Database Design

DBS can take care automatically of many things – but the user has to specify

- Requirements of the application
- Characteristics of the data

Two important concepts during DBS design:
- Data Model: How to describe the data?
- Data Schema: Concrete description of the data (using the chosen data model)
Data modeling

Excerpt of the Real World

Conceptual Schema (E/R- or UML-Schema)

Manual/intellectual Modeling

Semi-automatic Transformation

Relational Schema

XML Schema

Network Schema

Object-oriented Schema
Modeling a small example application: E/R

Real World: University

Students

Conceptual Modeling

Students

Lectures

StudNr
Name

attend

Lectures

LectureNr
Title

requires
Logical Data Models

- Network Model
- Hierarchical Model
- Relational Data Model
- XML Model
- Object-oriented Data Model
  - Object-relational Schema
- Deductive Data Model

* [Michael Stonebraker: What Goes Around Comes Around]
### Relational Data Model

<table>
<thead>
<tr>
<th>Students</th>
<th>attend</th>
<th>Lectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudNr</td>
<td>StudNr</td>
<td>LectureNr</td>
</tr>
<tr>
<td>26120</td>
<td>25403</td>
<td>5022</td>
</tr>
<tr>
<td>25403</td>
<td>26120</td>
<td>5001</td>
</tr>
</tbody>
</table>

**Select** Name  
**From** Students, attend, Lectures  
**Where** Students.StudNr = attend.StudNr and attend.LectureNr = Lectures.LectureNr and Lectures.Title = 'Grundzüge';
Modeling a small example application: UML

<table>
<thead>
<tr>
<th>Students</th>
<th>+Attendee (1..*) attend +Successor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+StudNr : int</td>
</tr>
<tr>
<td></td>
<td>+Name : String</td>
</tr>
<tr>
<td></td>
<td>+Semester : int</td>
</tr>
<tr>
<td></td>
<td>+GPA() : float</td>
</tr>
<tr>
<td></td>
<td>+SumWeeklyHours() : short</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lectures</th>
<th>requires</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+LectureNr : int</td>
</tr>
<tr>
<td></td>
<td>+Title : String</td>
</tr>
<tr>
<td></td>
<td>+WeeklyHours : int</td>
</tr>
<tr>
<td></td>
<td>+NumberAttendees() : int</td>
</tr>
<tr>
<td></td>
<td>+FailureRate() : float</td>
</tr>
</tbody>
</table>
Database Design

Database Abstraction Layers

1. Conceptual Design
2. Logical Design
3. Physical Database Design
Phases of Database Design

- Information Requirements
- Processing Requirements
- DBMS-Characteristics
- Hardware/OS-Characteristics

Requirements Engineering -> Scope Statement

Conceptual Modeling

ER Schema

Conceptual Design

Logical Modeling

Logical Schema

Logical Design

Physical Modeling

Physical Database Design
Software Development and Ability to Communicate

- How the customer explained it
- How the Project Leader understood it
- How the Analyst designed it
- How the Programmer wrote it
- How the Business Consultant described it
- How the project was documented
- What operations installed
- How the customer was billed
- How it was supported
- What the customer really needed
Schema Design

Approach in principle:

**Information Requirements**
- Interview
- Brainstorming
- Document‘s Analy.
- ...

**Semantical Data Modeling**
- ERM
- UML
- ...

**Logical Data Modeling**
- Hierarchical
- network
- relational
- object-oriented
- ...

**Database Installation / Tuning**
- IMS
- UDS
- DB2
- Ozone
- ...

**Time**

**Conceptual Schema Design**

**Logical Schema Design**

**Physical Schema Design**
Requirements Engineering

Entity description
Relation description
Process description

...
University Employees
- Quantity: 1000
- Attributes

- **EmpNumber**
  - Type: Integer
  - Domain: 0...999,999,999
  - Defined: 100%
  - Identifying: yes
  - Example: 007

- **Salary**
  - Type: decimal
  - Length: (7,2)
  - Unit: Euro per month
  - Defined: 10%
  - Identifying: no

- **Level**
  - Type: String
  - Length: 2
  - Defined: 100%
  - Identifying: no
  - Example: W2
Relation Description: exam

Involved Objects:
- Professor as Tester
- Student as Testee
- Lecture as Test Subject

Attributes of the Relation:
- Date
- Time
- Grade

Quantity: 100,000 per year
Process Description:

**Issue a Certificate**

- Frequency: semiannually

- Required Data
  * Tests
  * Examination Rules
  * Student’s Records
  * ...

- Priority: high

- Data Volume to be processed
  * 500 Students
  * 3000 Tests
  * 10 Versions of Examination Rules
Creating a Specification

The actual analysis is an iterative process:
• Customer tells developer his/her needs
• Developer notes everything down (s/he understood) in his/her „language“ . . .
• . . . and translates it into the "language" of the customer
• This is shown to the customer who does not agree with everything
• Change requests are agreed on
• Back to step 2
Phases of Database Design

- Information Requirements
- Processing Requirements
- Hardware/OS-Characteristics
- DBMS-Characteristics

1. Requirements Engineering
   - Scope Statement
2. Conceptual Modeling
   - ER Schema
3. Logical Modeling
   - Logical Schema
4. Physical Modeling
   - Physical Database Design

<table>
<thead>
<tr>
<th>Relation</th>
<th>Att1</th>
<th>Att2</th>
</tr>
</thead>
</table>
The ideal design (the ideal specification) is
• unique
• complete
• comprehensible (for all participants)
• nonredundant
• . . . and not reachable in reality
Entity/Relationship-Modeling

Entity:
- Students

Relationship:
- Attend

Attribute (property):
- LectureNr
- Title
- Weekly hours

Key (identification):
- StudNr
- Name
- Semester

Role:
- Attendee
University Schema

- **Students**
  - StudNr
  - Name
  - Semester

- **Lectures**
  - LectureNr
  - Weekly Hours
  - Title

- **Assistants**
  - PersNr
  - Name
  - Area

- **Professors**
  - PersNr
  - Name
  - Level
  - Room

**Relationships**
- Students attend Lectures
- Lectures require prerequisite
- Lectures follow-up
- Grade is tested by Lectures
- Grade is given by Professors
- Assistants work for Professors

**This diagram represents a University Schema with entities such as Students, Lectures, Assistants, and Professors, along with their attributes and relationships.**
Functionalities

\[ R \subseteq E_1 \times E_2 \]
Relationship 1:1

E₁ 1 → R 1 → E₂

e₁ out of E₁ takes part in 1 relation of type R

E₂ 1 → R 1 → E₁

e₂ out of E₂ takes part in 1 relation of type R

Example:

Car 1 → has 1 → License Plate

one car has one license plate
one license plate belongs to one car
Relationship 1:N

E₁ → 1 → R → N → E₂

e₁ out of E₁ takes part in N relations of type R

e₂ out of E₂ takes part in 1 relation of type R

Example:

Mentors → 1 → advise → N → Students

one mentor advises several students

one student is advised by one mentor
Relationship N:M

$e_1$ out of $E_1$ takes part in $M$ relations of type $R$
$e_2$ out of $E_2$ takes part in $N$ relation of type $R$

Example:

one actor stars in several movies
one movie has several actors
Recursive Relationship 1:N

Relationship 1:N

\[
\begin{array}{c}
E_1 \\
\text{role A} \\
\text{role B} \\
R \\
N \\
\end{array}
\]

e_1 \text{ out of } E_1 \text{ takes part in role A in } N \text{ relations of type } R

e_1 \text{ out of } E_1 \text{ takes part in role B in } 1 \text{ relation of type } R

Example:

\[
\begin{array}{c}
\text{Persons} \\
\text{mother} \\
\text{child} \\
\text{rel-ship} \\
N \\
\end{array}
\]

one person is mother of several persons (children)
one person is child of one person (mother)
Functionalities in $n$-ary Relationships

$$R : E_1 \times \ldots \times E_{k-1} \times E_{k+1} \times \ldots \times E_n \rightarrow E_k$$
Example Seminar

supervise : Professors x Students → Topics

supervise : Topics x Students → Professors
Thereby induced Consistency Constraints

1. Students may work on only one topic with the same professor (to cover a broad spectrum)

2. Students may work on the same topic only once – thus they may not work on the same topic again with another professor

3. Professors can reuse the same topic – i.e. give the same topic to different students

4. One topic can be given by different professors – but to different students
Occurrence of the Relationship *supervise*

Dashed lines represent illegal occurrences
One more Example

3-ary relationship:

One checkup is performed by one expert with several patients

One Patient gets only one checkup from one expert

One checkup is performed at one patient only by one expert
(min, max)-Notation

For every $e_i \in E_i$ there are

- at least $min_i$ tuples $(..., e_i, ...)$ $\in R$ and
- at most $max_i$ tuples $(..., e_i, ...)$ $\in R$
Example (min, max)

one mentor advises up to 20 students
one student is advised by exactly one mentor
Excercise for next class

Inform yourself about unary – binary – ternary relationships

Discussion / new examples next class!
Weak Entities

- Relationship between "strong" and "weak" type is 1:N (or 1:1 in rare cases) - why not N:M?
- The existence of a room depends on the existence of the associated building
- RoomNr is unique only within the building
- Key of Rooms is: RoomNr and BldNr
Tests as weak entity type

- Several professors design one test
- Several lectures are inquired in one test
Generalization / Specialization:

\[ G \xleftarrow{\text{Is-a}} S \]

**Example:**

\[ \text{animal} \xleftarrow{\text{Is-a}} \text{cat} \]
Generalization University

Diagram:

- University-Members
  - is-a
  - Students
    - StudNr
  - Employees
    - PersNr
  - is-a
    - Assistents
      - Area
  - is-a
    - Professors
      - Level
      - Room
Conclusion

University schema with generalization and (min, max)-notation
Aggregation

Database System Concepts for Non-Computer Scientists WS 2018/2019
Vehicles

is-a

Non-mot. Vehicles

is-a

Bikes

Part-of

Frames

Part-of

Pipes

Mot. Vehicles

is-a

Motorbikes

Part-of

Wheels

Part-of

Rims

Part-of

Handlebars

Part-of

Spokes

Aggregation and Generalization