relationships according to number of entities entering a relationship
- A relationship between two entity types (a so-called binary relationship) can be of the following types:
  - 1:1, 1:N, M:N
- *Consider a company with the entity types Department and Employee.*
  - Cardinality 1:1 - an employee can be a chief only of one department
  - Cardinality 1:N - an employee can belong only to one department
  - Cardinality M:N - an employee can belong to several departments

# Cardinality

- In order to determine cardinality, it can be useful to draw the following diagram:



# Mandatory/Obligatory Relationships

- Some entities has to have a relationship and some does not:
  - Mandatory relationship - each entity has to have a relationship
  - Obligatory relationship - there can be entities without relationship

- Linear notation:

```
RELATIONSHIP (EntityType1 : (min, max) ,  
              EntityType2 : (min, max) )
```

## Relationships, Example 2

- *A teacher does not have to teach, but a subject has to have a teacher. A teacher is teaching many subjects and a subject can be taught by many teachers.*

## Relationships, Example 2

- *A teacher does not have to teach, but a subject has to have a teacher. A teacher is teaching many subjects and a subject can be taught by many teachers.*

TEACH (Teacher : (0,M) , Subject : (1,N) )



# Constraints

- **Constraints** provide additional information for conceptual model
- They are invariants of the database which have to be always satisfied
- Their typically concern:
  - an attribute value (e.g., the format of email)
  - relationship among entities (e.g., a department has to have its chief)

## E-R diagram (ERD)

- Graphic representation of conceptual model
- Unfortunately, there is no standard for it, therefore, one can come across many different notations of ERDs
- Let us mention just of them:
  - Chen's notation
  - Crow's foot notation
    - Oracle data modeler
    - Toad data modeler

# Types of ERD

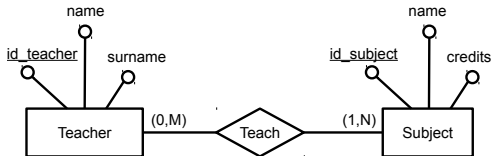


FIGURE: Chen's notation

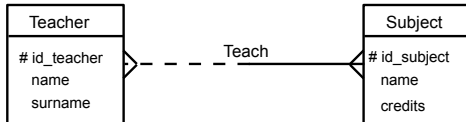


FIGURE: Crow's foot notation - Oracle

# Types of ERD

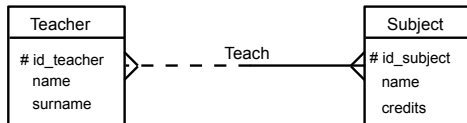


FIGURE: Crow's foot notation - Oracle

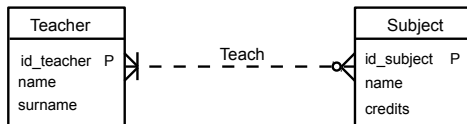


FIGURE: Crow's foot notation - Toad

## Cardinality in ERD

- Cardinality is represented by a number or by crow's foot
- See previous slide

# Mandatory/Obligatory relationships

- It is solved by many different ways, but there are two basic categories:
  - Information is contained in pair (min, max), which determine the cardinality as well
  - We use some graphical symbol that determine mandatory and obligatory relationships

## Mandatory/Obligatory relationships - (min, max)

- It is used in UML
- Pair (min, max) determine maximum and minimum number of entities that are in the relationship
- Having the TEACH rel. between Subject and Teacher
  - Subject : (0, N) - Teacher does not have to have a subject
  - Subject : (1, N) - Teacher has to have a subject
  - Subject : (0, 1) - Teacher does not have to have a subject
  - Subject : (1, 1) - Teacher has to have exactly one subject

# M/O relationships - graphical symbol

- *Subject has to have at least one teacher and teacher does not have to teach anything*

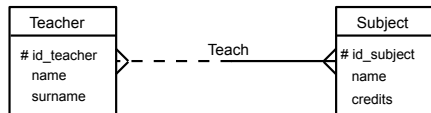


FIGURE: Crow's foot notation - Oracle

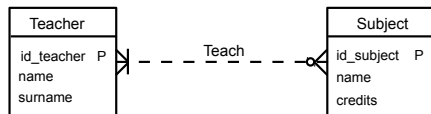


FIGURE: Crow's foot notation - Toad



# M/O relationships - graphical symbol

- *Subject has to have at least one teacher and teacher does not have to teach anything*

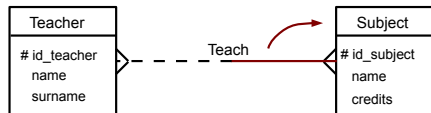


FIGURE: Crow's foot notation - Oracle

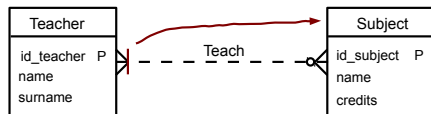


FIGURE: Crow's foot notation - Toad

# M/O relationships - graphical symbol

- *subject has to have at least one teacher and teacher does not have to teach anything*

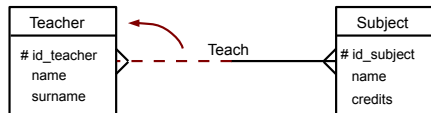


FIGURE: Crow's foot notation - Oracle

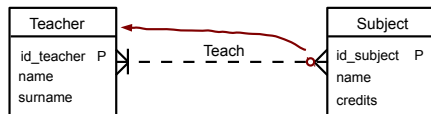


FIGURE: Crow's foot notation - Toad

## Weak entity types

- Sometimes a key is formed by attributes belonging to another entity types
- The entity make sense only with respect to a different entity
- Then we speak about a so-called *weak entity type*
- *Diploma thesis is determined by both its title and its supervisor.*

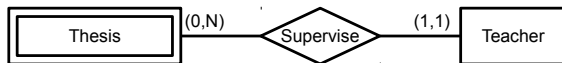


FIGURE: Chen's notation

## Weak entity types - Oracle, TOAD

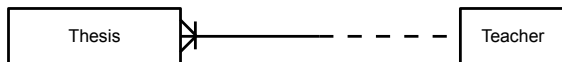


FIGURE: Crow's foot notation - Oracle

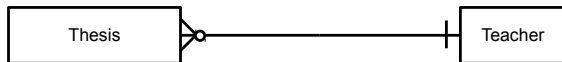


FIGURE: Crow's foot notation - Toad

## Weak entity types - Oracle, TOAD

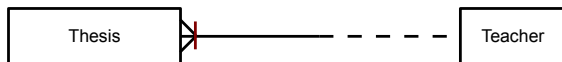
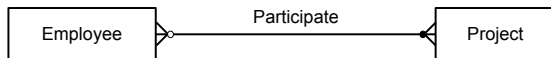


FIGURE: Crow's foot notation - Oracle

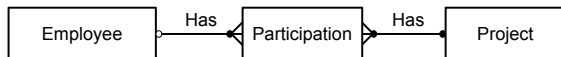


FIGURE: Crow's foot notation - Toad

# Decomposition of a relationship



- In the relational scheme, we decompose the relationship  $M : N$  by using a table



## Description of Data model in Project

- Table containing a more detailed description of attributes
- Every table corresponds to a single entity type

### User

	Data type	Lenght	Key	Null	Index	AC	Meaning
login	varchar	10	Y	N	Y		user's login
fname	varchar	20	N	N	N		user's first name
lname	varchar	20	N	N	Y		user's last name
phone	number	12	N	Y	N		phone number
type	varchar	10	N	N	N	1	user's cathegory
last_visit	Timestamp		N	Y	N		date of user's last login to the IS

1: data type must be one of these: admin, bidder, or user

# UML

- UML is another tool which enables us to design conceptual model of a system
- It represents an alternative to E-R diagrams (ERDs)
- It is a visual, object-oriented language modeling structural and dynamic aspects of a software work
- Unlike ERD, UML is a collection of modeling techniques that are applied to various aspects of software development
- Every UML technique provides different static or dynamic perspective of an application (a so-called model)

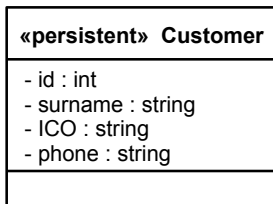


# UML versus ERD

<i>UML</i>	<i>ERD</i>
class	entity type
object	entity
attribute	attribute
association	relationship

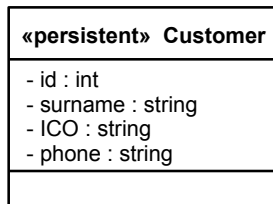
# Class

- describes a structure and a behavior of an object representing an instance of the class
- Three parts: class name, attributes, and methods



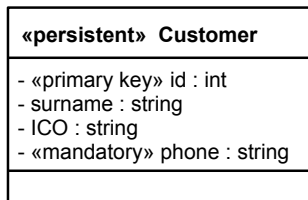
# Class

- describes a structure and a behavior of an object representing an instance of the class
- Three parts: class name, attributes, and methods



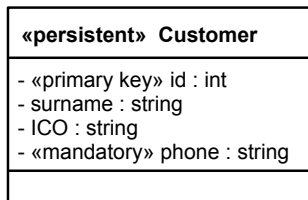
# Stereotypes

- extend the description (of attributes and class)
- are written between the symbols « »
- The notation « Persistent » means that the class (attribute) will be mapped to the database, i.e., to the relational database scheme
- Furthermore, we can specify a primary key, mandatory attributes, etc.



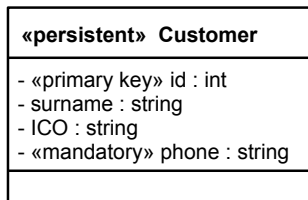
# Stereotypes

- extend the description (of attributes and class)
- are written between the symbols « »
- The notation « Persistent » means that the class (attribute) will be mapped to the database, i.e., to the relational database scheme
- Furthermore, we can specify a primary key, mandatory attributes, etc.



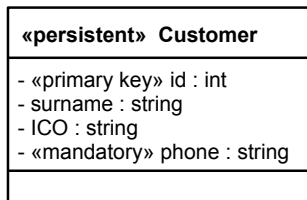
# Stereotypes

- extend the description (of attributes and class)
- are written between the symbols << >>
- The notation << Persistent >> means that the class (attribute) will be mapped to the database, i.e., to the relational database scheme
- Furthermore, we can specify a primary key, mandatory attributes, etc.



# Stereotypes

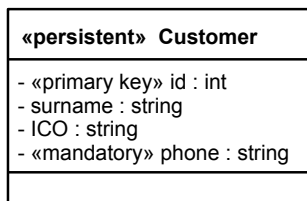
- extend the description (of attributes and class)
- are written between the symbols << >>
- The notation << Persistent >> means that the class (attribute) will be mapped to the database, i.e., to the relational database scheme
- Furthermore, we can specify a primary key, mandatory attributes, etc.



# Syntax of attributes

`[visibility] [«stereotype»] attribute : [type]`

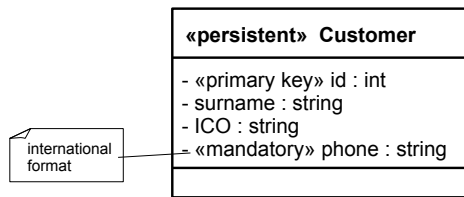
- `visibility` - gains the value
  - `+` for a public attribute
  - `#` for a protected attribute
  - `-` for a private attribute
- `«stereotype»` - adds more semantics to an attribute
- `attribute` - name of an attribute



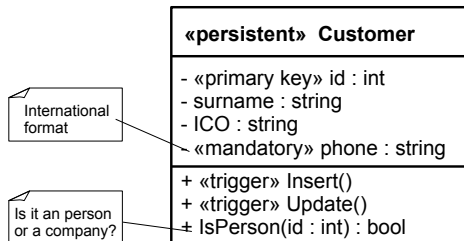


## Example: attributes and notes

Notes are used to specify an attribute in more detail, e.g., for further specification of format of the telephone number



## Example: Operations [1/2]



- The stereotype `trigger` indicates that a method is a trigger (i.e., a code which runs automatically during a DML operation)
- The method `IsPerson` returns an information, whether the customer is a person or a company; we should also decide, whether the method will be a stored procedure or whether it will be a part of the application

# Association

- is an equivalent to a relationship in ERD
- Again, the cardinality can be 1:1, 1:N, M:N
- When describing an association, we mention: name, role, cardinality, and whether the association is mandatory or not
- By an arrow we can determine the direction of an association

# Conceptual modeling summary

- Linear notation of entity types and linear notation of relationships among entity types
- Graphic illustration of data model:
  - E-R diagram or UML diagram
  - transformed diagram for database scheme (see the next lecture)
- Data model
- List of constraints

## Software for conceptual modeling

- Microsoft SQL Server 2008 Management Studio
- Oracle SQL Developer Data Modeler
- MySQL Workbench, (previously MySQL GUI Tools)
- Toad Data Modeler
- And a lot more:

[http://www.databaseanswers.org/modelling\\_tools.htm](http://www.databaseanswers.org/modelling_tools.htm)

## References

- R. Elmasri, S. Navathe. Fundamentals of Database Systems, Addison Wesley, ISBN 0-321-36957-2, 2010.
- UDBS web pages at <http://dbedu.cs.vsb.cz>