合肥大學
HEFEI UNIVERSITY

Datenbanken

30. Konzeptuelles Schema: Schwache Entitäten

Thomas Weise (汤卫思)
tweise@hfuu.edu.cn

Institute of Applied Optimization (IAO)
School of Artificial Intelligence and Big Data
Hefei University
Hefei, Anhui, China

应用优化研究所
人工智能与大数据学院
合肥大学
中国安徽省合肥市

Version: 2025-11-19

Databases

Dies ist ein Kurs über Datenbanken an der Universität Hefei (合肥大学).

Die Webseite mit dem Lehrmaterial dieses Kurses ist https://thomasweise.github.io/databases (siehe auch den QR-Kode unten rechts). Dort können Sie das Kursbuch (in Englisch) und diese Slides finden. Das Repository mit den Beispielen finden Sie unter https://github.com/thomasWeise/databasesCode.

Outline

Einleitung

Einleitung

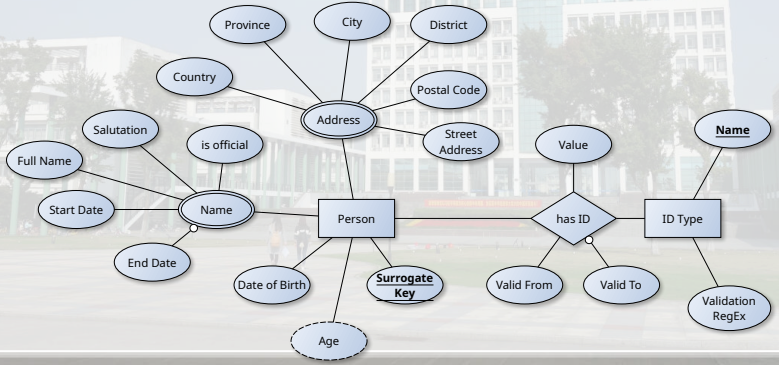
- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.



Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas...

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute.*

- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.
- Wenn wir die Regel von oben anwenden, dann kann jede Person nur eine ID eines bestimmten Typs haben.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.
- Wenn wir die Regel von oben anwenden, dann kann jede Person nur eine ID eines bestimmten Typs haben.
- Nur eine Mobiltelefonnummer.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.
- Wenn wir die Regel von oben anwenden, dann kann jede Person nur eine ID eines bestimmten Typs haben.
- Nur eine Mobiltelefonnummer. Nur eine Reisepassnummer.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.
- Wenn wir die Regel von oben anwenden, dann kann jede Person nur eine ID eines bestimmten Typs haben.
- Nur eine Mobiltelefonnummer. Nur eine Reisepassnummer. Nur eine Email-Adresse.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.
- Wenn wir die Regel von oben anwenden, dann kann jede Person nur eine ID eines bestimmten Typs haben.
- Nur eine Mobiltelefonnummer. Nur eine Reisepassnummer. Nur eine Email-Adresse.
- Reisepässe laufen aber ab. Eine Person hat also wahrscheinlich verschiedene Reisepassnummern über die Zeit, die sie hier bleibt.

Einleitung

- In der letzten Einheit haben wir gute Fortschritte beim Modellieren unserer Lehre-Management-Plattform gemacht.
- Besonders cool war, wie wir sowohl IDs als auch Kommunikationsmöglichkeiten mit Hilfe des neuen Entitätstyps *ID Type* modelliert haben.
- Dann schauen wir uns das ERD nochmal an.
- Und dann erinnern wir uns an etwas: *Beziehungstypen haben keine Schlüsselattribute. Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten identifiziert.*
- Welche Entitäten nehmen denn an unserer *Has ID*-Beziehung teil?
- *Person*-Entitäten, die durch ihren Ersatzschlüssel identifiziert werden, und *ID Type*-Entitäten, die durch ihren Namen identifiziert werden.
- Wenn wir die Regel von oben anwenden, dann kann jede Person nur eine ID eines bestimmten Typs haben.
- Nur eine Mobiltelefonnummer. Nur eine Reisepassnummer. Nur eine Email-Adresse.
- Reisepässe laufen aber ab. Eine Person hat also wahrscheinlich verschiedene Reisepassnummern über die Zeit, die sie hier bleibt.
- Wir haben das Problem also noch nicht ganz gelöst.

Schwachte Entitäten

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.
- Aber wir wollen das auch im konzeptuellen Modell richtig lösen.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.
- Aber wir wollen das auch im konzeptuellen Modell richtig lösen.
- Denn das konzeptuelle Modell definiert die Daten, die in unsere Datenbank kommen.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.
- Aber wir wollen das auch im konzeptuellen Modell richtig lösen.
- Denn das konzeptuelle Modell definiert die Daten, die in unsere Datenbank kommen.
- Es muss korrekt sein.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.
- Aber wir wollen das auch im konzeptuellen Modell richtig lösen.
- Denn das konzeptuelle Modell definiert die Daten, die in unsere Datenbank kommen.
- Es muss korrekt sein.
- Das logische Modell, das wir später entwickeln, darf nicht davon abweichen.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.
- Aber wir wollen das auch im konzeptuellen Modell richtig lösen.
- Denn das konzeptuelle Modell definiert die Daten, die in unsere Datenbank kommen.
- Es muss korrekt sein.
- Das logische Modell, das wir später entwickeln, darf nicht davon abweichen.
- Sonst bekommen wir irgendwann albtraumhafte Probleme, wenn das Projekt später gewartet und weiterentwickelt werden soll.

Konzeptuelles Problem

- Das Problem existiert eher auf der konzeptuellen Ebene.
- Natürlich könnten wir die Entitäten und Beziehungen auf der technischen Ebene so implementieren, wie wir sie *meinen*.
- Aber wir wollen das auch im konzeptuellen Modell richtig lösen.
- Denn das konzeptuelle Modell definiert die Daten, die in unsere Datenbank kommen.
- Es muss korrekt sein.
- Das logische Modell, das wir später entwickeln, darf nicht davon abweichen.
- Sonst bekommen wir irgendwann albtraumhafte Probleme, wenn das Projekt später gewartet und weiterentwickelt werden soll.
- Glücklicherweise können wir eine Lösung für das Problem in verschiedenen Klassen von Entitätstypen finden[48].

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

- Bisher haben wir alle unserer Daten als starke Entitäten modelliert.

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

- Bisher haben wir alle unserer Daten als starke Entitäten modelliert.
- Aber es gibt auch schwache Entitäten[41,47,48].

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

- Bisher haben wir alle unserer Daten als starke Entitäten modelliert.
- Aber es gibt auch schwache Entitäten[41,47,48].

Definition: Schwache Entität

Eine *schwache Entität* (EN: *weak entity*) wird nur durch ihre Attributwerte *und* den/die Primärschlüssel von einer oder mehreren anderen Entitäten identifiziert.

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

- Bisher haben wir alle unserer Daten als starke Entitäten modelliert.
- Aber es gibt auch schwache Entitäten[41,47,48].

Definition: Schwache Entität

Eine *schwache Entität* (EN: *weak entity*) wird nur durch ihre Attributwerte *und* den/die Primärschlüssel von einer oder mehreren anderen Entitäten identifiziert.

- Eine schwache Entität kann nicht alleine existieren.

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

- Bisher haben wir alle unserer Daten als starke Entitäten modelliert.
- Aber es gibt auch schwache Entitäten[41,47,48].

Definition: Schwache Entität

Eine *schwache Entität* (EN: *weak entity*) wird nur durch ihre Attributwerte *und* den/die Primärschlüssel von einer oder mehreren anderen Entitäten identifiziert.

- Eine schwache Entität kann nicht alleine existieren.
- Ihre Existenz hängt von der Existenz mindestens einer anderen Entität ab, welche ihr *Besitzer* (EN: *owner*) genannt wird.

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

Definition: Schwache Entität

Eine *schwache Entität* (EN: *weak entity*) wird nur durch ihre Attributwerte *und* den/die Primärschlüssel von einer oder mehreren anderen Entitäten identifiziert.

- Eine schwache Entität kann nicht alleine existieren.
- Ihre Existenz hängt von der Existenz mindestens einer anderen Entität ab, welche ihr *Besitzer* (EN: *owner*) genannt wird.
- Sie hat keinen Primärschlüssel.

Klassen von Entitätstypen

Definition: Starke Entität

Eine *starke Entität* (EN: *strong Entity*) existiert unabhängig von anderen Entitäten und hat ihren eigenen Primärschlüssel.

Definition: Schwache Entität

Eine *schwache Entität* (EN: *weak entity*) wird nur durch ihre Attributwerte *und* den/die Primärschlüssel von einer oder mehreren anderen Entitäten identifiziert.

- Eine schwache Entität kann nicht alleine existieren.
- Ihre Existenz hängt von der Existenz mindestens einer anderen Entität ab, welche ihr *Besitzer* (EN: *owner*) genannt wird.
- Sie hat keinen Primärschlüssel.
- Sie hat einen *Teilschlüssel* (EN: *partial key*), also eine Menge von Attributen, die sie zusammen mit den Primärschlüsseln ihrer Besitzern identifizieren.

Identifizierende Beziehungen

Definition: Identifizierende Beziehung

Eine *identifizierende Beziehung* (EN: *identifying relationship*) verbindet eine starke Entität mit einer schwachen Entität.

Identifizierende Beziehungen

Definition: Identifizierende Beziehung

Eine *identifizierende Beziehung* (EN: *identifying relationship*) verbindet eine starke Entität mit einer schwachen Entität.

- In einem ERD werden starke Entitäten durch Rechtecke repräsentiert.

Identifizierende Beziehungen

Definition: Identifizierende Beziehung

Eine *identifizierende Beziehung* (EN: *identifying relationship*) verbindet eine starke Entität mit einer schwachen Entität.

- In einem ERD werden starke Entitäten durch Rechtecke repräsentiert.
- Bisher haben wir nur starke Entitäten modelliert.

Identifizierende Beziehungen

Definition: Identifizierende Beziehung

Eine *identifizierende Beziehung* (EN: *identifying relationship*) verbindet eine starke Entität mit einer schwachen Entität.

- In einem ERD werden starke Entitäten durch Rechtecke repräsentiert.
- Bisher haben wir nur starke Entitäten modelliert.
- Schwache Entitäten werden durch Rechtecke mit doppeltem Rand dargestellt.

Identifizierende Beziehungen

Definition: Identifizierende Beziehung

Eine *identifizierende Beziehung* (EN: *identifying relationship*) verbindet eine starke Entität mit einer schwachen Entität.

- In einem ERD werden starke Entitäten durch Rechtecke repräsentiert.
- Bisher haben wir nur starke Entitäten modelliert.
- Schwache Entitäten werden durch Rechtecke mit doppeltem Rand dargestellt.
- Identifizierende Beziehungen, die sie mit anderen Entitäten verbinden, werden durch eine Raute (EN: *diamond*) mit doppeltem Rand dargestellt.

Identifizierende Beziehungen

Definition: Identifizierende Beziehung

Eine *identifizierende Beziehung* (EN: *identifying relationship*) verbindet eine starke Entität mit einer schwachen Entität.

- In einem ERD werden starke Entitäten durch Rechtecke repräsentiert.
- Bisher haben wir nur starke Entitäten modelliert.
- Schwache Entitäten werden durch Rechtecke mit doppeltem Rand dargestellt.
- Identifizierende Beziehungen, die sie mit anderen Entitäten verbinden, werden durch eine Raute (EN: *diamond*) mit doppeltem Rand dargestellt.
- (Eigentlich sprechen wir hier von Entitätstypen und Beziehungstypen, aber das auszuschreiben macht den Text schwerer zu lesen. . .)

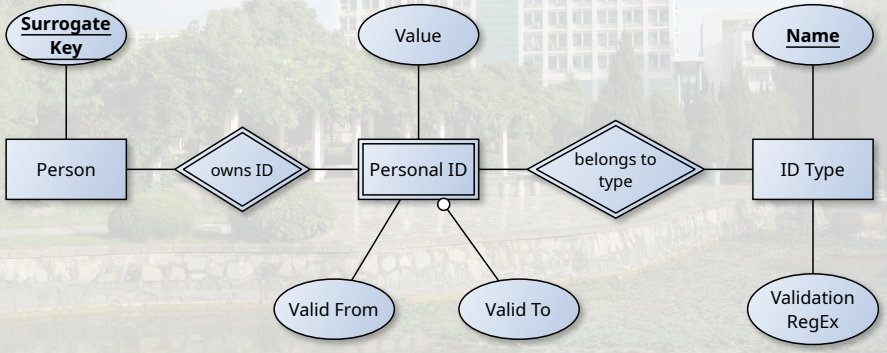
Beispiel

has ID als Schwache Entität

- Wenn wir unser „ID-Problem“ nochmal anschauen, dann können wir die Beziehung *has ID* als schwache Entität modellieren.

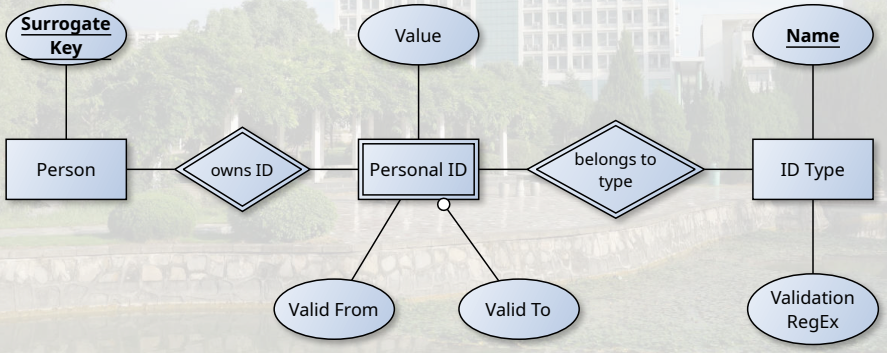
has ID als Schwache Entität

- Wenn wir unser „ID-Problem“ nochmal anschauen, dann können wir die Beziehung *has ID* als schwache Entität modellieren.
- Die schwache Entität würde durch ihren eigenen Attributwerte zusammen mit dem Primärschlüssel der entsprechenden *Person*-Entität and den Primärschlüssel der entsprechenden *ID type*-Entität identifiziert.



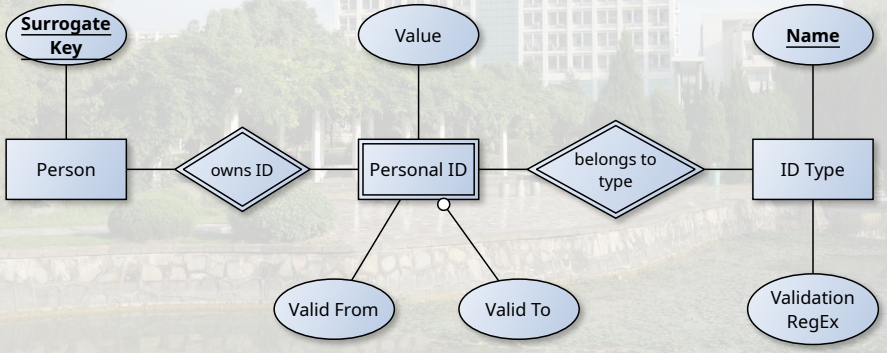
has ID als Schwache Entität

- Die schwache Entität würde durch ihren eigenen Attributwerte zusammen mit dem Primärschlüssel der entsprechenden *Person*-Entität and den Primärschlüssel der entsprechenden *ID type*-Entität identifiziert.
- Sie wäre eine schwache Entität die von zwei starken Entitäten abhängt.



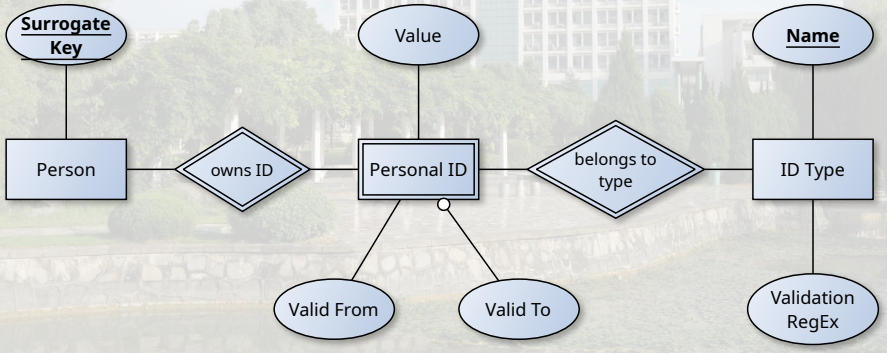
has ID als Schwache Entität

- Sie wäre eine schwache Entität die von zwei starken Entitäten abhängt.
- Wir nennen die neue schwache Entität *Personal ID*.



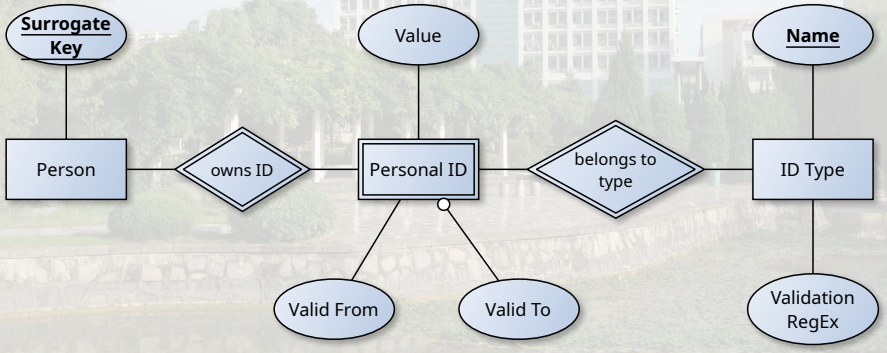
has ID als Schwache Entität

- Wir nennen die neue schwache Entität *Personal ID*.
- Sie hat genau die selben Attribute, wie die Beziehung, die sie ersetzt.



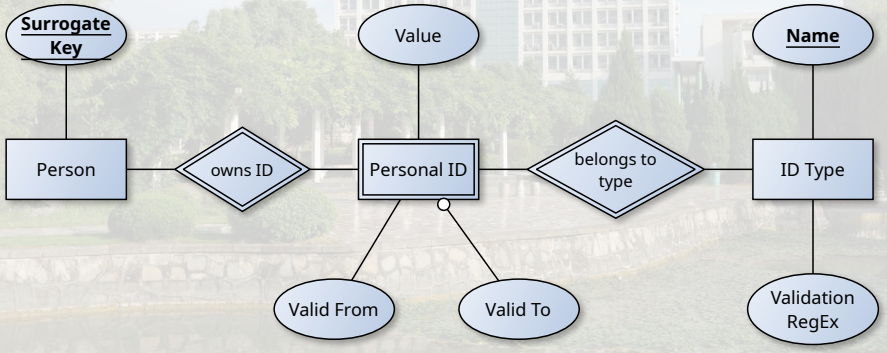
has ID als Schwache Entität

- Sie hat genau die selben Attribute, wie die Beziehung, die sie ersetzt.
- Sie ist mit einer identifizierenden Beziehung mit den Entitätstypen *Person* und *ID Type* verbunden.



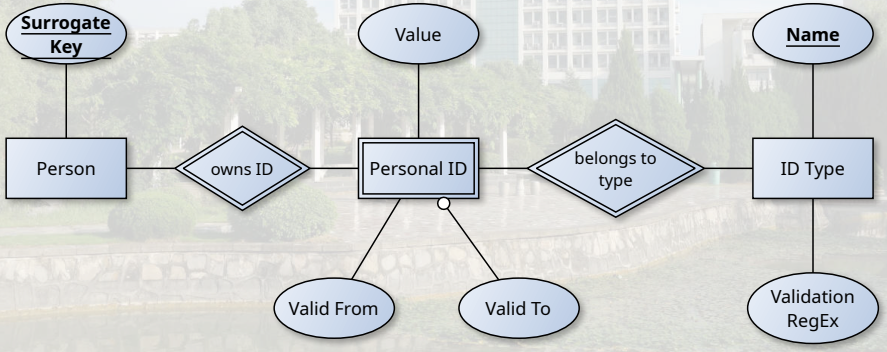
has ID als Schwache Entität

- Sie ist mit einer identifizierenden Beziehung mit den Entitätstypen *Person* und *ID Type* verbunden.
- Jede *Personal ID*-Entität wird identifiziert durch ihre Attribute *Value*, *Valid From*, und *Valid To* (die den Teilschlüssel bilden) zusammen mit den Primärschlüsseln *Surrogate Key* der *Person*-Entität und *Name* der *ID Type*-Entität.



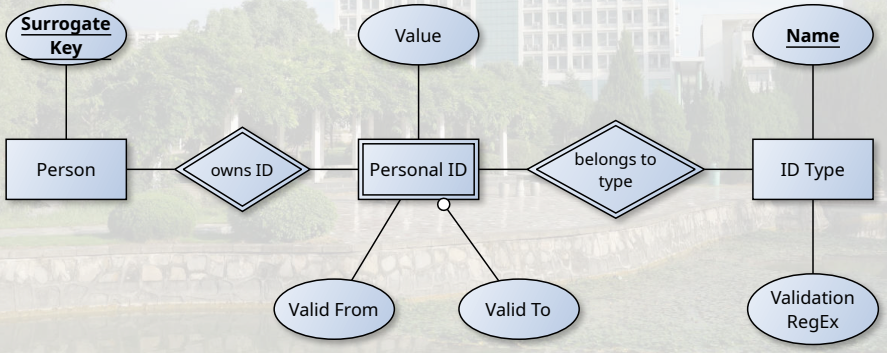
has ID als Schwache Entität

- Jede *Personal ID*-Entität wird identifiziert durch ihre Attribute *Value*, *Valid From*, und *Valid To* (die den Teilschlüssel bilden) zusammen mit den Primärschlüsseln *Surrogate Key* der *Person*-Entität und *Name* der *ID Type*-Entität.
- Jetzt kann eine Person beliebig viele Telefonnummern haben, so lange diese verschieden oder zumindest zu verschiedenen Zeiten gültig sind.



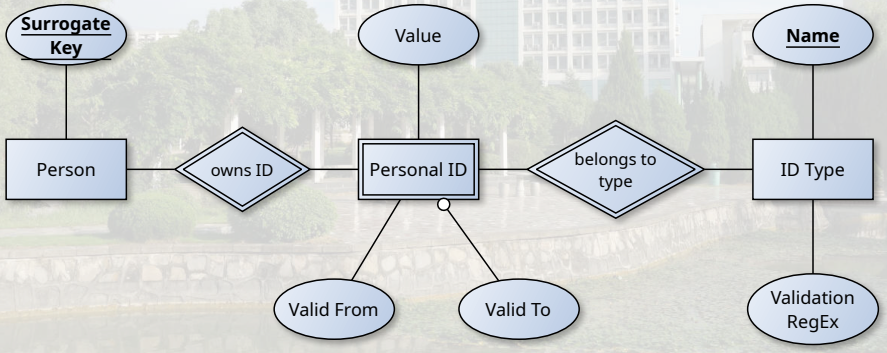
has ID als Schwache Entität

- Jetzt kann eine Person beliebig viele Telefonnummern haben, so lange diese verschieden oder zumindest zu verschiedenen Zeiten gültig sind.
- Eine Person kann verschiedene Visa haben, welche verschiedene Kennzahlen haben.



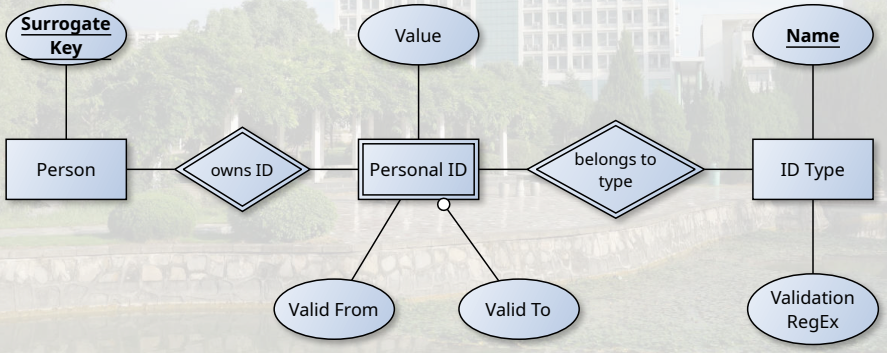
has ID als Schwache Entität

- Eine Person kann verschiedene Visa haben, welche verschiedene Kennzahlen haben.
- Nun haben wir wirklich die kommunikationsbezogenen und identifikationsbezogenen IDs einheitlich modelliert.



has ID als Schwache Entität

- Nun haben wir wirklich die kommunikationsbezogenen und identifikationsbezogenen IDs einheitlich modelliert.
- Und wir können neue ID-Formen kreieren, wann immer wir wollen.



Assoziative Entitäten

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

- In einem ERD sind assoziative Entitäten durch Rechtecke mit Rauten innen dargestellt.

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

- In einem ERD sind assoziative Entitäten durch Rechtecke mit Rauten innen dargestellt.
- Sie werden hauptsächlich verwendet, um viele-zu-vielen Beziehungen zu normalisieren und zu vereinfachen.

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

- In einem ERD sind assoziative Entitäten durch Rechtecke mit Rauten innen dargestellt.
- Sie werden hauptsächlich verwendet, um viele-zu-vielen Beziehungen zu normalisieren und zu vereinfachen.
- Sie verbinden oftmals Entitäten, die mehrere Interaktionen miteinander haben können.

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

- In einem ERD sind assoziative Entitäten durch Rechtecke mit Rauten innen dargestellt.
- Sie werden hauptsächlich verwendet, um viele-zu-vielen Beziehungen zu normalisieren und zu vereinfachen.
- Sie verbinden oftmals Entitäten, die mehrere Interaktionen miteinander haben können.
- Wir gehen nicht weiter darauf ein.

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

- In einem ERD sind assoziative Entitäten durch Rechtecke mit Rauten innen dargestellt.
- Sie werden hauptsächlich verwendet, um viele-zu-vielen Beziehungen zu normalisieren und zu vereinfachen.
- Sie verbinden oftmals Entitäten, die mehrere Interaktionen miteinander haben können.
- Wir gehen nicht weiter darauf ein.
- Ich verstehe sie eher als Möglichkeit, relationale Datenstrukturen darzustellen.

Assoziative Entitäten

Definition: Assoziative Entität

Eine *assoziative Entität* (EN: *associative entity*) ist eine Entität die Attribute und einen Primärschlüssel hat, aber auch als Beziehung dient, die Entitäten verbinden kann.

- In einem ERD sind assoziative Entitäten durch Rechtecke mit Rauten innen dargestellt.
- Sie werden hauptsächlich verwendet, um viele-zu-vielen Beziehungen zu normalisieren und zu vereinfachen.
- Sie verbinden oftmals Entitäten, die mehrere Interaktionen miteinander haben können.
- Wir gehen nicht weiter darauf ein.
- Ich verstehe sie eher als Möglichkeit, relationale Datenstrukturen darzustellen.
- Auf der konzeptuellen Ebene wollen wir das aber noch vermeiden ... das kommt dann erst auf der logischen Ebene.

Zusammenfassung

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.
- Mit schwachen Entitäten können wir nun eins-zu-vielen und viele-zu-vielen Beziehungen sauber modellieren.

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.
- Mit schwachen Entitäten können wir nun eins-zu-vielen und viele-zu-vielen Beziehungen sauber modellieren.
- Als Beispiel hatten wir die „has ID“-Beziehung.

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.
- Mit schwachen Entitäten können wir nun eins-zu-vielen und viele-zu-vielen Beziehungen sauber modellieren.
- Als Beispiel hatten wir die „has ID“-Beziehung.
- Durch das Ersetzen dieser Beziehung mit schwachen Entitäten kann eine Personen nun mehrere IDs vom selben Typ haben.

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.
- Mit schwachen Entitäten können wir nun eins-zu-vielen und viele-zu-vielen Beziehungen sauber modellieren.
- Als Beispiel hatten wir die „has ID“-Beziehung.
- Durch das Ersetzen dieser Beziehung mit schwachen Entitäten kann eine Personen nun mehrere IDs vom selben Typ haben.
- Da wir auch Emailadressen und Mobiltelefonnummern als IDs modellieren, ist das sehr wichtig.

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.
- Mit schwachen Entitäten können wir nun eins-zu-vielen und viele-zu-vielen Beziehungen sauber modellieren.
- Als Beispiel hatten wir die „has ID“-Beziehung.
- Durch das Ersetzen dieser Beziehung mit schwachen Entitäten kann eine Personen nun mehrere IDs vom selben Typ haben.
- Da wir auch Emailadressen und Mobiltelefonnummern als IDs modellieren, ist das sehr wichtig.
- Ein anderer Kandidat für schwache Entitäten wäre die Beziehung zwischen Professoren, Studenten und Modulen.

Zusammenfassung

- Und wieder sind wir einen Schritt voran gekommen.
- Mit schwachen Entitäten können wir nun eins-zu-vielen und viele-zu-vielen Beziehungen sauber modellieren.
- Als Beispiel hatten wir die „has ID“-Beziehung.
- Durch das Ersetzen dieser Beziehung mit schwachen Entitäten kann eine Personen nun mehrere IDs vom selben Typ haben.
- Da wir auch Emailadressen und Mobiltelefonnummern als IDs modellieren, ist das sehr wichtig.
- Ein anderer Kandidat für schwache Entitäten wäre die Beziehung zwischen Professoren, Studenten und Modulen.
- Damit könnten wir sauberer modellieren, dass ein Student sich in das selbe Module vom selben Professor in verschiedenen Semestern einschreibt.

谢谢您们！

Thank you!

Vielen Dank!

References I

[1] Adam Aspin und Karine Aspin. *Query Answers with MariaDB – Volume I: Introduction to SQL Queries*. Tetras Publishing, Okt. 2018. ISBN: 978-1-9996172-4-0. See also[2] (siehe S. 69, 77).

[2] Adam Aspin und Karine Aspin. *Query Answers with MariaDB – Volume II: In-Depth Querying*. Tetras Publishing, Okt. 2018. ISBN: 978-1-9996172-5-7. See also[1] (siehe S. 69, 77).

[3] Richard Barker. *Case*Method: Entity Relationship Modelling (Oracle)*. 1. Aufl. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., Jan. 1990. ISBN: 978-0-201-41696-1 (siehe S. 76).

[4] Daniel J. Barrett. *Efficient Linux at the Command Line*. Sebastopol, CA, USA: O'Reilly Media, Inc., Feb. 2022. ISBN: 978-1-0981-1340-7 (siehe S. 77, 78).

[5] Daniel Bartholomew. *Learning the MariaDB Ecosystem: Enterprise-level Features for Scalability and Availability*. New York, NY, USA: Apress Media, LLC, Okt. 2019. ISBN: 978-1-4842-5514-8 (siehe S. 77).

[6] Tim Berners-Lee. *Re: Qualifiers on Hypertext links...* Geneva, Switzerland: World Wide Web project, European Organization for Nuclear Research (CERN) und Newsgroups: alt.hypertext, 6. Aug. 1991. URL: https://www.w3.org/People/Berners-Lee/1991/08/art-6484.txt (besucht am 2025-02-05) (siehe S. 78).

[7] Alex Berson. *Client/Server Architecture*. 2. Aufl. Computer Communications Series. New York, NY, USA: McGraw-Hill, 29. März 1996. ISBN: 978-0-07-005664-0 (siehe S. 76).

[8] Silvia Botros und Jeremy Tinley. *High Performance MySQL*. 4. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., Nov. 2021. ISBN: 978-1-4920-8051-0 (siehe S. 77).

[9] Ed Bott. *Windows 11 Inside Out*. Hoboken, NJ, USA: Microsoft Press, Pearson Education, Inc., Feb. 2023. ISBN: 978-0-13-769132-6 (siehe S. 77).

[10] Ron Brash und Ganesh Naik. *Bash Cookbook*. Birmingham, England, UK: Packt Publishing Ltd, Juli 2018. ISBN: 978-1-78862-936-2 (siehe S. 76).

References II

[11] Ben Brumm. "A Guide to the Entity Relationship Diagram (ERD)". In: *Database Star*. Armadale, VIC, Australia: Elevated Online Services PTY Ltd., 30. Juli 2019–23. Dez. 2023. URL: https://www.databasestar.com/entity-relationship-diagram (besucht am 2025-03-29) (siehe S. 76).

[12] Jason Cannon. *High Availability for the LAMP Stack*. Shelter Island, NY, USA: Manning Publications, Juni 2022 (siehe S. 76, 77).

[13] Donald D. Chamberlin. "50 Years of Queries". *Communications of the ACM (CACM)* 67(8):110–121, Aug. 2024. New York, NY, USA: Association for Computing Machinery (ACM). ISSN: 0001-0782. doi:10.1145/3649887. URL: https://cacm.acm.org/research/50-years-of-queries (besucht am 2025-01-09) (siehe S. 78).

[14] Peter Pin-Shan Chen. "Entity-Relationship Modeling: Historical Events, Future Trends, and Lessons Learned". In: *Software Pioneers: Contributions to Software Engineering*. Hrsg. von Manfred Broy und Ernst Denert. Berlin/Heidelberg, Germany: Springer-Verlag GmbH Germany, Feb. 2002, S. 296–310. doi:10.1007/978-3-642-59412-0_17. URL: http://bit.csc.lsu.edu/%7Echen/pdf/Chen_Pioneers.pdf (besucht am 2025-03-06) (siehe S. 76).

[15] Peter Pin-Shan Chen. "The Entity-Relationship Model – Toward a Unified View of Data". *ACM Transactions on Database Systems (TODS)* 1(1):9–36, März 1976. New York, NY, USA: Association for Computing Machinery (ACM). ISSN: 0362-5915. doi:10.1145/320434.320440 (siehe S. 70, 76).

[16] Peter Pin-Shan Chen. "The Entity-Relationship Model: Toward a Unified View of Data". In: *1st International Conference on Very Large Data Bases (VLDB'1975)*. 22.–24. Sep. 1975, Framingham, MA, USA. Hrsg. von Douglas S. Kerr. New York, NY, USA: Association for Computing Machinery (ACM), 1975, S. 173. ISBN: 978-1-4503-3920-9. doi:10.1145/1282480.1282492. See[15] for a more comprehensive introduction. (Siehe S. 76).

[17] David Clinton und Christopher Negus. *Ubuntu Linux Bible*. 10. Aufl. Bible Series. Chichester, West Sussex, England, UK: John Wiley and Sons Ltd., 10. Nov. 2020. ISBN: 978-1-119-72233-5 (siehe S. 78).

[18] Edgar Frank „Ted" Codd. "A Relational Model of Data for Large Shared Data Banks". *Communications of the ACM (CACM)* 13(6):377–387, Juni 1970. New York, NY, USA: Association for Computing Machinery (ACM). ISSN: 0001-0782. doi:10.1145/362384.362685. URL: https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf (besucht am 2025-01-05) (siehe S. 77).

References III

[19] *Database Language SQL*. Techn. Ber. ANSI X3.135-1986. Washington, D.C., USA: American National Standards Institute (ANSI), 1986 (siehe S. 78).

[20] Matt David und Blake Barnhill. *How to Teach People SQL*. San Francisco, CA, USA: The Data School, Chart.io, Inc., 10. Dez. 2019–10. Apr. 2023. URL: https://dataschool.com/how-to-teach-people-sql (besucht am 2025-02-27) (siehe S. 78).

[21] *Database Language SQL*. International Standard ISO 9075-1987. Geneva, Switzerland: International Organization for Standardization (ISO), 1987 (siehe S. 78).

[22] Paul Deitel, Harvey Deitel und Abbey Deitel. *Internet & World Wide WebW[: How to Program*. 5. Aufl. Hoboken, NJ, USA: Pearson Education, Inc., Nov. 2011. ISBN: 978-0-13-299045-5 (siehe S. 78).

[23] Russell J.T. Dyer. *Learning MySQL and MariaDB*. Sebastopol, CA, USA: O'Reilly Media, Inc., März 2015. ISBN: 978-1-4493-6290-4 (siehe S. 77).

[24] Luca Ferrari und Enrico Pirozzi. *Learn PostgreSQL*. 2. Aufl. Birmingham, England, UK: Packt Publishing Ltd, Okt. 2023. ISBN: 978-1-83763-564-1 (siehe S. 77).

[25] Terry Halpin und Tony Morgan. *Information Modeling and Relational Databases*. 3. Aufl. Burlington, MA, USA/San Mateo, CA, USA: Morgan Kaufmann Publishers, Juli 2024. ISBN: 978-0-443-23791-1 (siehe S. 77).

[26] Jan L. Harrington. *Relational Database Design and Implementation*. 4. Aufl. Burlington, MA, USA/San Mateo, CA, USA: Morgan Kaufmann Publishers, Apr. 2016. ISBN: 978-0-12-849902-3 (siehe S. 77).

[27] Michael Hausenblas. *Learning Modern Linux*. Sebastopol, CA, USA: O'Reilly Media, Inc., Apr. 2022. ISBN: 978-1-0981-0894-6 (siehe S. 77).

[28] Matthew Helmke. *Ubuntu Linux Unleashed 2021 Edition*. 14. Aufl. Reading, MA, USA: Addison-Wesley Professional, Aug. 2020. ISBN: 978-0-13-668539-5 (siehe S. 76, 78).

[29] John Hunt. *A Beginners Guide to Python 3 Programming*. 2. Aufl. Undergraduate Topics in Computer Science (UTICS). Cham, Switzerland: Springer, 2023. ISBN: 978-3-031-35121-1. doi:10.1007/978-3-031-35122-8 (siehe S. 77).

References IV

[30] *Information Technology – Database Languages – SQL – Part 1: Framework (SQL/Framework), Part 1*. International Standard ISO/IEC 9075-1:2023(E), Sixth Edition, (ANSI X3.135). Geneva, Switzerland: International Organization for Standardization (ISO) und International Electrotechnical Commission (IEC), Juni 2023. URL: https://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_IEC_9075-1_2023_ed_6_-_id_76583_Publication_PDF_(en).zip (besucht am 2025-01-08). Consists of several parts, see https://modern-sql.com/standard for information where to obtain them. (Siehe S. 78).

[31] Shannon Kempe und Paul Williams. *A Short History of the ER Diagram and Information Modeling*. Studio City, CA, USA: Dataversity Digital LLC, 25. Sep. 2012. URL: https://www.dataversity.net/a-short-history-of-the-er-diagram-and-information-modeling (besucht am 2025-03-06) (siehe S. 76).

[32] Jay LaCroix. *Mastering Ubuntu Server*. 4. Aufl. Birmingham, England, UK: Packt Publishing Ltd, Sep. 2022. ISBN: 978-1-80323-424-3 (siehe S. 77).

[33] Kent D. Lee und Steve Hubbard. *Data Structures and Algorithms with Python*. Undergraduate Topics in Computer Science (UTICS). Cham, Switzerland: Springer, 2015. ISBN: 978-3-319-13071-2. doi:10.1007/978-3-319-13072-9 (siehe S. 77).

[34] Gloria Lotha, Aakanksha Gaur, Erik Gregersen, Swati Chopra und William L. Hosch. "Client-Server Architecture". In: *Encyclopaedia Britannica*. Hrsg. von The Editors of Encyclopaedia Britannica. Chicago, IL, USA: Encyclopædia Britannica, Inc., 3. Jan. 2025. URL: https://www.britannica.com/technology/client-server-architecture (besucht am 2025-01-20) (siehe S. 76).

[35] Mark Lutz. *Learning Python*. 6. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., März 2025. ISBN: 978-1-0981-7130-8 (siehe S. 77).

[36] *MariaDB Server Documentation*. Milpitas, CA, USA: MariaDB, 2025. URL: https://mariadb.com/kb/en/documentation (besucht am 2025-04-24) (siehe S. 77).

[37] Jim Melton und Alan R. Simon. *SQL: 1999 – Understanding Relational Language Components*. The Morgan Kaufmann Series in Data Management Systems. Burlington, MA, USA/San Mateo, CA, USA: Morgan Kaufmann Publishers, Juni 2001. ISBN: 978-1-55860-456-8 (siehe S. 78).

[38] Cameron Newham und Bill Rosenblatt. *Learning the Bash Shell – Unix Shell Programming: Covers Bash 3.0*. 3. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., 2005. ISBN: 978-0-596-00965-6 (siehe S. 76).

References V

[39] Regina O. Obe und Leo S. Hsu. *PostgreSQL: Up and Running*. 3. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., Okt. 2017. ISBN: 978-1-4919-6336-4 (siehe S. 77).

[40] Robert Orfali, Dan Harkey und Jeri Edwards. *Client/Server Survival Guide*. 3. Aufl. Chichester, West Sussex, England, UK: John Wiley and Sons Ltd., 25. Jan. 1999. ISBN: 978-0-471-31615-2 (siehe S. 76).

[41] Donnie Pinkston. "Converting E-R Diagrams to Relational Model". In: *CS101b – Introduction to Relational Databases*. Pasadena, CA, USA: California Institute of Technology (Caltech), Winter 2007. Kap. 17. URL: http://users.cms.caltech.edu/~donnie/dbcourse/intro0607/lectures/Lecture17.pdf (besucht am 2025-04-04) (siehe S. 26–31).

[42] *PostgreSQL Essentials: Leveling Up Your Data Work*. Sebastopol, CA, USA: O'Reilly Media, Inc., März 2024 (siehe S. 77).

[43] Abhishek Ratan, Eric Chou, Pradeeban Kathiravelu und Dr. M.O. Faruque Sarker. *Python Network Programming*. Birmingham, England, UK: Packt Publishing Ltd, Jan. 2019. ISBN: 978-1-78883-546-6 (siehe S. 76).

[44] Federico Razzoli. *Mastering MariaDB*. Birmingham, England, UK: Packt Publishing Ltd, Sep. 2014. ISBN: 978-1-78398-154-0 (siehe S. 77).

[45] Mike Reichardt, Michael Gundall und Hans D. Schotten. "Benchmarking the Operation Times of NoSQL and MySQL Databases for Python Clients". In: *47th Annual Conference of the IEEE Industrial Electronics Society (IECON'2021*. 13.–15. Okt. 2021, Toronto, ON, Canada. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers (IEEE), 2021, S. 1–8. ISSN: 2577-1647. ISBN: 978-1-6654-3554-3. doi:10.1109/IECON48115.2021.9589382 (siehe S. 77).

[46] Mark Richards und Neal Ford. *Fundamentals of Software Architecture: An Engineering Approach*. Sebastopol, CA, USA: O'Reilly Media, Inc., Jan. 2020. ISBN: 978-1-4920-4345-4 (siehe S. 76).

[47] Heinz Schweppe und Manuel Scholz. "Conceptual Database Design: Integrity Constraints and Modeling Patterns". In: *Einführung in die Datenbanksysteme. Datenbanken für die Bioinformatik*. Berlin, Germany: Freie Universität Berlin, Apr.–Okt. 2005. Kap. 2.3/2.4. URL: https://www.inf.fu-berlin.de/lehre/SS05/19517-V/FolienEtc/dbs05-03-ConceptualModeling2-2.pdf (besucht am 2025-03-27) (siehe S. 26–31).

References VI

[48] Yuriy Shamshin. "Conceptual Database Model. Entity Relationship Diagram (ERD)". In: *Databases*. Riga, Latvia: ISMA University of Applied Sciences, Mai 2024. Kap. 04. URL: https://dbs.academy.lv/lection/dbs_LS04EN_erd.pdf (besucht am 2025-03-29) (siehe S. 18–31, 76).

[49] Ellen Siever, Stephen Figgins, Robert Love und Arnold Robbins. *Linux in a Nutshell*. 6. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., Sep. 2009. ISBN: 978-0-596-15448-6 (siehe S. 77).

[50] John Miles Smith und Philip Yen-Tang Chang. "Optimizing the Performance of a Relational Algebra Database Interface". *Communications of the ACM (CACM)* 18(10):568–579, Okt. 1975. New York, NY, USA: Association for Computing Machinery (ACM). ISSN: 0001-0782. doi:10.1145/361020.361025 (siehe S. 77).

[51] "SQL Commands". In: *PostgreSQL Documentation*. 17.4. The PostgreSQL Global Development Group (PGDG), 20. Feb. 2025. Kap. Part VI. Reference. URL: https://www.postgresql.org/docs/17/sql-commands.html (besucht am 2025-02-25) (siehe S. 78).

[52] Ryan K. Stephens und Ronald R. Plew. *Sams Teach Yourself SQL in 21 Days*. 4. Aufl. Sams Tech Yourself. Indianapolis, IN, USA: SAMS Technical Publishing und Hoboken, NJ, USA: Pearson Education, Inc., Okt. 2002. ISBN: 978-0-672-32451-2 (siehe S. 74, 78).

[53] Ryan K. Stephens, Ronald R. Plew, Bryan Morgan und Jeff Perkins. *SQL in 21 Tagen. Die Datenbank-Abfragesprache SQL vollständig erklärt (in 14/21 Tagen)*. 6. Aufl. Burgthann, Bayern, Germany: Markt+Technik Verlag GmbH, Feb. 1998. ISBN: 978-3-8272-2020-2. Translation of[52] (siehe S. 78).

[54] Allen Taylor. *Introducing SQL and Relational Databases*. New York, NY, USA: Apress Media, LLC, Sep. 2018. ISBN: 978-1-4842-3841-7 (siehe S. 77, 78).

[55] Alkin Tezuysal und Ibrar Ahmed. *Database Design and Modeling with PostgreSQL and MySQL*. Birmingham, England, UK: Packt Publishing Ltd, Juli 2024. ISBN: 978-1-80323-347-5 (siehe S. 77).

[56] Linus Torvalds. "The Linux Edge". *Communications of the ACM (CACM)* 42(4):38–39, Apr. 1999. New York, NY, USA: Association for Computing Machinery (ACM). ISSN: 0001-0782. doi:10.1145/299157.299165 (siehe S. 77).

References VII

[57] Sander van Vugt. *Linux Fundamentals*. 2. Aufl. Hoboken, NJ, USA: Pearson IT Certification, Juni 2022. ISBN: 978-0-13-792931-3 (siehe S. 77).

[58] Thomas Weise (汤卫思). *Databases*. Hefei, Anhui, China (中国安徽省合肥市): Hefei University (合肥大学), School of Artificial Intelligence and Big Data (人工智能与大数据学院), Institute of Applied Optimization (应用优化研究所, IAO), 2025. URL: https://thomasweise.github.io/databases (besucht am 2025-01-05) (siehe S. 76, 77).

[59] Thomas Weise (汤卫思). *Programming with Python*. Hefei, Anhui, China (中国安徽省合肥市): Hefei University (合肥大学), School of Artificial Intelligence and Big Data (人工智能与大数据学院), Institute of Applied Optimization (应用优化研究所, IAO), 2024–2025. URL: https://thomasweise.github.io/programmingWithPython (besucht am 2025-01-05) (siehe S. 77).

[60] Matthew West. *Developing High Quality Data Models*. Version: 2.0, Issue: 2.1. London, England, UK: Shell International Limited und European Process Industries STEP Technical Liaison Executive (EPISTLE); Burlington, MA, USA/San Mateo, CA, USA: Morgan Kaufmann Publishers, 8. Dez. 1995–Dez. 2010. ISBN: 978-0-12-375107-2. URL: https://www.researchgate.net/publication/286610894 (besucht am 2025-03-24). Edited by Julian Fowler (siehe S. 76).

[61] *What is a Relational Database?* Armonk, NY, USA: International Business Machines Corporation (IBM), 20. Okt. 2021–12. Dez. 2024. URL: https://www.ibm.com/think/topics/relational-databases (besucht am 2025-01-05) (siehe S. 77).

[62] Ulf Michael „Monty“ Widenius, David Axmark und Uppsala, Sweden: MySQL AB. *MySQL Reference Manual – Documentation from the Source*. Sebastopol, CA, USA: O'Reilly Media, Inc., 9. Juli 2002. ISBN: 978-0-596-00265-7 (siehe S. 77).

[63] Kinza Yasar und Craig S. Mullins. *Definition: Database Management System (DBMS)*. Newton, MA, USA: TechTarget, Inc., Juni 2024. URL: https://www.techtarget.com/searchdatamanagement/definition/database-management-system (besucht am 2025-01-11) (siehe S. 76).

[64] Giorgio Zarrelli. *Mastering Bash*. Birmingham, England, UK: Packt Publishing Ltd, Juni 2017. ISBN: 978-1-78439-687-9 (siehe S. 76).

Glossary (in English) I

Bash is a the shell used under Ubuntu Linux, i.e., the program that „runs“ in the terminal and interprets your commands, allowing you to start and interact with other programs[10,38,64]. Learn more at https://www.gnu.org/software/bash.

client In a client-server architecture, the client is a device or process that requests a service from the server. It initiates the communication with the server, sends a request, and receives the response with the result of the request. Typical examples for clients are web browsers in the internet as well as clients for database management systems (DBMSes), such as psql.

client-server architecture is a system design where a central server receives requests from one or multiple clients[7,34,40,43,46]. These requests and responses are usually sent over network connections. A typical example for such a system is the World Wide Web (WWW), where web servers host websites and make them available to web browsers, the clients. Another typical example is the structure of database (DB) software, where a central server, the DBMS, offers access to the DB to the different clients. Here, the client can be some terminal software shipping with the DBMS, such as psql, or the different applications that access the DBs.

DB A *database* is an organized collection of structured information or data, typically stored electronically in a computer system. Databases are discussed in our book *Databases*[58].

DBMS A *database management system* is the software layer located between the user or application and the DB. The DBMS allows the user/application to create, read, write, update, delete, and otherwise manipulate the data in the DB[63].

DBS A *database system* is the combination of a DB and a the corresponding DBMS, i.e., basically, an installation of a DBMS on a computer together with one or multiple DBs. DBS = DB + DBMS.

ERD Entity relationship diagrams show the relationships between objects, e.g., between the tables in a DB and how they reference each other[3,11,14–16,31,48,60]

IT information technology

LAMP Stack A system setup for web applications: Linux, Apache (a web server), MySQL, and the server-side scripting language PHP[12,28].

Glossary (in English) II

Linux is the leading open source operating system, i.e., a free alternative for Microsoft Windows[4,27,49,56,57]. We recommend using it for this course, for software development, and for research. Learn more at https://www.linux.org. Its variant Ubuntu is particularly easy to use and install.

MariaDB An open source relational database management system that has forked off from MySQL[1,2,5,23,36,44]. See https://mariadb.org for more information.

Microsoft Windows is a commercial proprietary operating system[9]. It is widely spread, but we recommend using a Linux variant such as Ubuntu for software development and for our course. Learn more at https://www.microsoft.com/windows.

MySQL An open source relational database management system[8,23,45,55,62]. MySQL is famous for its use in the LAMP Stack. See https://www.mysql.com for more information.

PostgreSQL An open source object-relational DBMS[24,39,42,55]. See https://postgresql.org for more information.

psql is the client program used to access the PostgreSQL DBMS server.

Python The Python programming language[29,33,35,59], i.e., what you will learn about in our book[59]. Learn more at https://python.org.

relational database A relational DB is a database that organizes data into rows (tuples, records) and columns (attributes), which collectively form tables (relations) where the data points are related to each other[18,25,26,50,54,58,61].

server In a client-server architecture, the server is a process that fulfills the requests of the clients. It usually waits for incoming communication carring the requests from the clients. For each request, it takes the necessary actions, performs the required computations, and then sends a response with the result of the request. Typical examples for servers are web servers[12] in the internet as well as DBMSes. It is also common to refer to the computer running the server processes as server as well, i.e., to call it the „server computer“[32].

Glossary (in English) III

SQL The *Structured Query Language* is basically a programming language for querying and manipulating relational databases[13,19–21,30,37,51–54]. It is understood by many DBMSes. You find the Structured Query Language (SQL) commands supported by PostgreSQL in the reference[51].

terminal A terminal is a text-based window where you can enter commands and execute them[4,17]. Knowing what a terminal is and how to use it is very essential in any programming- or system administration-related task. If you want to open a terminal under Microsoft Windows, you can Druck auf ⊞+R, dann Schreiben von `cmd`, dann Druck auf ↵. Under Ubuntu Linux, Ctrl+Alt+T opens a terminal, which then runs a Bash shell inside.

Ubuntu is a variant of the open source operating system Linux[17,28]. We recommend that you use this operating system to follow this class, for software development, and for research. Learn more at https://ubuntu.com. If you are in China, you can download it from https://mirrors.ustc.edu.cn/ubuntu-releases.

WWW World Wide Web[6,22]