Teil I

What are Databases?
Teil II

Relational Databases – Data as Tables
Teil III

Entity-Relationship Model
Entity-Relationship Model

1 Database Model
Entity-Relationship Model

1. Database Model

2. Semantics of Database Models
Entity-Relationship Model

1. Database Model

2. Semantics of Database Models

3. ER Model
Entity-Relationship Model

1. Database Model
2. Semantics of Database Models
3. ER Model
4. Further ER Model Concepts
Educational objective for today . . .

- Knowing the concepts of the entity-relationship model
Educational objective for today . . .

- Knowing the concepts of the entity-relationship model
- Ability to conceptually model an application domain
Basics of Database Models

A **database model** is a system of concepts to describe databases. It defines the syntax and semantics of database descriptions for a database system.

- Database descriptions = database schemata
A Database Model Defines . . .

1. **Static properties**
   - Objects
   - Relationships
   - including the primitive data types, which can describe data about the relations and objects,

2. **Dynamic properties** such as
   - Operations
   - Relationships between operations,

3. **Integrity constraints** of
   - Objects
   - Operations
Database Models

- Classical database models are especially suited for
  - Large amounts of data with a relatively static structure and
  - Describing static properties and integrity constraints

- Design models: (E)ER model, UML, . . .

- Realization models: relational model, object-oriented models, . . .
## Databases versus Programming Languages

<table>
<thead>
<tr>
<th>Database concept</th>
<th>Type system of a programming language</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Database Model</strong></td>
<td><strong>Type system</strong></td>
</tr>
<tr>
<td>Relation, Attribute ...</td>
<td>int, struct ...</td>
</tr>
<tr>
<td><strong>Database schema</strong></td>
<td>Declaration of variable</td>
</tr>
<tr>
<td>relation WINE = (...)</td>
<td>var x: int, y: struct Wine</td>
</tr>
<tr>
<td><strong>Database</strong></td>
<td><strong>Values</strong></td>
</tr>
<tr>
<td>WINE(4961, 'Chardonnay', 'White', ...)</td>
<td>42, 'Cabernet Sauvignon'</td>
</tr>
</tbody>
</table>
## Levels of Abstraction

<table>
<thead>
<tr>
<th>Models</th>
<th>Data</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstract</td>
<td>entity-relationship model</td>
<td>structograms</td>
</tr>
<tr>
<td>concrete</td>
<td>hierarchical model</td>
<td>Pascal</td>
</tr>
<tr>
<td></td>
<td>network model</td>
<td>C, C++</td>
</tr>
<tr>
<td></td>
<td>relational model</td>
<td>Java, C#</td>
</tr>
</tbody>
</table>
Overview of Database Models

close to implementation → abstract

Mid 1960

1970

1980

1990

2000

HM

NWM

RM

SQL

NF²

eNF²

ODDM (C++)

OODM

ORM / SQL-99

ORM / SQL-99

ER

SDM

OEM

Sattler / Saake

Database Systems

Last Edited: October 2014
Overview of Database Models /2

- HM: hierarchical model, NWM: network model, RM: relational model
- NF²: model of nested (non-first-normal form = NF²) relations, eNF²: extended NF² model
- ER: entity-relationship model, SDM: semantic data models
- OODM / C++: object-oriented data models based on object-oriented programming languages, such as C++, OEM: object-oriented design models (e.g., UML), ORDM: object-relational data models
Semantics of Database Models

- Not focus of this lecture …
- Idea: formalizes temporal developments of databases
  - Sequences of states $\langle \sigma_0, \sigma_1, \sigma_2, \ldots \rangle$
  - Every state $\sigma_i$ is a concrete database
  - Sequence is created by changes in the database
- Distinction between all possible values and current values
  - $\mu$: possible values: “Which wines could exist?”
  - $\sigma$: current values: “Which wines are currently stored in this state?” (sigma for “state”)
Sequences of Database States

(Zinfandel, Red, 2004, Helena)
(Pinot Noir, Red, 2001, Creek)

σ₀ -> σ₁

(Zinfandel, Red, 2004, Helena)
(Pinot Noir, Red, 2001, Creek)
(Creek Shiraz, Red, 2003, Creek)

σ₁ -> σ₂
The ER Model

Entity: object of the real or a virtual world, about which information is to be stored, e.g., Products (wine, catalog), winemaker or critic; but also information about events, e.g., Orders

Relationship: describes a relationship between entities, e.g., a customer orders a wine or wine is being offered by a winemaker

Attribute: represents a property of an entity or a relationship, e.g., Name of customer, Color of a wine or Date of an order
ER Example

- **Grape**
  - Name
  - Side dish
  - **Made of**
    - Proportion
    - Color
  - **Located in**
    - Producer
  - **Produced by**
    - Wine
  - **Recommends**
    - Dish
  - Res. Sugar
  - Year

- **Dish**
  - Name
  - **Recommends**
    - Critic

- **Critic**
  - Name
  - **Organization**

- **Wine**
  - Name
  - **Made of**
    - Grape
  - **Produced by**
    - Producer

- **Producer**
  - Name
  - **Located in**
    - Area
  - **Has**
    - License
      - LicenseNo.
      - Amount

- **Area**
  - Name
  - **Located in**
    - Country
  - **Region**
    - Vineyard
    - Address

- **Organization**
  - License

- **Country**
  - License
  - **LicenseNo.**

- **Critic**
  - **Side dish**
  - Name
  - **Organization**

- **License**
  - LicenseNo.
  - Amount

- **Producer**
  - Name
  - **Located in**
    - Region

- **Area**
  - Name
  - **Located in**
    - Country

- **Organization**
  - License
    - LicenseNo..
    - Amount

- **License**
  - LicenseNo.
  - Amount
Values

- **Values**: primitive elements of data, which can be represented directly
- Value domains are described by **datatypes**, which, apart from the set of possible values, also characterize the basic operations on those values
- ER model: pre-defined primitive datatypes, such as the integers `int`, the character sequences `string`, dates `date` etc.
- Every datatype represents a domain, including operations and predicates on values of this domain
Entities

- **Entities** are the pieces of information to be represented in a database.
- In contrast to values, entities cannot be represented directly. They can only be observed through their properties.
- Entities are grouped according to their **entity types**, such as $E_1, E_2, \ldots$

**Wine**

Set of current entities:

$$\sigma(E_1) = \{e_1, e_2, \ldots, e_n\}$$
Attribute

- **Attribute** models properties of entities or relationships

- All entities of an entity type have the same kinds of properties; attributes are therefore declared for the entity type

\[ E(A_1 : D_1, \ldots, A_m : D_m) \]

- Textual notation \[ E(A_1 : D_1, \ldots, A_m : D_m) \]
Key-based Identification

- Key attributes: Subset of all attributes of an entity type $E(A_1, \ldots, A_m)$

$$\{S_1, \ldots, S_k\} \subseteq \{A_1, \ldots, A_m\}$$

- In every state of the database, current values of the key attributes uniquely identify instances of the entity type $E$

- If multiple keys would be possible: Choice of a primary key

- Notation: Highlight by underlining:

$$E(\ldots, \underline{S_1}, \ldots, \underline{S_i}, \ldots)$$
Relationship Types

- Relationships between entities are grouped into relationship types.
- In general: arbitrary number \( n \geq 2 \) of entity types can participate in a relationship type.
- Every \( n \)-ary relationship type \( R \) refers to \( n \) entity types \( E_1, \ldots, E_n \).
- Instances of a relationship type

\[
\sigma(R) \subseteq \sigma(E_1) \times \sigma(E_2) \times \cdots \times \sigma(E_n)
\]
Relationship Types /2

- **Notation**

  ![ER Diagram]

  - Textual notation: $R(E_1, E_2, \ldots, E_n)$
  - If an entity type participates in a relationship type multiple times: roles can be assigned

    married(Wife: Person, Husband: Person)
Relationship Attributes

- Relationships can also have attributes
- Attributes are declared at the relationship type; this also holds for the set of possible values \( \sim \) relationship attributes

Textual notation: \( R(E_1, \ldots, E_n; A_1, \ldots, A_k) \)
Characteristics of Relationships

- **Cardinality** or degree:
  - Number of participating entity types
  - Often: binary
  - Example: *Supplier supplies Product*

- **Cardinality** or functionality:
  - Number of incoming instances of an entity type
  - Forms: 1:1, 1:n, m:n
  - Represents integrity constraints
  - Example: *maximum of 5 Products per Order*
Binary vs. N-ary Relationships

Dish

 Recommends

 Wine

Critic
Binary vs. N-ary Relationships

Dish → Recommends → Wine → Critic

Dish → D-C → Wine

Dish ∈ D-W
Instances in the Example

Dish

Wine

Critic

\[ d_1 \]
\[ d_2 \]
\[ w_1 \]
\[ w_2 \]
\[ c_1 \]
\[ c_2 \]
Instances in the Example
Reconstruction of Instances

But also:

\[ d_1 \rightarrow c_1 \rightarrow w_1 \]
\[ d_1 \rightarrow c_2 \rightarrow w_1 \]
\[ d_2 \rightarrow c_1 \rightarrow w_2 \]
\[ d_2 \rightarrow c_2 \rightarrow w_1 \]
Reconstruction of Instances

\[ d_1 - c_1 - w_1 \]
Reconstruction of Instances

\[ d_1 \rightarrow c_1 \rightarrow w_1 \]
\[ d_1 \rightarrow c_2 \rightarrow w_2 \]
Reconstruction of Instances

- $d_1 - c_1 - w_1$
- $d_1 - c_2 - w_2$
- $d_2 - c_2 - w_1$
Reconstruction of Instances

- $d_1 - c_1 - w_1$
- $d_1 - c_2 - w_2$
- $d_2 - c_2 - w_1$
- But also: $d_1 - c_2 - w_1$
1:1-Relationships

- Every $e_1$ of entity type $E_1$ is assigned to at most one entity $e_2$ out of $E_2$ and vice versa.

- Examples: *Brochure describes Product*, *Husband is married to Wife*
1:N Relationships

- Every entity $e_1$ of entity type $E_1$ is assigned to an arbitrary number of entities $E_2$, but for every entity $e_2$, there is at most one $e_1$ in $E_1$
- Examples: Supplier \textit{supplies} Product, Mother \textit{has} Children
N:1 Relationship

- Inverse of 1:N, also functional relationship
- Binary relationships that define a function:
  Every entity of entity type $E_1$ is assigned to at most one entity of entity type $E_2$.

$$R : E_1 \rightarrow E_2$$
1:1 Relationship

Licence ← Has → Producer
M:N Relationships

- No restrictions
- Example: *Order consists of Products*
[min, max] Notation

- Restricts the possible number of times an instance of an entity type can participate in a relationship by giving a minimum and a maximum value.
- Notation for expressing cardinalities in a relationship type:

  \[ R(E_1, \ldots, E_i[min_i, max_i], \ldots, E_n) \]

- Cardinality constraints: \( min_i \leq |\{r \mid r \in R \land r.E_i = e_i\}| \leq max_i \)
- Special notation for \( max_i \) is \( * \)
Expressing Cardinalities

- \([0, \ast]\) means “no restrictions” (default)
- \(R(E_1[0, 1], E_2)\) corresponds to a (partial) functional relationship \(R : E_1 \rightarrow E_2\), because every instance out of \(E_1\) is assigned to at most one instance out of \(E_2\)
- Total functional relationships are modelled by \(R(E_1[1, 1], E_2)\)
Expressing Cardinalities: Examples

- Partial functional relationship
  
  \[
  \text{stored\_on(}\text{Product}[0,1],\text{Shelf}[0,3])
  \]
  
  “Every product in the warehouse is stored on one shelf. However, products that are currently out of stock are not assigned to a shelf. At most three products can share the same shelf.”

- Total functional relationship
  
  \[
  \text{supplies(}\text{Supplier}[0,*],\text{Product}[1,1])
  \]
  
  “Every product is supplied by exactly one supplier. However, a supplier can very well supply more than one product.”
Alternative Ways to Express Cardinalities

![Diagram showing two alternative ways to express cardinalities between Product, Delivered By, and Supplier entities.]

1. **Product** has a cardinality of [1,1] associated with **Supplier**, with a multiplicity of [0,*].
2. **Product** has a multiplicity of N associated with **Supplier**, with a cardinality of 1.
Dependent Entity Types

- **Dependent Entity Type**: Identification through functional relationship

![Diagram of Vintage Year Belongs To Wine]

- Dependent entities in the ER model: Functional relationship used as key
Dependent Entity Types /2

- Possible instantiations for dependent entities

- Name: Pinot Noir
  - Color: Red
  - Year: 2004
    - Res. Sugar: 1,2

- Name: Riesling Reserve
  - Color: Weiß
  - Year: 2003
    - Res. Sugar: 1,4

- Name: Zinfandel
  - Color: Red
  - Year: 1999
    - Res. Sugar: 6,7

- Name: Riesling Reserve
  - Color: Weiß
  - Year: 2004
    - Res. Sugar: 1,2
Dependent Entity Types /3

- Alternative notation

```
<table>
<thead>
<tr>
<th>Vintage Year</th>
<th>Belongs To</th>
<th>Wine</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Res. Sugar</td>
<td>Year</td>
<td>Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Color</td>
</tr>
</tbody>
</table>
```

Sattler / Saake

Database Systems

Last Edited: October 2014
The IS-A Relationship

- Specialization/generalization relationship or IS-A relationship
- Textual notation: $E_1$ IS-A $E_2$
- IS-A relationship semantically corresponds to an injective functional relationship
Properties of the IS-A Relationship

- Every sparkling wine instance is assigned to exactly one wine instance.
  - sparkling wine instances are identified by their functional IS-A relationship.
- Not every wine is a sparkling wine.
- Attributes of the entity type Wine also apply to sparkling wines: “inherited” attributes.
  
  \[
  \text{Sparkling\_wine} (\text{Name, Color, Production})
  \]
  
  of Wine

- Not only attribute declarations are inherited, but also the current values of each instance.
Instantiations of IS-A Relationship
Alternative Notation for IS-A Relationship

Entity: Sparkling Wine
- Production

Entity: Wine
- Name
- Color

Sparkling Wine is a type of Wine.
Expressing Cardinalities: IS-A

- It holds for every relationship $E_1$ IS-A $E_2$ that: $\text{IS-A}(E_1[1, 1], E_2[0, 1])$
- Every instance of $E_1$ participates exactly once in the IS-A relationship, whereas instances of the supertype $E_2$ do not have to participate
- This does not affect aspects like attribute inheritance
Optionality of Attributes

Entity-Relationship Model

Further ER Model Concepts

Vineyard
Address
Producer
Located In
Area
Name
Country
Region

Sattler / Saake
Database Systems
Last Edited: October 2014
### Overview of Concepts

<table>
<thead>
<tr>
<th>Term</th>
<th>Informal Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity</td>
<td>The piece of information to be represented</td>
</tr>
<tr>
<td>Entity type</td>
<td>Grouping of entities with the same properties</td>
</tr>
<tr>
<td>Relationship type</td>
<td>Grouping of relationships between entities</td>
</tr>
<tr>
<td>Attribute</td>
<td>Property value of an entity or a relationship</td>
</tr>
<tr>
<td>Key</td>
<td>Identifying property of an entity</td>
</tr>
<tr>
<td>Cardinalities</td>
<td>Restrict relationship types with regards to the number of times an entity can participate in a relationship</td>
</tr>
<tr>
<td>Degree</td>
<td>Number of entity types that participate in a relationship type</td>
</tr>
<tr>
<td>Functional relationship</td>
<td>Relationship Type with functional property</td>
</tr>
<tr>
<td>Dependent entities</td>
<td>Entities that cannot exist independently from other entities</td>
</tr>
<tr>
<td>IS-A relationship</td>
<td>Specialization of entity types</td>
</tr>
<tr>
<td>Optionality</td>
<td>Attribute or functional relationships as partial functions</td>
</tr>
</tbody>
</table>
Summary

- Database model, database schema, database (instance)
Summary

- Database model, database schema, database (instance)
- Entity-relationship model
Summary

- Database model, database schema, database (instance)
- Entity-relationship model
- Further concepts of the ER model
Summary

- Database model, database schema, database (instance)
- Entity-relationship model
- Further concepts of the ER model

*Based on: chapter 3 in Datenbanken - Konzepte und Sprachen von Gunter Saake, Kai-Uwe Sattler und Andreas Heuer* and chapter 7 in Fundamentals of Database Systems by Ramez Elmasri and Shamkant B. Navathe
Control Questions

What defines a database model? What is the distinction between model and schema?
Control Questions

- What defines a database model? What is the distinction between model and schema?
- Which concepts does the ER model define?
Control Questions

- What defines a database model? What is the distinction between model and schema?
- Which concepts does the ER model define?
- Which properties characterize relationship types?
Control Questions

- What defines a database model? What is the distinction between model and schema?
- Which concepts does the ER model define?
- Which properties characterize relationship types?
- How are dependent entity types different from regular entity types?