合肥大學
HEFEI UNIVERSITY

Datenbanken

29. Konzeptuelles Schema: Beziehungen

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: 2025-10-28

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

- Wenn wir nur einen Entitätstyp in unserer Datenbank hätten, dann würde es keinen Sinn ergeben, eine Datenbank zu benutzen.

Einleitung

- Wenn wir nur einen Entitätstyp in unserer Datenbank hätten, dann würde es keinen Sinn ergeben, eine Datenbank zu benutzen.
- Dann würden wir eher einfache Dokumente wir CSV[28] oder XML[2,8,17] Dateien verwenden.

Einleitung

- Wenn wir nur einen Entitätstyp in unserer Datenbank hätten, dann würde es keinen Sinn ergeben, eine Datenbank zu benutzen.
- Dann würden wir eher einfache Dokumente wir CSV[28] oder XML[2,8,17] Dateien verwenden.
- Unserer Lehrmanagementplattform hat aber sicher verschiedene Entitätstypen.

Einleitung

- Wenn wir nur einen Entitätstyp in unserer Datenbank hätten, dann würde es keinen Sinn ergeben, eine Datenbank zu benutzen.
- Dann würden wir eher einfache Dokumente wir CSV[28] oder XML[2,8,17] Dateien verwenden.
- Unserer Lehrmanagementplattform hat aber sicher verschiedene Entitätstypen.
- Wenn diese Entitätstypen nichts miteinander zu tun hätten, wie z. B. „Studenten“, „Wetter“ und „Produkt“, dann wäre es immer noch besser, diese einfach in mehreren Dateien zu speichern.

Einleitung

- Wenn wir nur einen Entitätstyp in unserer Datenbank hätten, dann würde es keinen Sinn ergeben, eine Datenbank zu benutzen.
- Dann würden wir eher einfache Dokumente wir CSV[28] oder XML[2,8,17] Dateien verwenden.
- Unserer Lehrmanagementplattform hat aber sicher verschiedene Entitätstypen.
- Wenn diese Entitätstypen nichts miteinander zu tun hätten, wie z. B. „Studenten“, „Wetter“ und „Produkt“, dann wäre es immer noch besser, diese einfach in mehreren Dateien zu speichern.
- Aber unsere Entitätstypen werden *sehr viel* in Beziehung zueinander stehen.

Einleitung

- Wenn wir nur einen Entitätstyp in unserer Datenbank hätten, dann würde es keinen Sinn ergeben, eine Datenbank zu benutzen.
- Dann würden wir eher einfache Dokumente wir CSV[28] oder XML[2,8,17] Dateien verwenden.
- Unserer Lehrmanagementplattform hat aber sicher verschiedene Entitätstypen.
- Wenn diese Entitätstypen nichts miteinander zu tun hätten, wie z. B. „Studenten“, „Wetter“ und „Produkt“, dann wäre es immer noch besser, diese einfach in mehreren Dateien zu speichern.
- Aber unsere Entitätstypen werden *sehr viel* in Beziehung zueinander stehen.
- Wir modellieren wir also diese Beziehungen?

Beziehungen

Beziehungen und Beziehungstypen

Definition: Beziehung

Eine Beziehung (EN: *relationship (instance)*) ist eine Verbindung von zwei oder mehr Entitäten.

Beziehungen und Beziehungstypen

Definition: Beziehung

Eine Beziehung (EN: *relationship (instance)*) ist eine Verbindung von zwei oder mehr Entitäten.

- Ein Beispiel einer Beziehung ist *Mr. Bibbo schreibt sich in das Modul Programming with Python ein.*

Beziehungen und Beziehungstypen

Definition: Beziehung

Eine Beziehung (EN: *relationship (instance)*) ist eine Verbindung von zwei oder mehr Entitäten.

- Ein Beispiel einer Beziehung ist *Mr. Bibbo schreibt sich in das Modul Programming with Python ein.*

Definition: Beziehungstyp

Ein Beziehungstyp (EN: *relationship type*) ist die Menge aller möglichen Beziehungen zwischen zwei oder mehr Entitätsmengen.

Beziehungen und Beziehungstypen

Definition: Beziehung

Eine Beziehung (EN: *relationship (instance)*) ist eine Verbindung von zwei oder mehr Entitäten.

- Ein Beispiel einer Beziehung ist *Mr. Bibbo schreibt sich in das Modul Programming with Python ein.*

Definition: Beziehungstyp

Ein Beziehungstyp (EN: *relationship type*) ist die Menge aller möglichen Beziehungen zwischen zwei oder mehr Entitätsmengen.

Beziehungen und Beziehungstypen

Definition: Beziehung

Eine Beziehung (EN: *relationship (instance)*) ist eine Verbindung von zwei oder mehr Entitäten.

- Ein Beispiel einer Beziehung ist *Mr. Bibbo schreibt sich in das Modul Programming with Python ein.*

Definition: Beziehungstyp

Ein Beziehungstyp (EN: *relationship type*) ist die Menge aller möglichen Beziehungen zwischen zwei oder mehr Entitätsmengen.

- Der *schreibt-sich-ein* Beziehungstyp könnte zwischen dem Entitätstyp *Student* und dem Entitätstyp *Module* existieren.

Beziehungen und Beziehungstypen

Definition: Beziehung

Eine Beziehung (EN: *relationship (instance)*) ist eine Verbindung von zwei oder mehr Entitäten.

- Ein Beispiel einer Beziehung ist *Mr. Bibbo schreibt sich in das Modul Programming with Python ein.*

Definition: Beziehungstyp

Ein Beziehungstyp (EN: *relationship type*) ist die Menge aller möglichen Beziehungen zwischen zwei oder mehr Entitätsmengen.

- Der *schreibt-sich-ein* Beziehungstyp könnte zwischen dem Entitätstyp *Student* und dem Entitätstyp *Module* existieren.
- Transitive Verben[10] in den Anforderungen stehen oft für Beziehungstypen[4].

Grad einer Beziehung

Definition: Grad einer Beziehung

Der Grad (EN: *degree*) einer Beziehung (oder eines Beziehungstyps) entspricht der Anzahl der daran teilnehmenden Entitäten (oder Entitätstypen).

Grad einer Beziehung

Definition: Grad einer Beziehung

Der Grad (EN: *degree*) einer Beziehung (oder eines Beziehungstyps) entspricht der Anzahl der daran teilnehmenden Entitäten (oder Entitätstypen).

- Es kann binäre Beziehungen gegebn, also solche, wo zwei Entitäten teilnehmen.

Grad einer Beziehung

Definition: Grad einer Beziehung

Der Grad (EN: *degree*) einer Beziehung (oder eines Beziehungstyps) entspricht der Anzahl der daran teilnehmenden Entitäten (oder Entitätstypen).

- Es kann binäre Beziehungen gegebn, also solche, wo zwei Entitäten teilnehmen. Z. B. ist die die Student-Modul Beziehung binär, also *Student schreibt sich in Modul ein*.

Grad einer Beziehung

Definition: Grad einer Beziehung

Der Grad (EN: *degree*) einer Beziehung (oder eines Beziehungstyps) entspricht der Anzahl der daran teilnehmenden Entitäten (oder Entitätstypen).

- Es kann binäre Beziehungen gegebn, also solche, wo zwei Entitäten teilnehmen. Z. B. ist die die Student-Modul Beziehung binär, also *Student schreibt sich in Modul ein*.
- Es kann auch ternäre Beziehungen geben, wo drei Entitäten teilnehmen.

Grad einer Beziehung

Definition: Grad einer Beziehung

Der Grad (EN: *degree*) einer Beziehung (oder eines Beziehungstyps) entspricht der Anzahl der daran teilnehmenden Entitäten (oder Entitätstypen).

- Es kann binäre Beziehungen gegebn, also solche, wo zwei Entitäten teilnehmen. Z. B. ist die die Student-Modul Beziehung binär, also *Student schreibt sich in Modul ein*.
- Es kann auch ternäre Beziehungen geben, wo drei Entitäten teilnehmen. Z. B. *Student schreibt sich in Modul ein das von Professor gelehrt wird*.
- Und so weiter...

Rollen in einer Beziehung

Definition: Rollen in einer Beziehung

Jede Entität, die an einer Beziehung teilnimmt, kann eine Rolle haben (EN: *role*), welche die Art definiert, auf die die Entitäten in der Beziehung teilnimmt.

Rollen in einer Beziehung

Definition: Rollen in einer Beziehung

Jede Entität, die an einer Beziehung teilnimmt, kann eine Rolle haben (EN: *role*), welche die Art definiert, auf die die Entitäten in der Beziehung teilnimmt.

- Stellen wir uns nochmal die ternäre Beziehung *Student schreibt sich in Modul ein das von Professor gelehrt wird*.

Rollen in einer Beziehung

Definition: Rollen in einer Beziehung

Jede Entität, die an einer Beziehung teilnimmt, kann eine Rolle haben (EN: *role*), welche die Art definiert, auf die die Entitäten in der Beziehung teilnimmt.

- Stellen wir uns nochmal die ternäre Beziehung *Student schreibt sich in Modul ein das von Professor gelehrt wird*.
- Dann hat der Student die Rolle *schreibt sich ein*.

Rollen in einer Beziehung

Definition: Rollen in einer Beziehung

Jede Entität, die an einer Beziehung teilnimmt, kann eine Rolle haben (EN: *role*), welche die Art definiert, auf die die Entitäten in der Beziehung teilnimmt.

- Stellen wir uns nochmal die ternäre Beziehung *Student schreibt sich in Modul ein das von Professor gelehrt wird*.
- Dann hat der Student die Rolle *schreibt sich ein*.
- Der Professor hat die Rolle *lehrt*.

Beziehungsattribute

Definition: Beziehungsattribute

Ein Beziehungstyp kann Attribute haben, die die Eigenschaften der Beziehungen, die er definiert, beschreiben.

Beziehungsattribute

Definition: Beziehungsattribute

Ein Beziehungstyp kann Attribute haben, die die Eigenschaften der Beziehungen, die er definiert, beschreiben.

- Wir könnten z. B. schreiben *Herr Bebbo schreibt sich in das Modul Databases im Sommersemester 2025 ein.*

Beziehungsattribute

Definition: Beziehungsattribute

Ein Beziehungstyp kann Attribute haben, die die Eigenschaften der Beziehungen, die er definiert, beschreiben.

- Wir könnten z. B. schreiben *Herr Bebbo schreibt sich in das Modul Databases im Sommersemester 2025 ein.*
- Das Attribut *Semester* der Beziehung ist in diesem Kontext sinnvoll.

Beziehungsattribute

Definition: Beziehungsattribute

Ein Beziehungstyp kann Attribute haben, die die Eigenschaften der Beziehungen, die er definiert, beschreiben.

- Wir könnten z. B. schreiben *Herr Bebbo schreibt sich in das Modul Databases im Sommersemester 2025 ein.*
- Das Attribut *Semester* der Beziehung ist in diesem Kontext sinnvoll.
- Es gehört nämlich weder zum Student *Herr Bebbo* noch zum Modul *Databases*.

Beziehungsattribute

Definition: Beziehungsattribute

Ein Beziehungstyp kann Attribute haben, die die Eigenschaften der Beziehungen, die er definiert, beschreiben.

- Wir könnten z. B. schreiben *Herr Bebbo schreibt sich in das Modul Databases im Sommersemester 2025 ein.*
- Das Attribut *Semester* der Beziehung ist in diesem Kontext sinnvoll.
- Es gehört nämlich weder zum Student *Herr Bebbo* noch zum Modul *Databases*.
- Anders als Entitäten haben Beziehungen aber keine Schlüsselattribute.

Beziehungsattribute

Definition: Beziehungsattribute

Ein Beziehungstyp kann Attribute haben, die die Eigenschaften der Beziehungen, die er definiert, beschreiben.

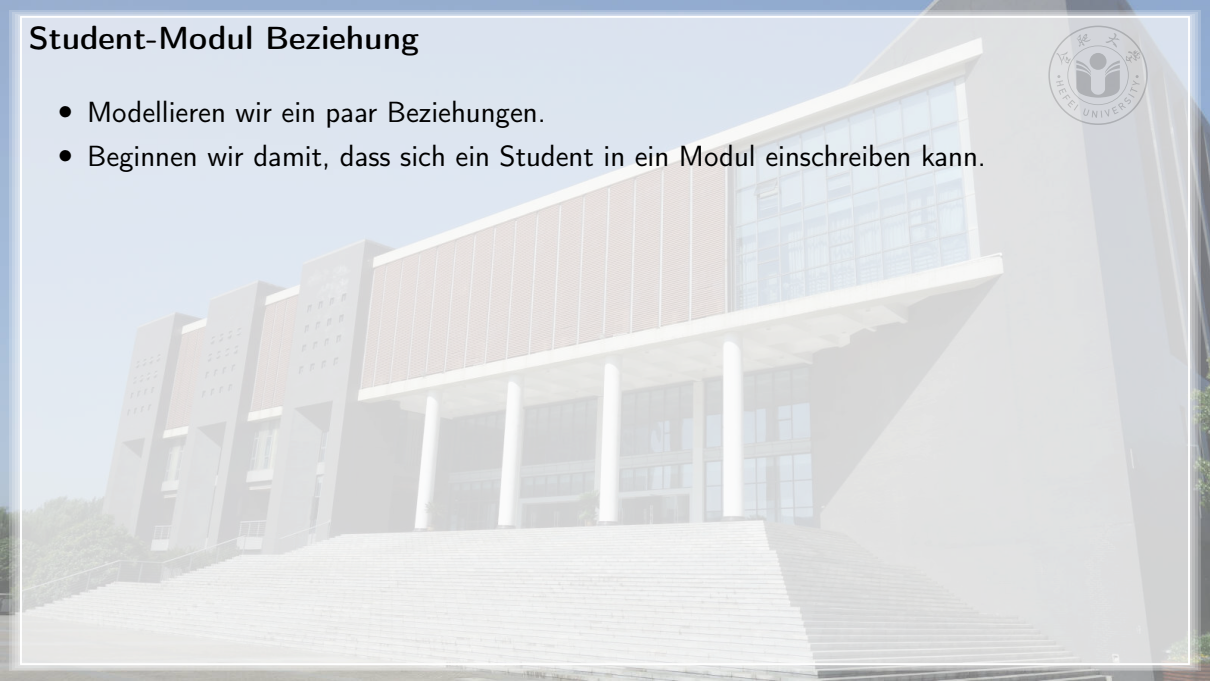
- Wir könnten z. B. schreiben *Herr Bebbo schreibt sich in das Modul Databases im Sommersemester 2025 ein.*
- Das Attribut *Semester* der Beziehung ist in diesem Kontext sinnvoll.
- Es gehört nämlich weder zum Student *Herr Bebbo* noch zum Modul *Databases*.
- Anders als Entitäten haben Beziehungen aber keine Schlüsselattribute.
- Die Beziehungen werden durch die Primärschlüssel der teilnehmenden Entitäten definiert[12].

Beispiele

Student-Modul Beziehung

- Modellieren wir ein paar Beziehungen.

Student-Modul Beziehung



- Modellieren wir ein paar Beziehungen.
- Beginnen wir damit, dass sich ein Student in ein Modul einschreiben kann.

Student-Modul Beziehung

- Modellieren wir ein paar Beziehungen.
- Beginnen wir damit, dass sich ein Student in ein Modul einschreiben kann.
- In einem ERD werden Beziehungen als Rauten (EN: *diamonds*) gemalt, die mit den in-Beziehung-stehenden Entitätstypen verbunden sind.

Student-Modul Beziehung

- Modellieren wir ein paar Beziehungen.
- Beginnen wir damit, dass sich ein Student in ein Modul einschreiben kann.
- In einem ERD werden Beziehungen als Rauten (EN: *diamonds*) gemalt, die mit den in-Beziehung-stehenden Entitätstypen verbunden sind.

Student-Modul Beziehung

- Modellieren wir ein paar Beziehungen.
- Beginnen wir damit, dass sich ein Student in ein Modul einschreiben kann.
- In einem ERD werden Beziehungen als Rauten (EN: *diamonds*) gemalt, die mit den in-Beziehung-stehenden Entitätstypen verbunden sind.
- In dem ERD hier sehen, dass der Entitätstyp *Student* mit dem Entitätstyp *Modul* über die Beziehung *enrolls into* (also *schreibt sich ein*) verbunden ist.

Student-Modul Beziehung

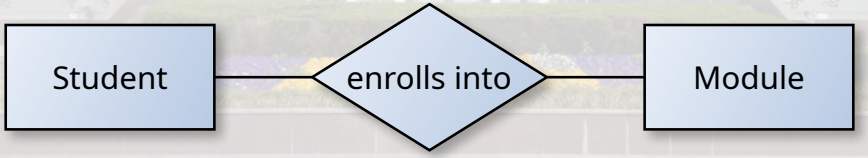
- Modellieren wir ein paar Beziehungen.
- Beginnen wir damit, dass sich ein Student in ein Modul einschreiben kann.
- In einem ERD werden Beziehungen als Rauten (EN: *diamonds*) gemalt, die mit den in-Beziehung-stehenden Entitätstypen verbunden sind.
- In dem ERD hier sehen, dass der Entitätstyp *Student* mit dem Entitätstyp *Modul* über die Beziehung *enrolls into* (also *schreibt sich ein*) verbunden ist.
- Das ist eine binäre Beziehung, weil immer zwei Entitäten teilnehmen.

Student-Modul Beziehung

- Modellieren wir ein paar Beziehungen.
- Beginnen wir damit, dass sich ein Student in ein Modul einschreiben kann.
- In einem ERD werden Beziehungen als Rauten (EN: *diamonds*) gemalt, die mit den in-Beziehung-stehenden Entitätstypen verbunden sind.
- In dem ERD hier sehen, dass der Entitätstyp *Student* mit dem Entitätstyp *Modul* über die Beziehung *enrolls into* (also *schreibt sich ein*) verbunden ist.
- Das ist eine binäre Beziehung, weil immer zwei Entitäten teilnehmen.
- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.

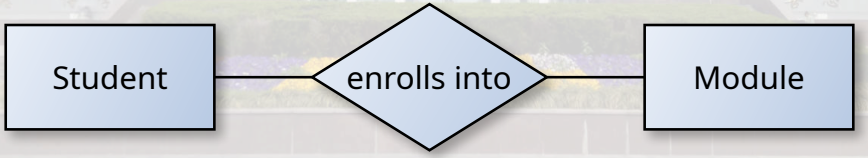
Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.



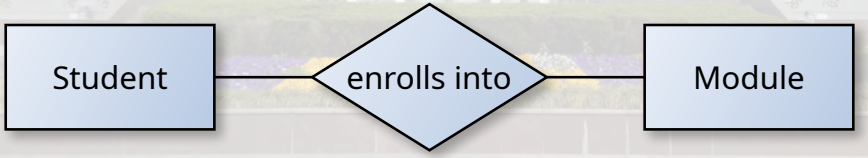
Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.
- Module werden von Professoren unterrichtet.



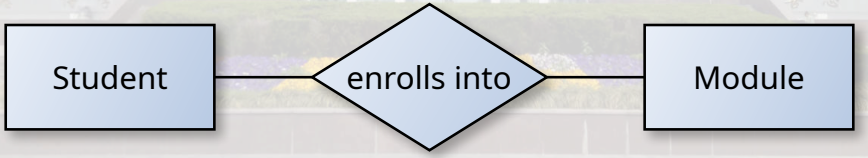
Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.
- Module werden von Professoren unterrichtet.
- Das taucht hier nicht auf.



Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.
- Module werden von Professoren unterrichtet.
- Das taucht hier nicht auf Das Diagramm sagt gar nichts über die Beziehung von Student, Professor, und Modul aus.



Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.
- Module werden von Professoren unterrichtet.
- Das taucht hier nicht auf Das Diagramm sagt gar nichts über die Beziehung von Student, Professor, und Modul aus.
- Wir fügen die Beziehung „Professor teaches Module“ hinzu.

Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.
- Module werden von Professoren unterrichtet.
- Das taucht hier nicht auf Das Diagramm sagt gar nichts über die Beziehung von Student, Professor, und Modul aus.
- Wir fügen die Beziehung „Professor teaches Module“ hinzu.
- Das löst das Problem leider überhaupt nicht.

Student-Modul Professor (1)

- Entitäten des Types *Student* können sich also in Entitäten des Types *Modul* einschreiben.
- Module werden von Professoren unterrichtet.
- Das taucht hier nicht auf Das Diagramm sagt gar nichts über die Beziehung von Student, Professor, und Modul aus.
- Wir fügen die Beziehung „Professor teaches Module“ hinzu.
- Das löst das Problem leider überhaupt nicht.
- Wir können darstellen, dass sich eine Studentin in ein bestimmtes Modul einschreibt.

Student-Modul Professor (1)

- Module werden von Professoren unterrichtet.
- Das taucht hier nicht auf Das Diagramm sagt gar nichts über die Beziehung von Student, Professor, und Modul aus.
- Wir fügen die Beziehung „Professor teaches Module“ hinzu.
- Das löst das Problem leider überhaupt nicht.
- Wir können darstellen, dass sich eine Studentin in ein bestimmtes Modul einschreibt.
- Viele Studentinnen können sich in das selbe Modul einschreiben.

Student-Modul Professor (1)

- Das taucht hier nicht auf Das Diagramm sagt gar nichts über die Beziehung von Student, Professor, und Modul aus.
- Wir fügen die Beziehung „Professor teaches Module“ hinzu.
- Das löst das Problem leider überhaupt nicht.
- Wir können darstellen, dass sich eine Studentin in ein bestimmtes Modul einschreibt.
- Viele Studentinnen können sich in das selbe Modul einschreiben.
- Wir können auch darstellen, dass ein Professor dieses Modul lehrt.

Student-Modul Professor (1)

- Wir fügen die Beziehung „Professor teaches Module“ hinzu.
- Das löst das Problem leider überhaupt nicht.
- Wir können darstellen, dass sich eine Studentin in ein bestimmtes Modul einschreibt.
- Viele Studentinnen können sich in das selbe Modul einschreiben.
- Wir können auch darstellen, dass ein Professor dieses Modul lehrt.
- Mehrere Professoren können ein Modul lehren.

Student-Modul Professor (1)

- Das löst das Problem leider überhaupt nicht.
- Wir können darstellen, dass sich eine Studentin in ein bestimmtes Modul einschreibt.
- Viele Studentinnen können sich in das selbe Modul einschreiben.
- Wir können auch darstellen, dass ein Professor dieses Modul lehrt.
- Mehrere Professoren können ein Modul lehren.
- Vielleicht unterrichtet dieses Jahr Professor Weise (汤卫思) das Modul *Databases* – nächstes Jahr macht das vielleicht Professor 李.

Student-Modul Professor (1)

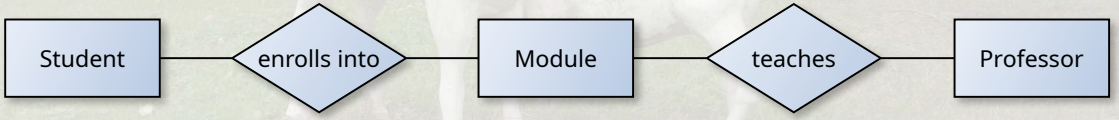
- Wir können darstellen, dass sich eine Studentin in ein bestimmtes Modul einschreibt.
- Viele Studentinnen können sich in das selbe Modul einschreiben.
- Wir können auch darstellen, dass ein Professor dieses Modul lehrt.
- Mehrere Professoren können ein Modul lehren.
- Vielleicht unterrichtet dieses Jahr Professor Weise (汤卫思) das Modul *Databases* – nächstes Jahr macht das vielleicht Professor 李.
- Wir können nicht darstellen, dass eine Studentin an einem Modul teilnimmt, das ein bestimmer Professor lehrt.

Student-Modul Professor (1)

- Viele Studentinnen können sich in das selbe Modul einschreiben.
- Wir können auch darstellen, dass ein Professor dieses Modul lehrt.
- Mehrere Professoren können ein Modul lehren.
- Vielleicht unterrichtet dieses Jahr Professor Weise (汤卫思) das Modul *Databases* – nächstes Jahr macht das vielleicht Professor 李.
- Wir können nicht darstellen, dass eine Studentin an einem Modul teilnimmt, das ein bestimmer Professor lehrt.
- Denn die beiden binären Beziehungen, die wir gemalt haben, sind unabhängig von einander.

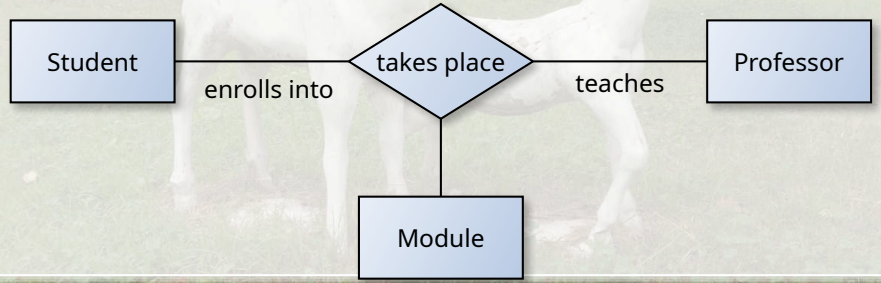
Student-Modul Professor (2)

- Das können wir reparieren, in dem wir eine ternäre Beziehung benutzen.



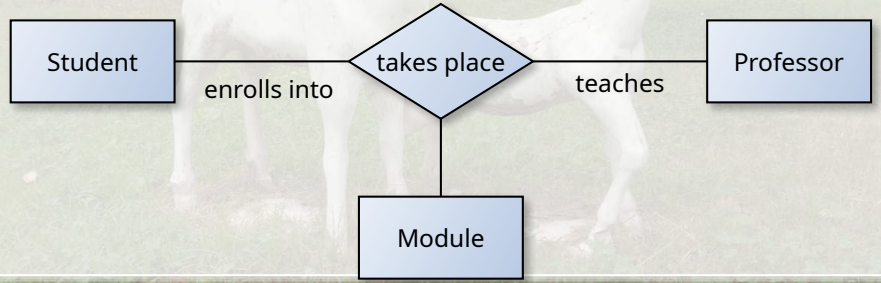
Student-Modul Professor (2)

- Das können wir reparieren, in dem wir eine ternäre Beziehung benutzen.



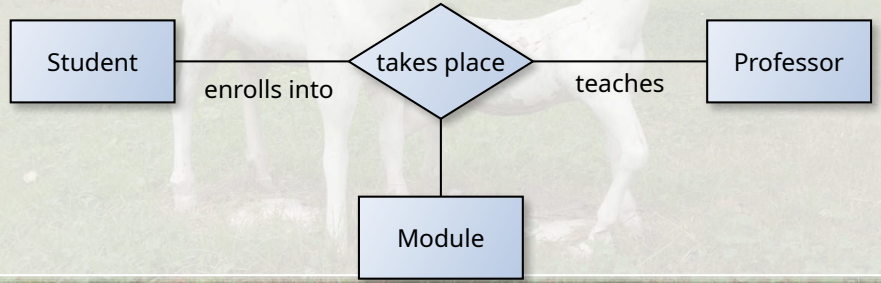
Student-Modul Professor (2)

- Das können wir reparieren, in dem wir eine ternäre Beziehung benutzen.
- In unserem ERD nehmen nun drei Entitätstypen an der Beziehung teil.



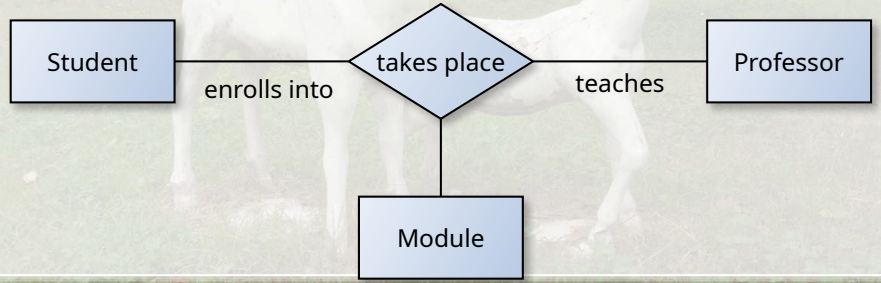
Student-Modul Professor (2)

- Das können wir reparieren, in dem wir eine ternäre Beziehung benutzen.
- In unserem ERD nehmen nun drei Entitätstypen an der Beziehung teil.
- Die Professorin lehrt das Modul, die Studentin schreibt sich in das Modul ein das die Professorin lehrt.



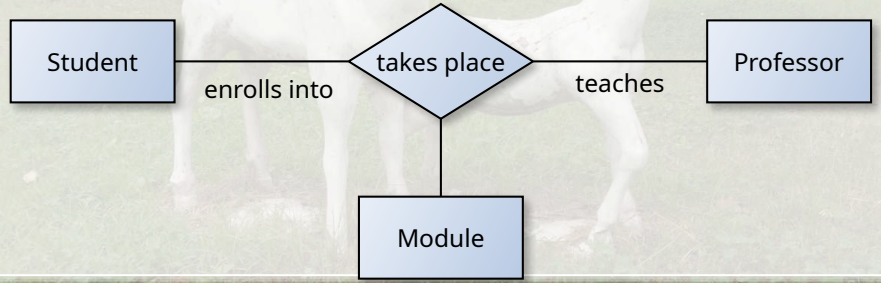
Student-Modul Professor (2)

- Das können wir reparieren, in dem wir eine ternäre Beziehung benutzen.
- In unserem ERD nehmen nun drei Entitätstypen an der Beziehung teil.
- Die Professorin lehrt das Modul, die Studentin schreibt sich in das Modul ein das die Professorin lehrt.
- Dieses Modell ist besser.



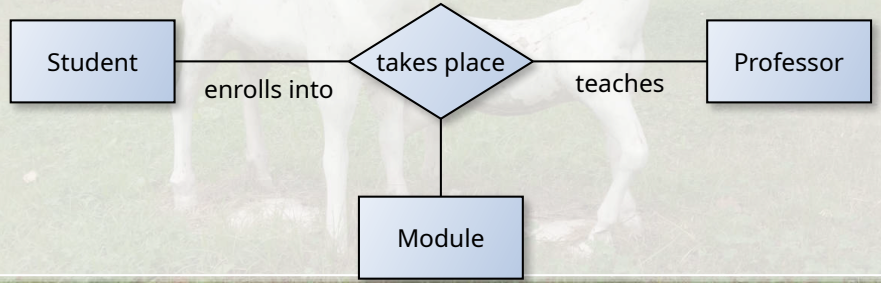
Student-Modul Professor (2)

- Das können wir reparieren, in dem wir eine ternäre Beziehung benutzen.
- In unserem ERD nehmen nun drei Entitätstypen an der Beziehung teil.
- Die Professorin lehrt das Modul, die Studentin schreibt sich in das Modul ein das die Professorin lehrt.
- Dieses Modell ist besser.
- Wir sagen aber noch nicht, wann das Modul stattfindet.



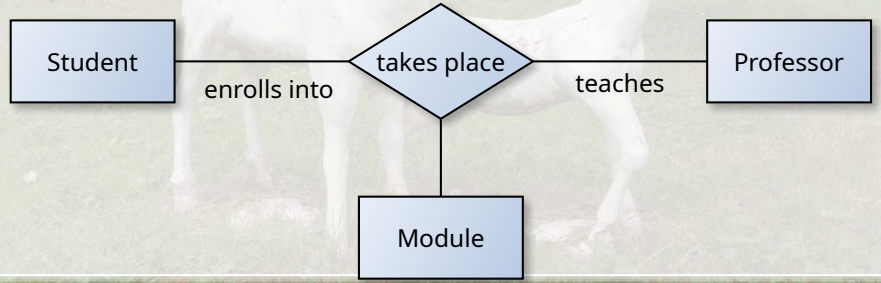
Student-Modul Professor (2)

- In unserem ERD nehmen nun drei Entitätstypen an der Beziehung teil.
- Die Professorin lehrt das Modul, die Studentin schreibt sich in das Modul ein das die Professorin lehrt.
- Dieses Modell ist besser.
- Wir sagen aber noch nicht, wann das Modul stattfindet.
- Wir haben ja gesagt, dass eine Beziehung durch die Primärschlüssel der teilnehmen Entitäten bestimmt wird.



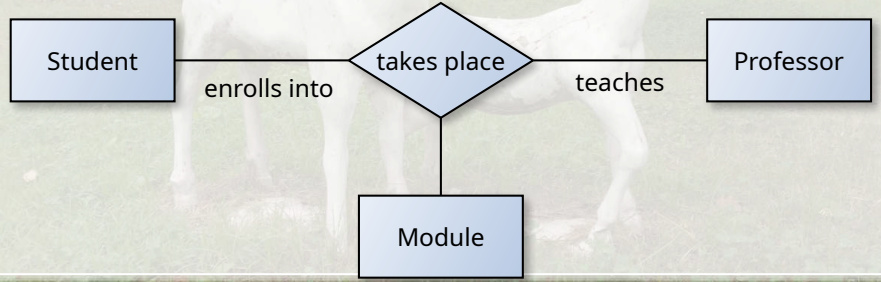
Student-Modul Professor (2)

- Dieses Modell ist besser.
- Wir sagen aber noch nicht, wann das Modul stattfindet.
- Wir haben ja gesagt, dass eine Beziehung durch die Primärschlüssel der teilnehmen Entitäten bestimmt wird.
- Was passiert, wenn ein Student am Modul teilnimmt, aber leider das Examen nicht besteht und das Modul wiederholen muss?



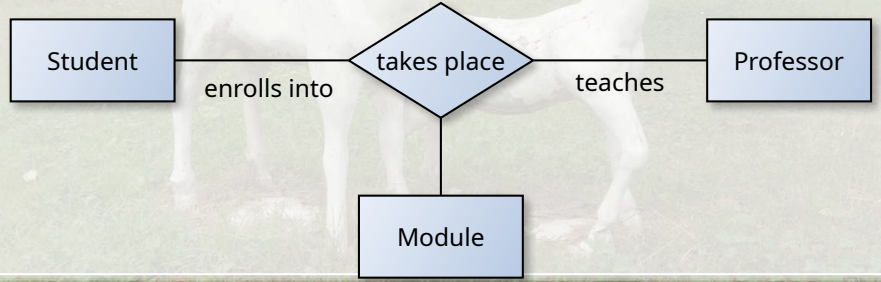
Student-Modul Professor (2)

- Dieses Modell ist besser.
- Wir sagen aber noch nicht, wann das Modul stattfindet.
- Wir haben ja gesagt, dass eine Beziehung durch die Primärschlüssel der teilnehmen Entitäten bestimmt wird.
- Was passiert, wenn ein Student am Modul teilnimmt, aber leider das Examen nicht besteht und das Modul wiederholen muss?
- Was wenn die Professorin es nochmal unterrichtet?



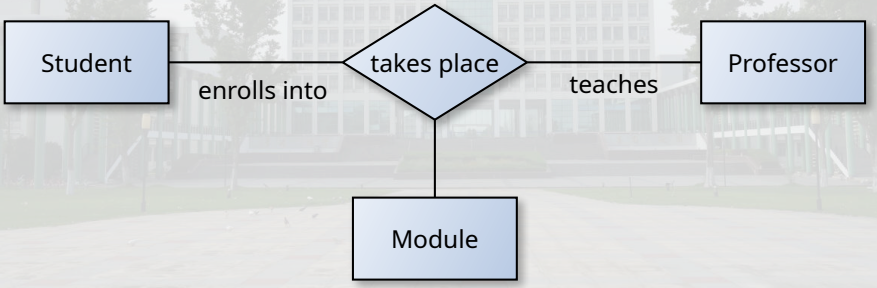
Student-Modul Professor (2)

- Wir sagen aber noch nicht, wann das Modul stattfindet.
- Wir haben ja gesagt, dass eine Beziehung durch die Primärschlüssel der teilnehmen Entitäten bestimmt wird.
- Was passiert, wenn ein Student am Modul teilnimmt, aber leider das Examen nicht besteht und das Modul wiederholen muss?
- Was wenn die Professorin es nochmal unterrichtet?
- Dann haben wir zwei Beziehungen, wo alle Entitäten gleich sind.



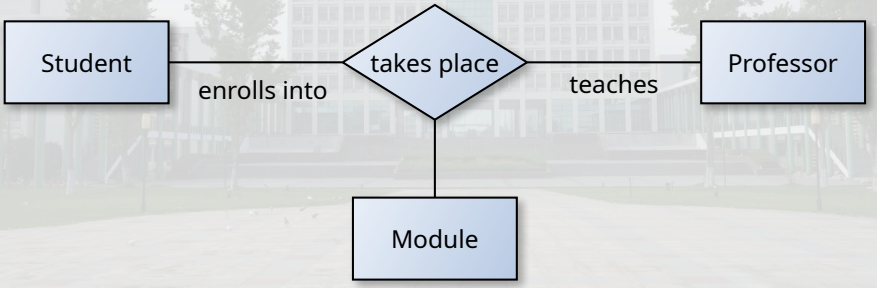
Student-Modul Professor (3)

- Wie können wir das Problem lösen?



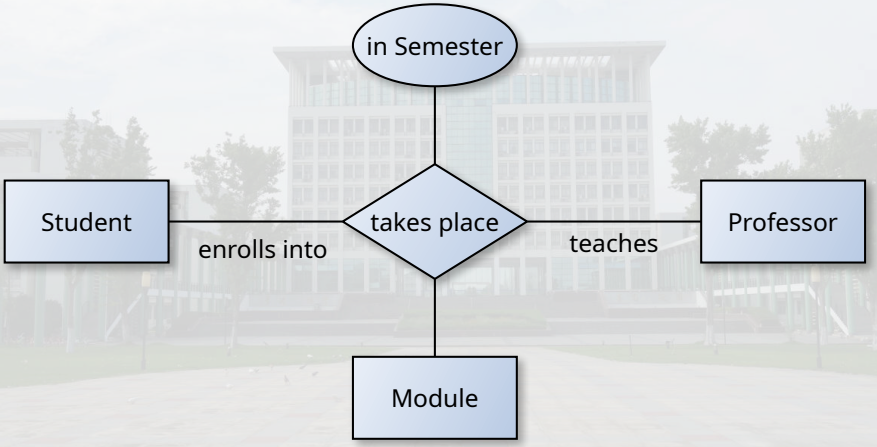
Student-Modul Professor (3)

- Wie können wir das Problem lösen?
- In dem wir der Beziehung ein Attribut *in Semester* geben.



Student-Modul Professor (3)

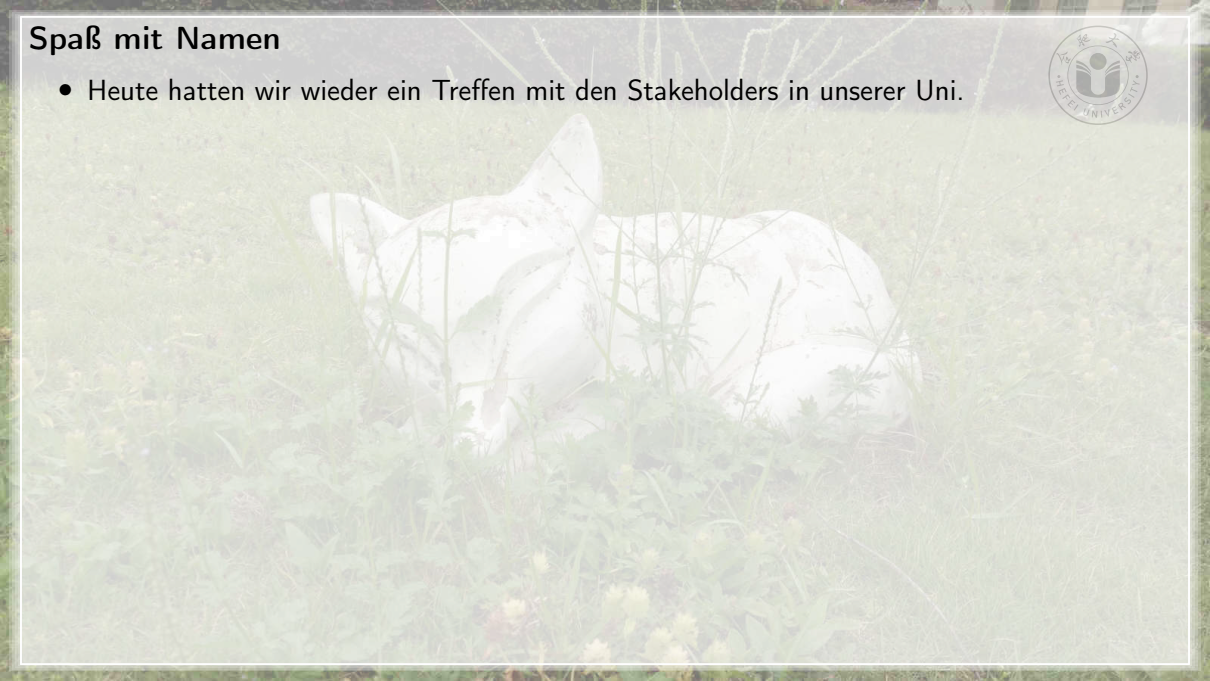
- Wie können wir das Problem lösen?
- In dem wir der Beziehung ein Attribut *in Semester* geben.



Modellieren von Personen, Studenten, und Mitarbeitern

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.



Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus . . . aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus . . . aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus . . . aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus . . . aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.

Spaß mit Namen

- Heute hatten wir wieder ein Treffen mit den Stakeholders in unserer Uni.
- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus . . . aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?

Spaß mit Namen

- Wir haben ihnen das ERD mit dem Entitätstyp für Studenten gezeigt.
- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen *„Originalname“*?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.

Spaß mit Namen

- Es sieht gut aus ... aber es gibt Probleme.
- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.

Spaß mit Namen

- Es gibt Probleme mit Namen.
- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.

Spaß mit Namen

- Leute können nähmlich mehrere Namane haben.
- Ausländische Austauschstudenten haben z. B. ihren originalen Namen.
- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.

Spaß mit Namen

- Sie können aber auch zusätzlich einen chinesischen Namen haben.
- Eine *Frau Elizabeth Prudence McDouglas* könnte z. B. auch als 邓小花女士 in unserer Uni bekannt sein.
- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen *„Originalname“*?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.

Spaß mit Namen

- Natürlich würden wir in offiziellen Dokumenten nur den Originalname verwenden.
- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.
- Egal. Es kann viele Gründe geben, warum eine Person entweder mehrere Namen hat oder sich der Name einer Person ändert.

Spaß mit Namen

- In Uni-internen Dokumenten oder auf Platzkarten könnte aber der chinesifizierte Name verwendet werden.
- Wir sagen „*Originalname*“?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.
- Egal. Es kann viele Gründe geben, warum eine Person entweder mehrere Namen hat oder sich der Name einer Person ändert.
- Den Name einer Person in einer Datenbank zu ändern, ist allerdings keine Option.

Spaß mit Namen

- Wir sagen *„Originalname“*?
- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.
- Egal. Es kann viele Gründe geben, warum eine Person entweder mehrere Namen hat oder sich der Name einer Person ändert.
- Den Name einer Person in einer Datenbank zu ändern, ist allerdings keine Option.
- Dann würde ja der alte Name verschwinden.

Spaß mit Namen

- Im Westen ist es nicht ungewöhnlich, dass Leute ihre Namen *ändern* wenn sie heiraten.
- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.
- Egal. Es kann viele Gründe geben, warum eine Person entweder mehrere Namen hat oder sich der Name einer Person ändert.
- Den Name einer Person in einer Datenbank zu ändern, ist allerdings keine Option.
- Dann würde ja der alte Name verschwinden.
- Dann hätten wir irgendwann alte Dokumente in der echten Welt die nicht mehr zu den geänderten Namen in der Datenbank passen.

Spaß mit Namen

- Was passiert, wenn *Elizabeth* nun *Herrn Heinrich Gieselher von Görlitz-Zittau* heiratet?
- Sie kann ihren Namen unverändert lassen.
- Sie kann den Familiennamen ihres Ehemannes annehmen und zu*Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.
- Egal. Es kann viele Gründe geben, warum eine Person entweder mehrere Namen hat oder sich der Name einer Person ändert.
- Den Name einer Person in einer Datenbank zu ändern, ist allerdings keine Option.
- Dann würde ja der alte Name verschwinden.
- Dann hätten wir irgendwann alte Dokumente in der echten Welt die nicht mehr zu den geänderten Namen in der Datenbank passen.
- Also müssen Namen ein *mehrwertiges* Attribut werden.

Spaß mit Namen

- Sie kann den Familiennamen ihres Ehemannes annehmen und zu *Mrs. Elizabeth Prudence von Görlitz-Zittau* werden.
- Oder ihr Ehemann kann ihren Familiennamen annehmen.
- Oder das neue Ehepaar entscheidet sich für einen zusammengesetzten Familiennamen.
- Vielleicht heißt sie dann *Frau Elizabeth Prudence von Görlitz-Zittau-McDouglas* der vielleicht *Frau Elizabeth Prudence McDouglas-von-Görlitz-Zittau*.
- Egal. Es kann viele Gründe geben, warum eine Person entweder mehrere Namen hat oder sich der Name einer Person ändert.
- Den Name einer Person in einer Datenbank zu ändern, ist allerdings keine Option.
- Dann würde ja der alte Name verschwinden.
- Dann hätten wir irgendwann alte Dokumente in der echten Welt die nicht mehr zu den geänderten Namen in der Datenbank passen.
- Also müssen Namen ein *mehrwertiges* Attribut werden.
- Wir werden wahrscheinlich jedem Namen auch noch einen Gültigkeitszeitraum zuordnen müssen, um mit Namensänderungen vernünftig umgehen zu können.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.

Personen, Studenten, und Mitarbeiter

- In dem Moment, wo wir anfangen, Namen zu modellieren, merken wir, dass das nicht nur ein Studenten-Problem ist.
- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.
- Die selbe Person könnte sich z. B. in mehrere Studiengänge einschreiben.

Personen, Studenten, und Mitarbeiter

- Das selbe Problem wird später wieder auftauchen, wenn wir die Mitarbeiter der Uni modellieren.
- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.
- Die selbe Person könnte sich z. B. in mehrere Studiengänge einschreiben.
- Vielleicht macht eine Person erstmal einen BSc in Informatik und hängt dann einen MSc dran.

Personen, Studenten, und Mitarbeiter

- Die können auch komische Namen haben.
- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.
- Die selbe Person könnte sich z. B. in mehrere Studiengänge einschreiben.
- Vielleicht macht eine Person erstmal einen BSc in Informatik und hängt dann einen MSc dran.
- Theoretisch könnte sogar ein Mitarbeiter einen Studiengang besuchen.

Personen, Studenten, und Mitarbeiter

- Die haben auch Mobiltelenfonnummern, Ausweisnummern, und Adressen.
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.
- Die selbe Person könnte sich z. B. in mehrere Studiengänge einschreiben.
- Vielleicht macht eine Person erstmal einen BSc in Informatik und hängt dann einen MSc dran.
- Theoretisch könnte sogar ein Mitarbeiter einen Studiengang besuchen.
- Vielleicht will ein Chemie-Lecturer noch einen MSc in Informatik machen.

Personen, Studenten, und Mitarbeiter

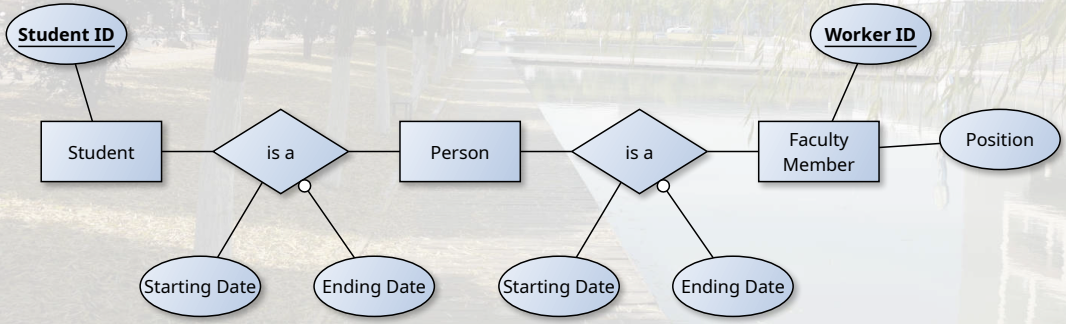
- Wir wollen das aber nicht alles zweimal modellieren.
- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.
- Die selbe Person könnte sich z. B. in mehrere Studiengänge einschreiben.
- Vielleicht macht eine Person erstmal einen BSc in Informatik und hängt dann einen MSc dran.
- Theoretisch könnte sogar ein Mitarbeiter einen Studiengang besuchen.
- Vielleicht will ein Chemie-Lecturer noch einen MSc in Informatik machen.
- Es wäre immer noch die selbe Person, und es ergibt wenig Sinn, all ihre Daten mehrfach zu speichern.

Personen, Studenten, und Mitarbeiter

- Das würde unser System nur komplexer machen und wir könnten Inkonsistenzen bekommen.
- Wir entscheiden uns, den neuen Entitätstyp *Person* einzuführen.
- Studenten und Mitarbeiter sind Personen.
- Wir werden alle Attribute, die sowohl Studenten als auch Mitarbeiter haben, in die Personen-Datensätze.
- Das ergibt sogar sehr viel Sinn.
- Die selbe Person könnte sich z. B. in mehrere Studiengänge einschreiben.
- Vielleicht macht eine Person erstmal einen BSc in Informatik und hängt dann einen MSc dran.
- Theoretisch könnte sogar ein Mitarbeiter einen Studiengang besuchen.
- Vielleicht will ein Chemie-Lecturer noch einen MSc in Informatik machen.
- Es wäre immer noch die selbe Person, und es ergibt wenig Sinn, all ihre Daten mehrfach zu speichern.
- Wir führen den neuen Entitätstype *Person* ein.

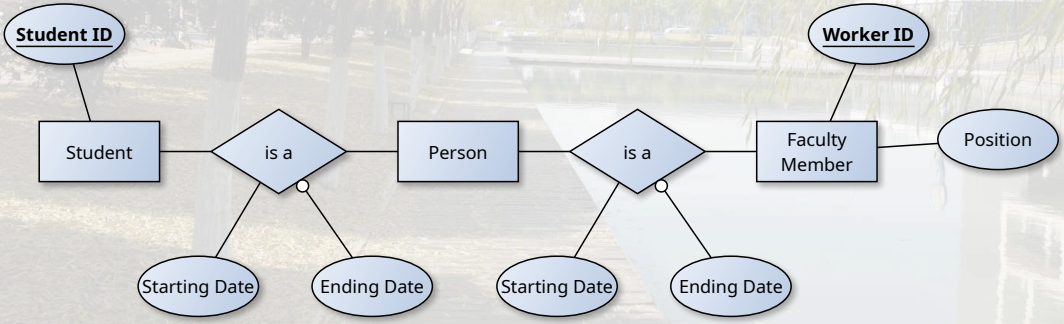
Personen, Studenten, und Mitarbeiter

- Wir führen den neuen Entitätstype *Person* ein.



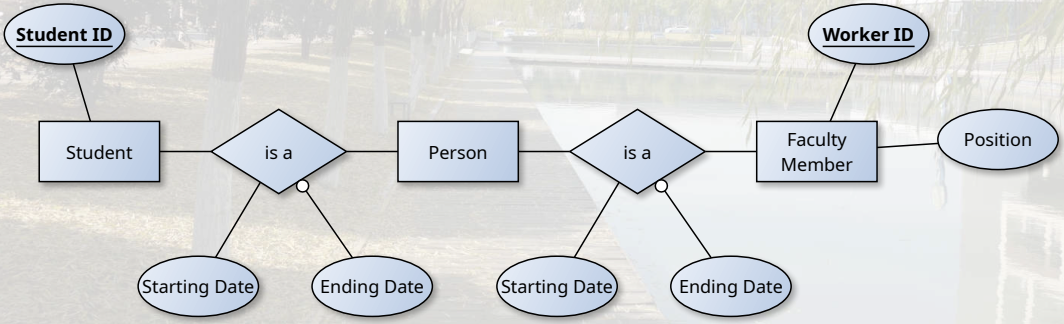
Personen, Studenten, und Mitarbeiter

- Wir führen den neuen Entitätstype *Person* ein.
- Später hängen wir alle gemeinsamen Attribute (wie die Namen) an diesen Entitätstyp.



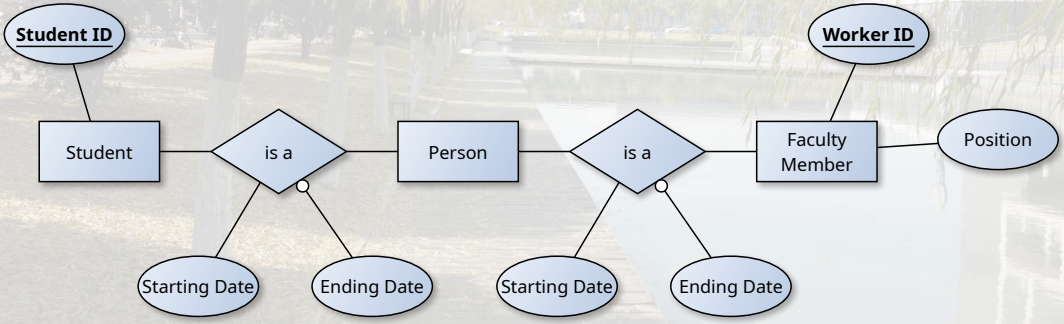
Personen, Studenten, und Mitarbeiter

- Wir führen den neuen Entitätstype *Person* ein.
- Später hängen wir alle gemeinsamen Attribute (wie die Namen) an diesen Entitätstyp.
- Wir denken erstmal nicht darüber nach, welchen Primärschlüssel wir für Personen verwenden werden.



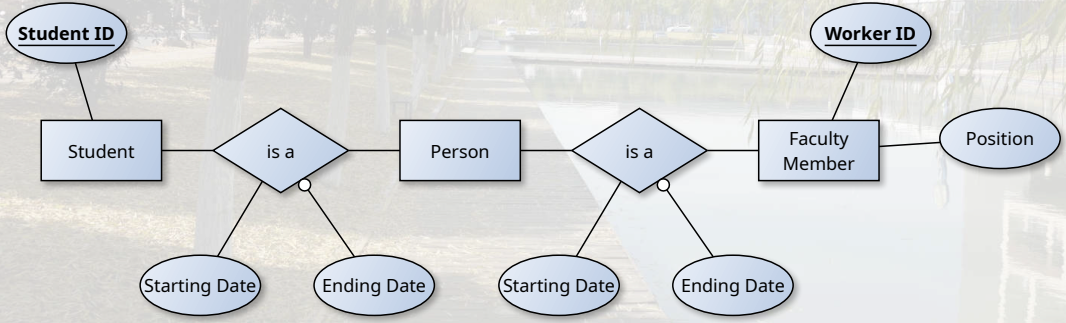
Personen, Studenten, und Mitarbeiter

- Wir führen den neuen Entitätstype *Person* ein.
- Später hängen wir alle gemeinsamen Attribute (wie die Namen) an diesen Entitätstyp.
- Wir denken erstmal nicht darüber nach, welchen Primärschlüssel wir für Personen verwenden werden.
- Der Entitätstyp *Student* braucht jetzt erstmal nur noch das Primärschlüsselattribut *Student ID*.



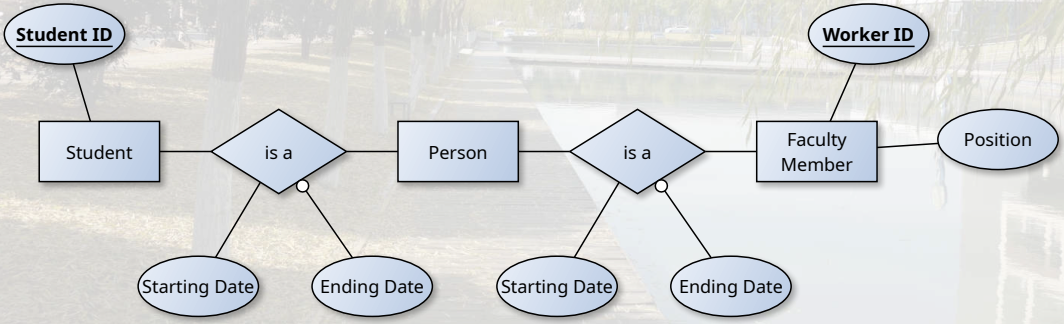
Personen, Studenten, und Mitarbeiter

- Später hängen wir alle gemeinsamen Attribute (wie die Namen) an diesen Entitätstyp.
- Wir denken erstmal nicht darüber nach, welchen Primärschlüssel wir für Personen verwenden werden.
- Der Entitätstyp *Student* braucht jetzt erstmal nur noch das Primärschlüsselattribut *Student ID*.
- Eine Person kann ein Student sein, was wir mit einem Beziehungstyp modellieren.



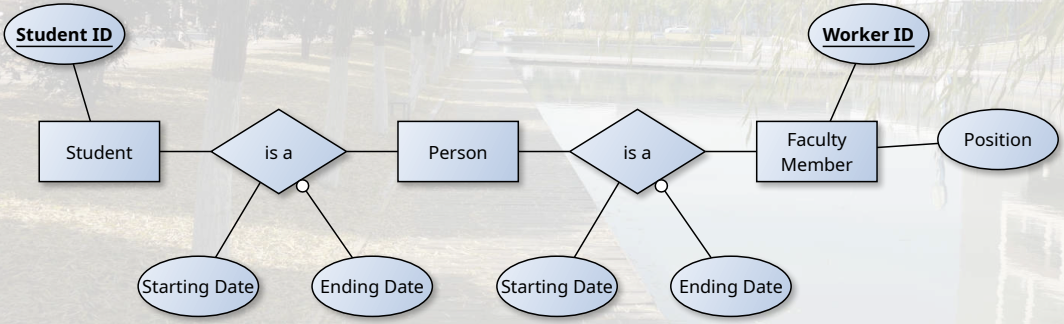
Personen, Studenten, und Mitarbeiter

- Wir denken erstmal nicht darüber nach, welchen Primärschlüssel wir für Personen verwenden werden.
- Der Entitätstyp *Student* braucht jetzt erstmal nur noch das Primärschlüsselattribut *Student ID*.
- Eine Person kann ein Student sein, was wir mit einem Beziehungstyp modellieren.
- So eine Beziehung hat ein Startdatum und ein optionales Enddatum.



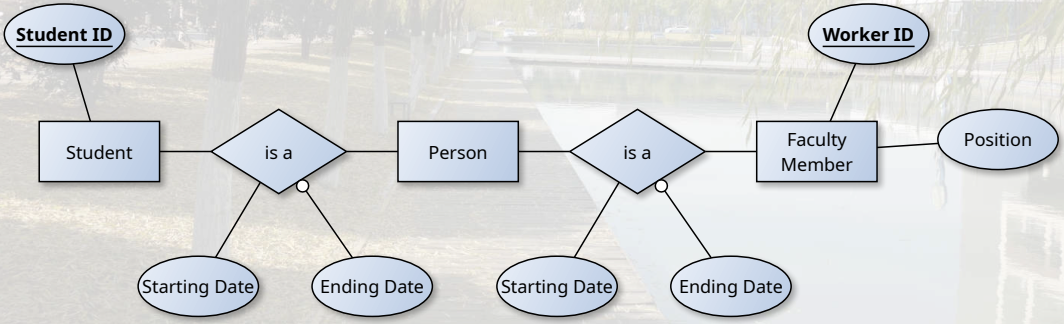
Personen, Studenten, und Mitarbeiter

- Der Entitätstyp *Student* braucht jetzt erstmal nur noch das Primärschlüsselattribut *Student ID*.
- Eine Person kann ein Student sein, was wir mit einem Beziehungstyp modellieren.
- So eine Beziehung hat ein Startdatum und ein optionales Enddatum.
- Das Enddatum ist `NULL` für alle Studenten, die aktuell eingeschrieben sind.



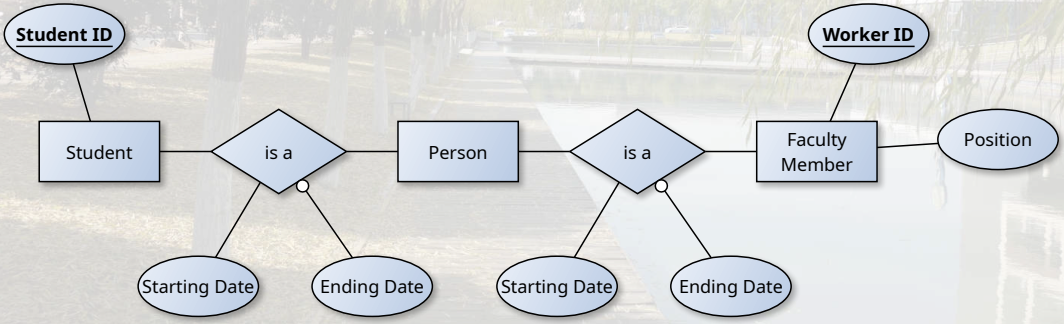
Personen, Studenten, und Mitarbeiter

- Eine Person kann ein Student sein, was wir mit einem Beziehungstyp modellieren.
- So eine Beziehung hat ein Startdatum und ein optionales Enddatum.
- Das Enddatum ist `NULL` für alle Studenten, die aktuell eingeschrieben sind.
- Für Mitarbeiter (EN: *Faculty Member*) ist die Situation ähnlich.



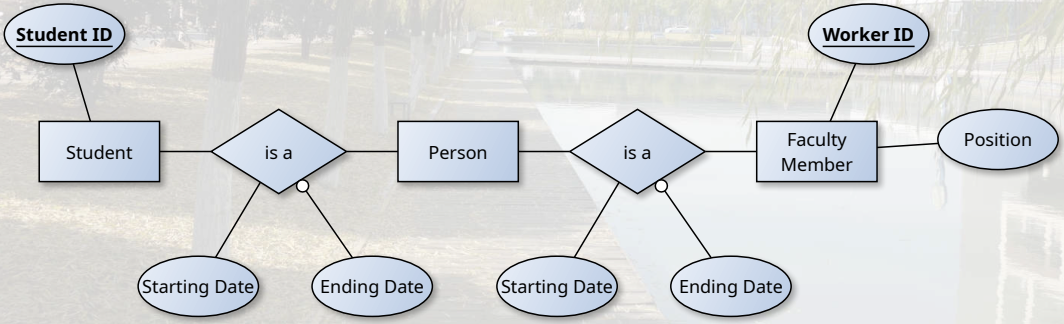
Personen, Studenten, und Mitarbeiter

- So eine Beziehung hat ein Startdatum und ein optionales Enddatum.
- Das Enddatum ist `NULL` für alle Studenten, die aktuell eingeschrieben sind.
- Für Mitarbeiter (EN: *Faculty Member*) ist die Situation ähnlich.
- Sie werden durch ihre Worker ID identifiziert.



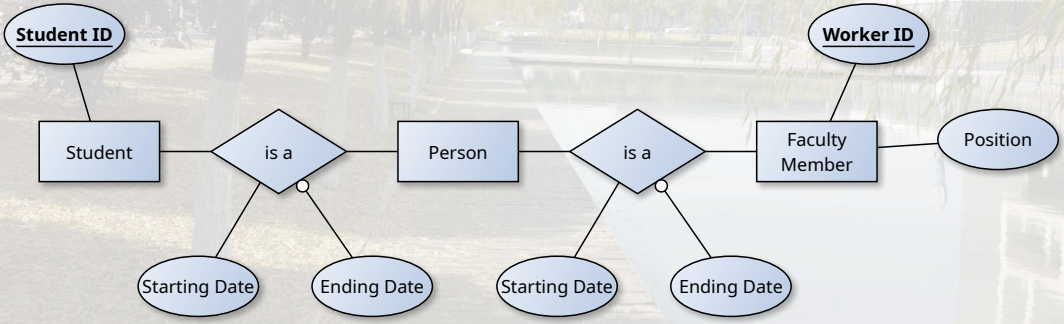
Personen, Studenten, und Mitarbeiter

- Das Enddatum ist `NULL` für alle Studenten, die aktuell eingeschrieben sind.
- Für Mitarbeiter (EN: *Faculty Member*) ist die Situation ähnlich.
- Sie werden durch ihre Worker ID identifiziert.
- Sie haben auch eine Position, z. B. Lecturer, Associate Professor, oder Full Professor.



Personen, Studenten, und Mitarbeiter

- Für Mitarbeiter (EN: *Faculty Member*) ist die Situation ähnlich.
- Sie werden durch ihre Worker ID identifiziert.
- Sie haben auch eine Position, z. B. Lecturer, Associate Professor, oder Full Professor.
- Auch diese Funktion hat ein Startdatum und ein optionales Enddatum.



Namen und Personen

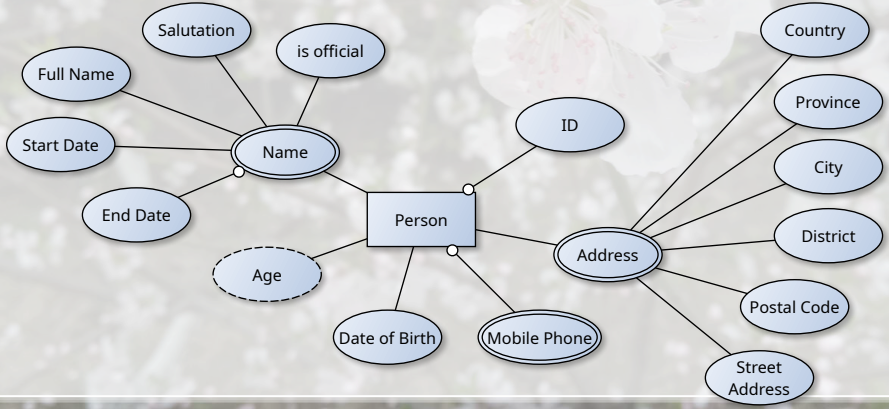
- Wir kopieren nun die Attribute aus dem alten Entitätstyp für Studenten in den neuen Entitätstyp für Personen.

Namen und Personen

- Wir kopieren nun die Attribute aus dem alten Entitätstyp für Studenten in den neuen Entitätstyp für Personen.
- Natürlich nicht die Student ID.

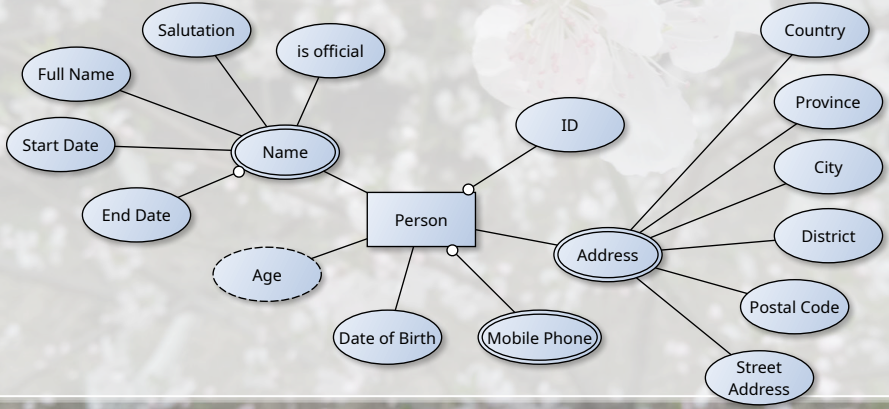
Namen und Personen

- Wir kopieren nun die Attribute aus dem alten Entitätstyp für Studenten in den neuen Entitätstyp für Personen.
- Natürlich nicht die Student ID.
- Wir lösen auch das Problem mit dem Attribut *Name*.



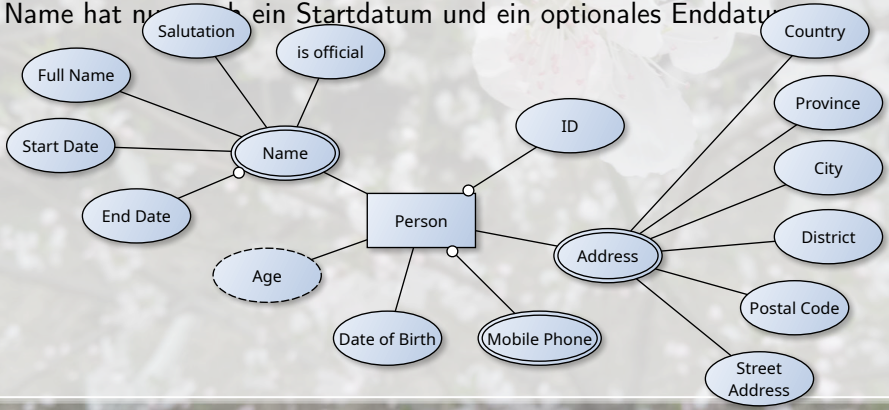
Namen und Personen

- Wir kopieren nun die Attribute aus dem alten Entitätstyp für Studenten in den neuen Entitätstyp für Personen.
- Natürlich nicht die Student ID.
- Wir lösen auch das Problem mit dem Attribut *Name*.
- Es ist jetzt mehrwertig.



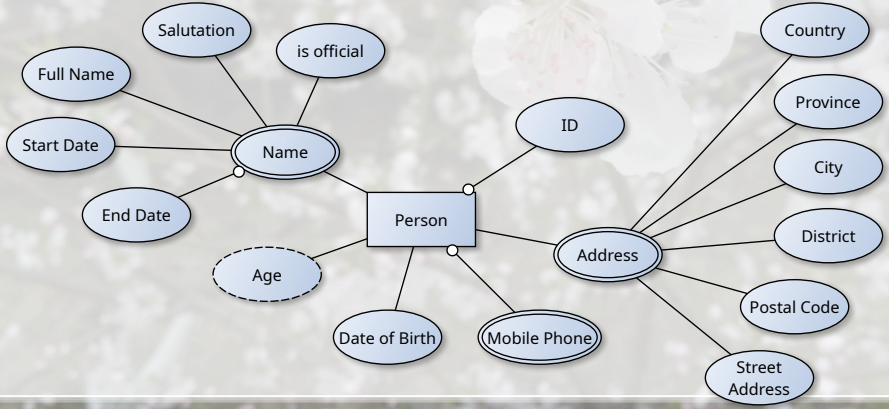
Namen und Personen

- Wir kopieren nun die Attribute aus dem alten Entitätstyp für Studenten in den neuen Entitätstyp für Personen.
- Natürlich nicht die Student ID.
- Wir lösen auch das Problem mit dem Attribut *Name*.
- Es ist jetzt mehrwertig.
- Jeder Name hat nu[illegible]h ein Startdatum und ein optionales Enddatu[illegible]



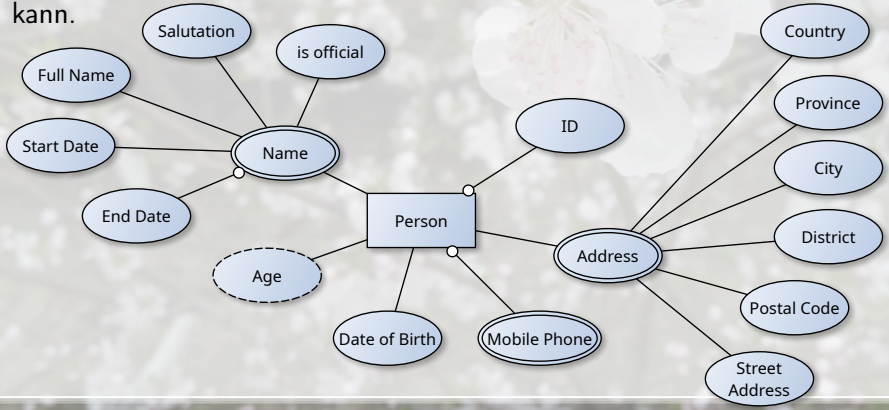
Namen und Personen

- Natürlich nicht die Student ID.
- Wir lösen auch das Problem mit dem Attribut *Name*.
- Es ist jetzt mehrwertig.
- Jeder Name hat nun auch ein Startdatum und ein optionales Enddatum.
- Ein Name kann *official* sein (Ja/Nein) – dann wird er für Dokumente verwendet.



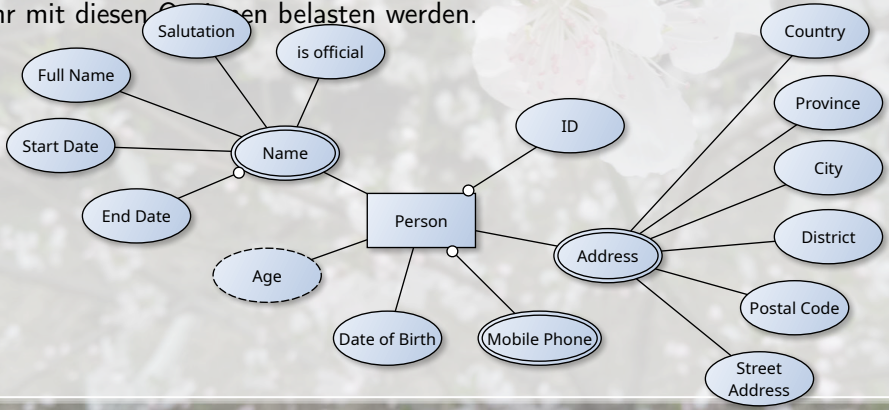
Namen und Personen

- Es ist jetzt mehrwertig.
- Jeder Name hat nun auch ein Startdatum und ein optionales Enddatum.
- Ein Name kann *official* sein (Ja/Nein) – dann wird er für Dokumente verwendet.
- Wir werden eine Einschränkung definieren, dass immer nur genau ein Name *offical* sein kann, dass sein Startdatum nicht nach heute liegen muss, und dass er kein Enddatum haben kann.



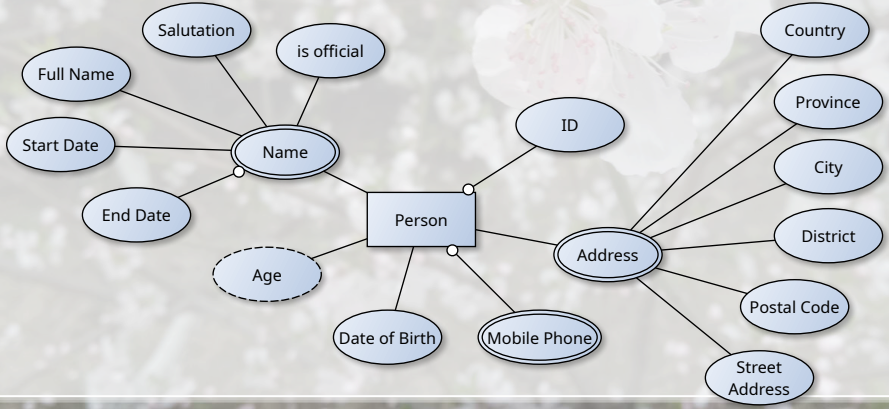
Namen und Personen

- Ein Name kann *official* sein (Ja/Nein) – dann wird er für Dokumente verwendet.
- Wir werden eine Einschränkung definieren, dass immer nur genau ein Name *offical* sein kann, dass sein Startdatum nicht nach heute liegen muss, und dass er kein Enddatum haben kann.
- Wir stellen uns vor, dass das wir die Person, die die Daten in die Datenbank eingibt, nicht zu sehr mit diesen Details belasten werden.



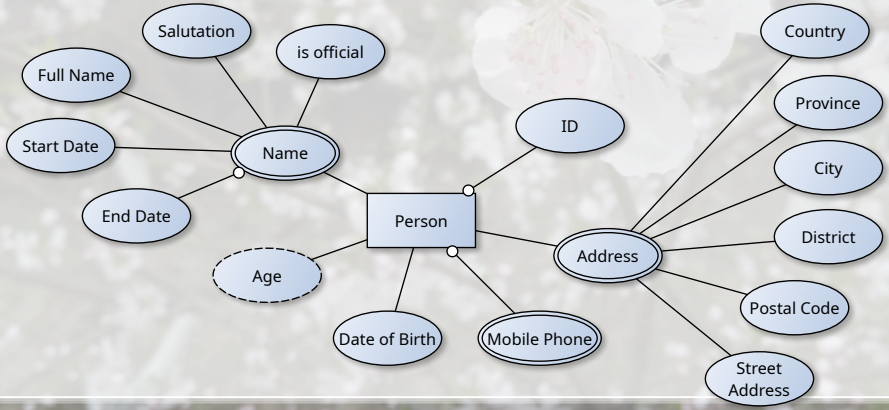
Namen und Personen

- Wir stellen uns vor, dass das wir die Person, die die Daten in die Datenbank eingibt, nicht zu sehr mit diesen Optionen belasten werden.
- Sie kann dann einfach nur einen Name eingeben, dessen Startdatum dann als heute gesetzt wird und der automatisch als *official* markiert wird.



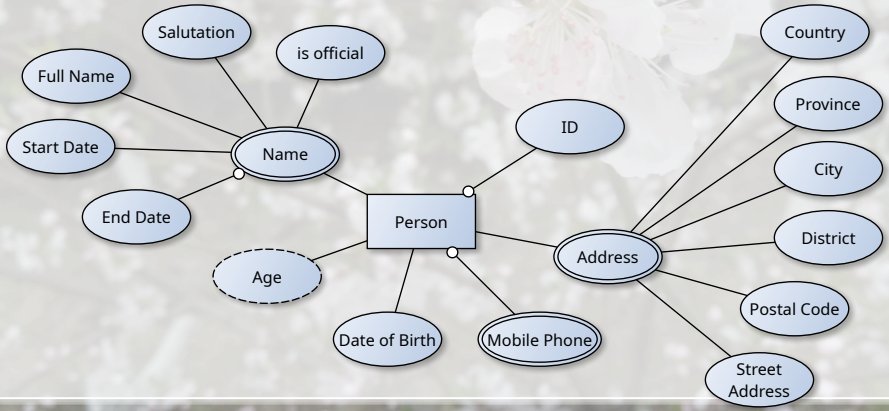
Namen und Personen

- Sie kann dann einfach nur einen Name eingeben, dessen Startdatum dann als heute gesetzt wird und der automatisch als *official* markiert wird.
- Das System wird auch erstmal die *Salutation* und den *Full Name* auf den selben Wert setzen.



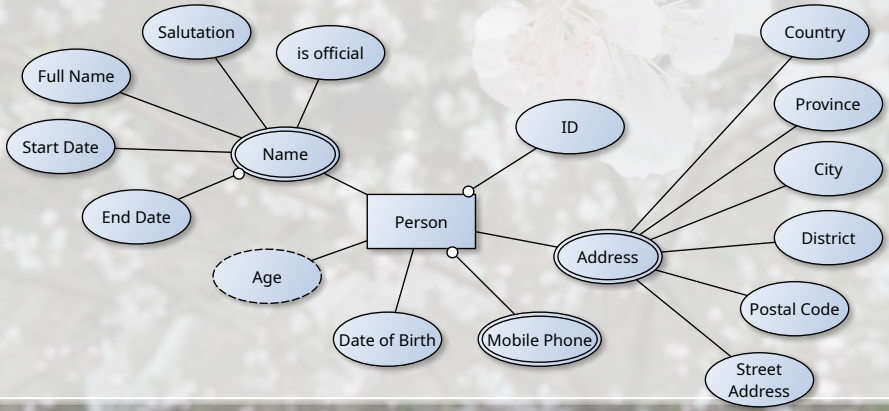
Namen und Personen

- Das System wird auch erstmal die *Salutation* und den *Full Name* auf den selben Wert setzen.
- Die Daten können aber dann später nachbearbeitet werden.



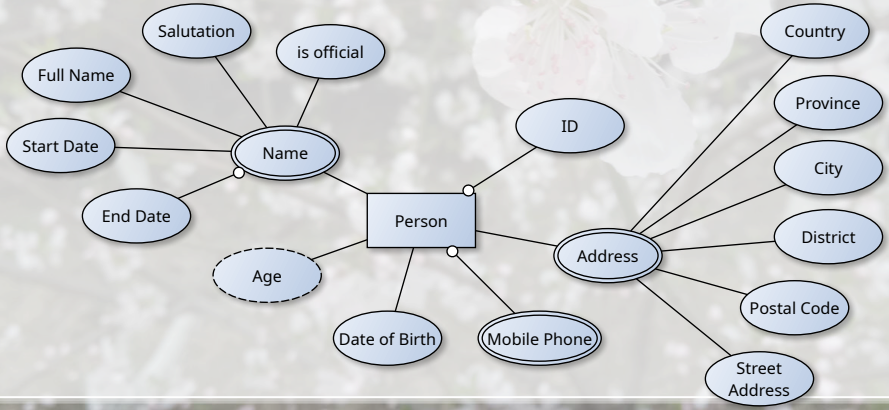
Namen und Personen

- Die Daten können aber dann später nachbearbeitet werden.
- Dann macht das Eingeben von Daten wenig Arbeit aber wir haben alle notwendigen Optionen, um auch mit den komischsten Situationen klarzukommen.



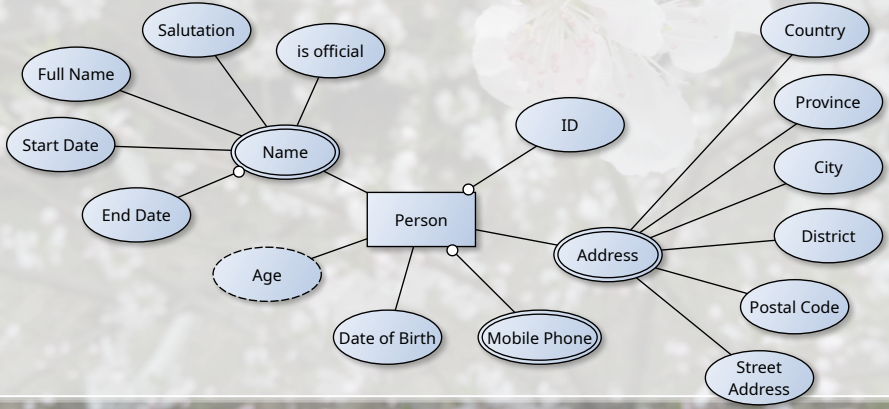
Namen und Personen

- Dann macht das Eingeben von Daten wenig Arbeit aber wir haben alle notwendigen Optionen, um auch mit den komischsten Situationen klarzukommen.
- Beachten Sie, dass es wichtig, all solche Sonderfälle von Anfang an zu beachten.



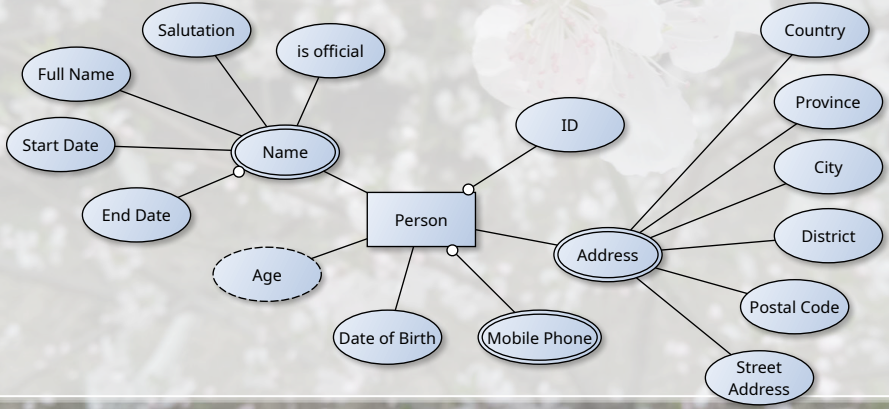
Namen und Personen

- Beachten Sie, dass es wichtig, all solche Sonderfälle von Anfang an zu beachten.
- Wir bemerken, dass wir nun keinen vernünftigen Kandidaten für den Primärschlüssel mehr haben.



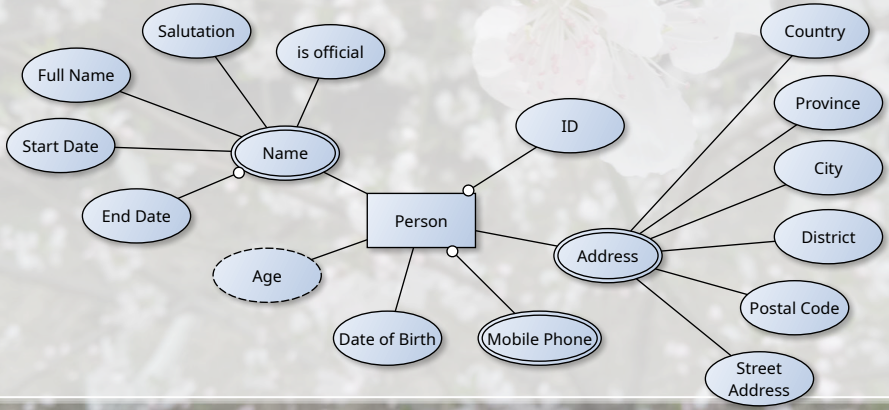
Namen und Personen

- Wir bemerken, dass wir nun keinen vernünftigen Kandidaten für den Primärschlüssel mehr haben.
- *Name* und *Address* sind beides zusammengesetzte Attribute deren Werte nicht einzigartig seien müssen.



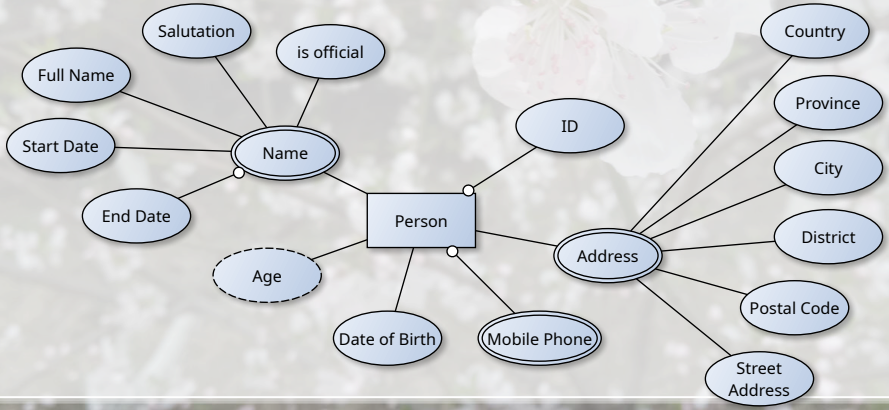
Namen und Personen

- *Name* und *Address* sind beides zusammengesetzte Attribute deren Werte nicht einzigartig seien müssen.
- Die Ausweisnummer *ID* ist optional.



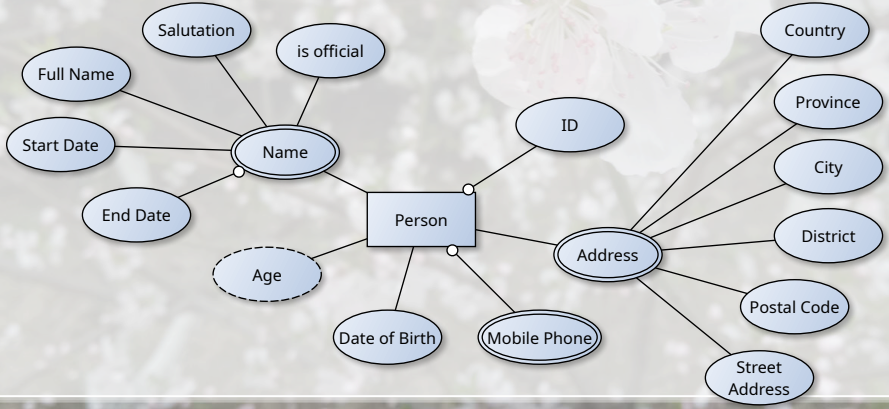
Namen und Personen

- Die Ausweisnummer *ID* ist optional.
- *Mobile Phone* ist sowohl optional als auch mehrwertig.



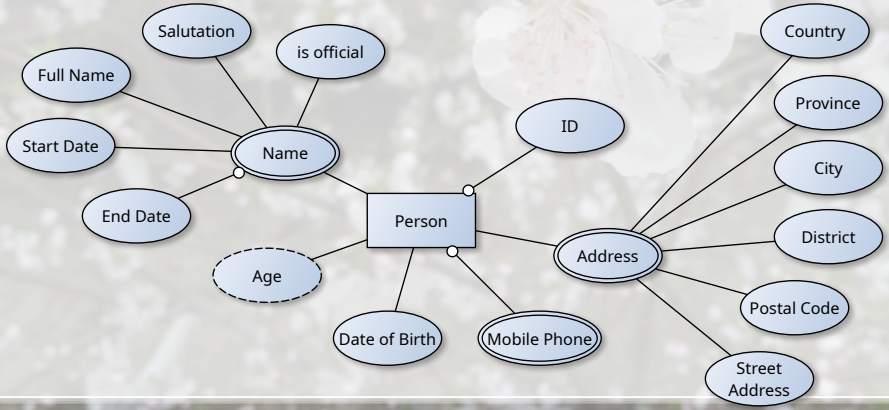
Namen und Personen

- *Mobile Phone* ist sowohl optional als auch mehrwertig.
- *Date of Birth* ist definitiv nicht einzigartig und *Age* ist auch noch abgeleitet.



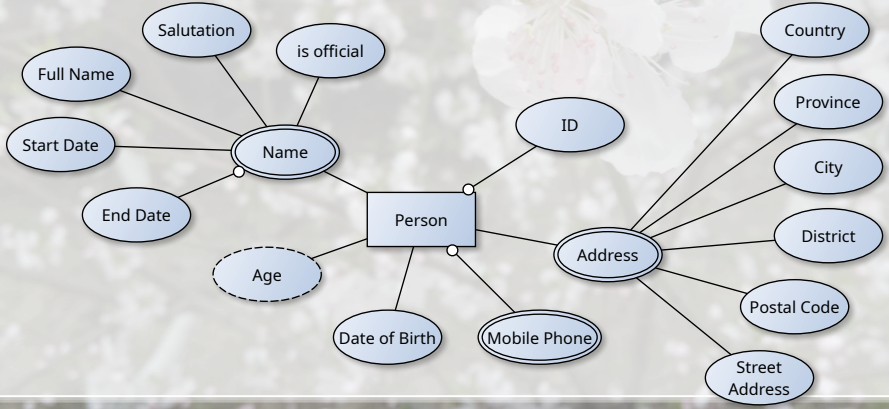
Namen und Personen

- *Date of Birth* ist definitiv nicht einzigartig und *Age* ist auch noch abgeleitet.
- Keine davon und auch keine Kombination davon ist nicht-optional *und* einzigartig.



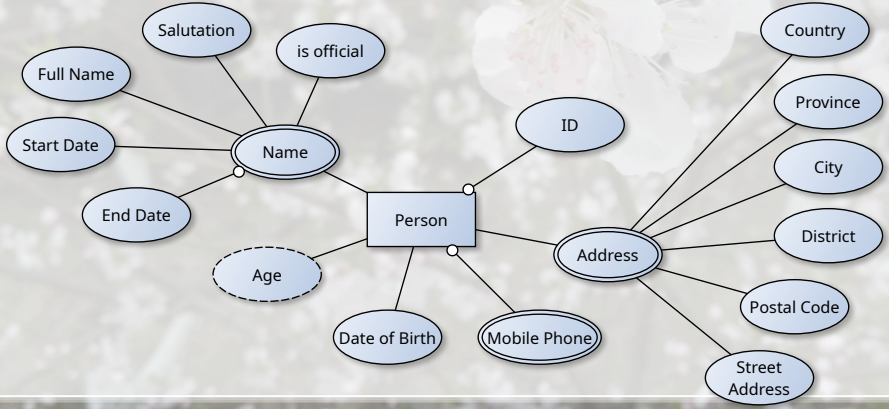
Namen und Personen

- Keine davon und auch keine Kombination davon ist nicht-optional *und* einzigartig.
- Natürlich werden wir später andere ID-Werte wie Reisepassnummer und Emailadresse hinzufügen.



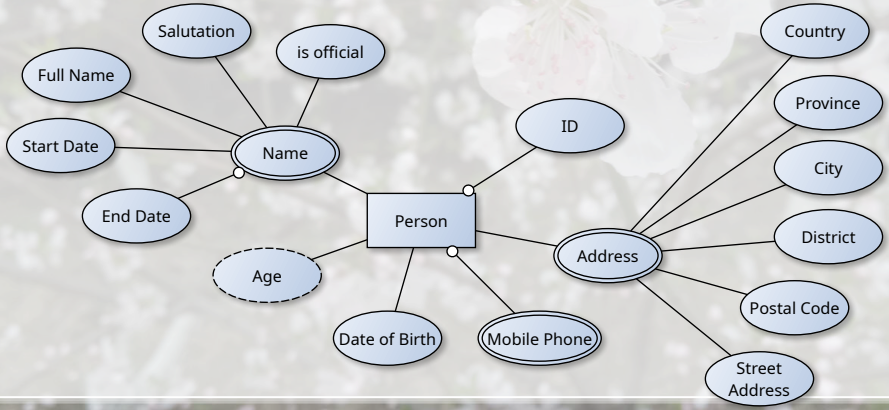
Namen und Personen

- Natürlich werden wir später andere ID-Werte wie Reisepassnummer und Emailadresse hinzufügen.
- Unsere Anwendung könnte erzwingen, dass jede Person entweder eine Reisepassnummer oder eine Ausweisnummer hat und das diese einzigartig seien müssen.



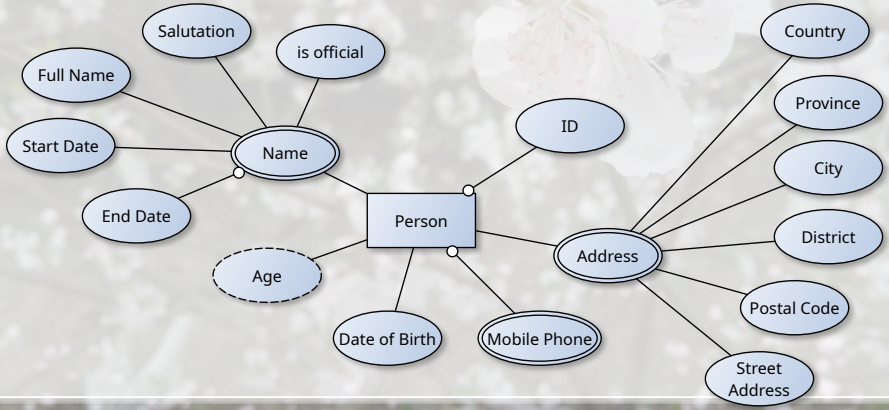
Namen und Personen

- Unsere Anwendung könnte erzwingen, dass jede Person entweder eine Reisepassnummer oder eine Ausweisnummer hat und das diese einzigartig seien müssen.
- Daraus einen Primärschlüssel zu bauen wäre aber sehr umständlich.



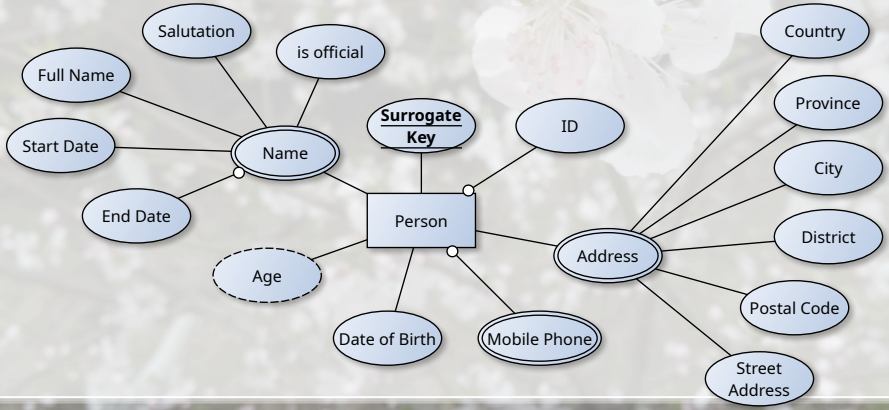
Namen und Personen

- Daraus einen Primärschlüssel zu bauen wäre aber sehr umständlich.
- Die einfachste Lösung ist also ein Ersatzschlüssel, ein so genannter (EN: *surrogate key*).



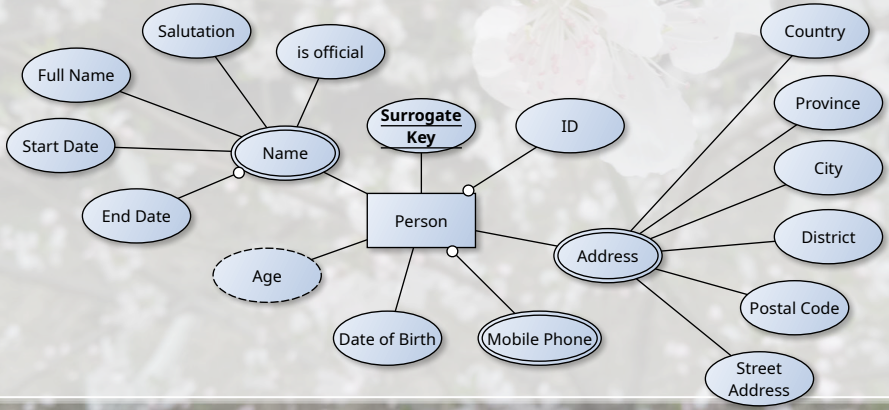
Namen und Personen

- Die einfachste Lösung ist also ein Ersatzschlüssel, ein so genannter (EN: *surrogate key*).
- So nennen wir den Schlüssel auch gleich in unserem neuen Modell.



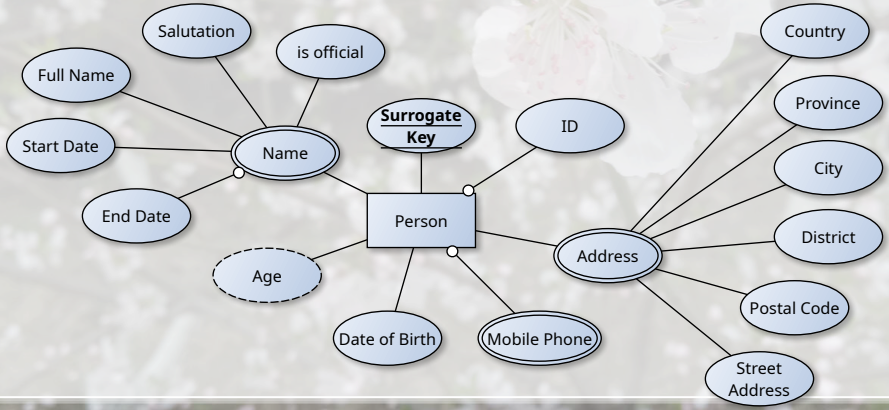
Namen und Personen

- So nennen wir den Schlüssel auch gleich in unserem neuen Modell.
- Wir nehmen dazu an, dass das DBMS, was wir später verwenden werden, irgendwie einzigartige Werte generieren kann.



Namen und Personen

- Wir nehmen dazu an, dass das DBMS, was wir später verwenden werden, irgendwie einzigartige Werte generieren kann.
- Damit sieht unser Entitätstyp *Person* nun schon recht vernünftig aus.



Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern ... die allerdings nicht dauerhaft sind.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern ... die allerdings nicht dauerhaft sind.
- Chinesische Ausweisnummern eine Person identifizieren und gleich bleiben, identifizieren Reisepassnummern den Reisepass als Dokument.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern ... die allerdings nicht dauerhaft sind.
- Chinesische Ausweisnummern eine Person identifizieren und gleich bleiben, identifizieren Reisepassnummern den Reisepass als Dokument.
- Ein Reisepass ist oftmals 10 Jahre gültig, danach bekommt die Person einen neuen Pass mit einer neuen Nummer.

Spaß mit IDs

- Nun, wo wir das Problem mit den Namen gelöst haben, wollen wir uns mit IDs und ID-Dokumenten beschäftigen.
- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern ... die allerdings nicht dauerhaft sind.
- Chinesische Ausweisnummern eine Person identifizieren und gleich bleiben, identifizieren Reisepassnummern den Reisepass als Dokument.
- Ein Reisepass ist oftmals 10 Jahre gültig, danach bekommt die Person einen neuen Pass mit einer neuen Nummer.
- Wenn ein Ausländer China betritt, dann braucht sie Visum, welches wiederum eine einmalige Nummer hat und eine Gültigkeitsdauer[38].

Spaß mit IDs

- Bis jetzt haben wir die IDs als die von der Regierung ausgegebene Ausweisnummer (中国公民身份号码[37]) modelliert.
- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern . . . die allerdings nicht dauerhaft sind.
- Chinesische Ausweisnummern eine Person identifizieren und gleich bleiben, identifizieren Reisepassnummern den Reisepass als Dokument.
- Ein Reisepass ist oftmals 10 Jahre gültig, danach bekommt die Person einen neuen Pass mit einer neuen Nummer.
- Wenn ein Ausländer China betritt, dann braucht sie Visum, welches wiederum eine einmalige Nummer hat und eine Gültigkeitsdauer[38].
- Es gibt viele verschiedene Typen für Visa für verschiedene Tätigkeiten, z. B. sind X1- und X2-Visa zum Studieren und Z-Visa zum Arbeiten.

Spaß mit IDs

- Das war ein optionales Attribut, weil ausländische Austauschstudenten und Mitarbeiter so etwas nicht haben.
- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern ... die allerdings nicht dauerhaft sind.
- Chinesische Ausweisnummern eine Person identifizieren und gleich bleiben, identifizieren Reisepassnummern den Reisepass als Dokument.
- Ein Reisepass ist oftmals 10 Jahre gültig, danach bekommt die Person einen neuen Pass mit einer neuen Nummer.
- Wenn ein Ausländer China betritt, dann braucht sie Visum, welches wiederum eine einmalige Nummer hat und eine Gültigkeitsdauer[38].
- Es gibt viele verschiedene Typen für Visa für verschiedene Tätigkeiten, z. B. sind X1- und X2-Visa zum Studieren und Z-Visa zum Arbeiten.
- Ausländer, die in China arbeiten (z. B. an einer Uni) brauchen dann noch eine Arbeitserlaubnis[36], die wiederum eine einmalige Kennzahl hat und eine Gültigkeitsdauert.

Spaß mit IDs

- Gut, sie haben sicherlich schon ein Ausweisdokument, nur kein chinesisches.
- Diese ausländischen Dokumente wären für uns nutzlos.
- Andererseits haben Ausländer, die nach China kommen, einen Reisepass[21].
- Reisepässe haben auch eindeutige Nummern ... die allerdings nicht dauerhaft sind.
- Chinesische Ausweisnummern eine Person identifizieren und gleich bleiben, identifizieren Reisepassnummern den Reisepass als Dokument.
- Ein Reisepass ist oftmals 10 Jahre gültig, danach bekommt die Person einen neuen Pass mit einer neuen Nummer.
- Wenn ein Ausländer China betritt, dann braucht sie Visum, welches wiederum eine einmalige Nummer hat und eine Gültigkeitsdauer[38].
- Es gibt viele verschiedene Typen für Visa für verschiedene Tätigkeiten, z. B. sind X1- und X2-Visa zum Studieren und Z-Visa zum Arbeiten.
- Ausländer, die in China arbeiten (z. B. an einer Uni) brauchen dann noch eine Arbeitserlaubnis[36], die wiederum eine einmalige Kennzahl hat und eine Gültigkeitsdauert.
- Wenn wir das als Attribute der Person-Entität modellieren wollen, kann das beliebig kompliziert werden.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode . . . das könnte sich ja ändern.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode . . . das könnte sich ja ändern.
- Oder vielleicht gibt es später andere Visa-Typen.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode ... das könnte sich ja ändern.
- Oder vielleicht gibt es später andere Visa-Typen.
- Dann können wir einfach einen neuen ID-Typ hinzufügen.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode ... das könnte sich ja ändern.
- Oder vielleicht gibt es später andere Visa-Typen.
- Dann können wir einfach einen neuen ID-Typ hinzufügen.
- Wir können auch weitere Attribute an diesen Entitätstyp hänge, z. B. *is for communication*, *is ID document*, usw., um mehr Kontext bereitzustellen.

Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode ... das könnte sich ja ändern.
- Oder vielleicht gibt es später andere Visa-Typen.
- Dann können wir einfach einen neuen ID-Typ hinzufügen.
- Wir können auch weitere Attribute an diesen Entitätstyp hänge, z. B. *is for communication*, *is ID document*, usw., um mehr Kontext bereitzustellen.
- Damit haben wir nun alle Kommunikations- und Identifikationswerte in einen Entitätstyp vereinigt.

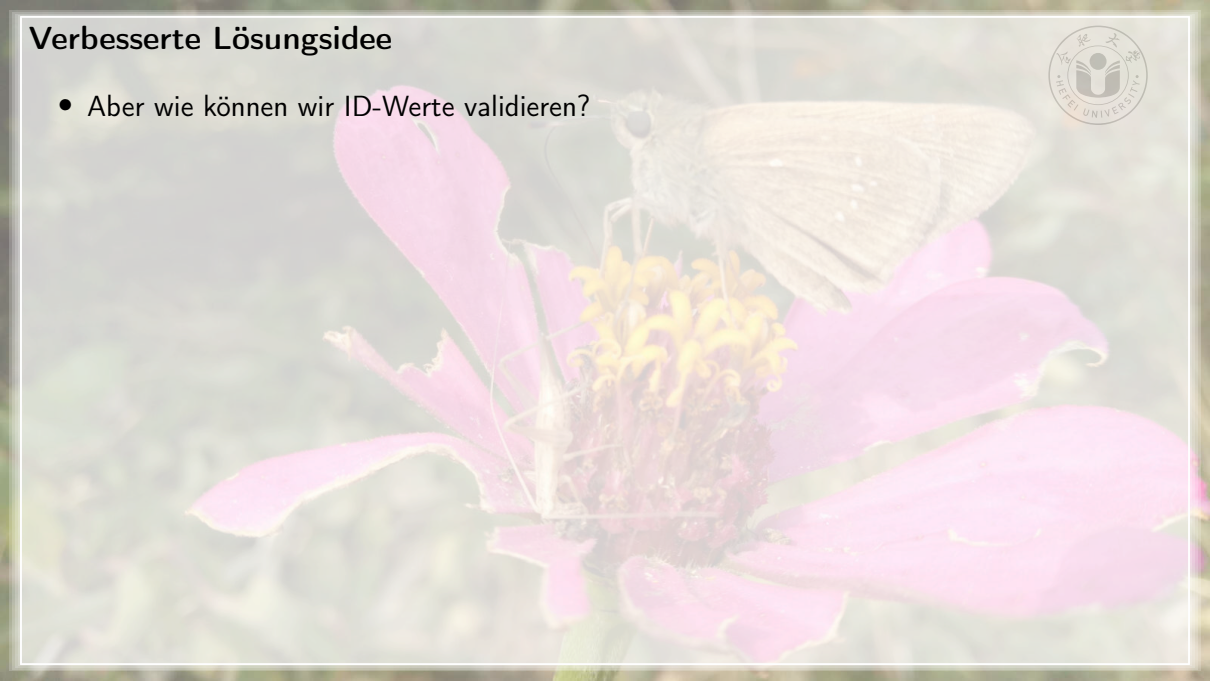
Lösungsidee

- Wie wäre es damit: Wir erstellen einen Entitätstyp für *ID Type*s.
- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode ... das könnte sich ja ändern.
- Oder vielleicht gibt es später andere Visa-Typen.
- Dann können wir einfach einen neuen ID-Typ hinzufügen.
- Wir können auch weitere Attribute an diesen Entitätstyp hänge, z. B. *is for communication*, *is ID document*, usw., um mehr Kontext bereitzustellen.
- Damit haben wir nun alle Kommunikations- und Identifikationswerte in einen Entitätstyp vereinigt.
- Wir könnten nun mehrere ID-Werte und mehrere Telefonnummern oder mehrere Email-Adressen pro Person speichern.

Lösungsidee

- Eine Entät dieses Typs speichert den Name des ID-Typs, welcher der Primärschlüssel ist.
- Das könnte z. B. „Mobile Phone (China)“, „Mobile Phone (Intl)“, „ID (中华人民共和国居民身份证)“, „Passport (护照),“ „Work Permit (中华人民共和国外国人工作许可证)“, „X1-Visa“, „X2-Visa“, „Z-Visa“, „Email Address“, „WeChat User Name“, usw. sein.
- Damit wird unser System gleich Zukunftssicher:
- Emails werden gerade langsam weniger wichtig und WeChat wird die Hauptkommunikationsmethode ... das könnte sich ja ändern.
- Oder vielleicht gibt es später andere Visa-Typen.
- Dann können wir einfach einen neuen ID-Typ hinzufügen.
- Wir können auch weitere Attribute an diesen Entitätstyp hänge, z. B. *is for communication*, *is ID document*, usw., um mehr Kontext bereitzustellen.
- Damit haben wir nun alle Kommunikations- und Identifikationswerte in einen Entitätstyp vereinigt.
- Wir könnten nun mehrere ID-Werte und mehrere Telefonnummern oder mehrere Email-Adressen pro Person speichern.
- Jeder ID-Wert wäre mit einem ID-Typ assoziiert.

Verbesserte Lösungsidee



- Aber wie können wir ID-Werte validieren?

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.
- Nun, wir können das der App überlassen, die später für die Dateneingabe benutzt werden wird.

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.
- Nun, wir können das der App überlassen, die später für die Dateneingabe benutzt werden wird.
- Wir können aber auch ein paar grundlegende Tests gleich auf Ebene der Datenbank einführen.

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.
- Nun, wir können das der App überlassen, die später für die Dateneingabe benutzt werden wird.
- Wir können aber auch ein paar grundlegende Tests gleich auf Ebene der Datenbank einführen.
- Wir haben ja schon Regular Expressions kennengelernt, also Text-Strings, die ein Format beschreiben, mit dem andere Text-Strings verglichen werden können.

Verbesserte Lösungsidee

- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.
- Nun, wir können das der App überlassen, die später für die Dateneingabe benutzt werden wird.
- Wir können aber auch ein paar grundlegende Tests gleich auf Ebene der Datenbank einführen.
- Wir haben ja schon Regular Expressions kennengelernt, also Text-Strings, die ein Format beschreiben, mit dem andere Text-Strings verglichen werden können.
- Wir speichern also zu jedem ID-Type auch ein regex „Validation RegEx“ als Attribut.

Verbesserte Lösungsidee

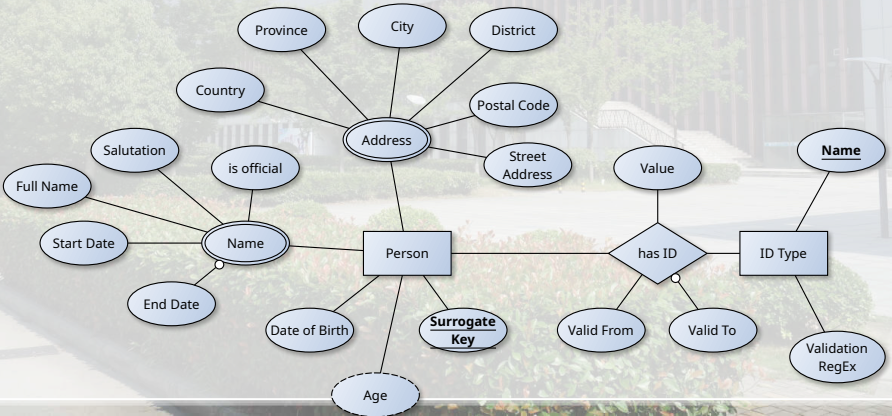
- Aber wie können wir ID-Werte validieren?
- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.
- Nun, wir können das der App überlassen, die später für die Dateneingabe benutzt werden wird.
- Wir können aber auch ein paar grundlegende Tests gleich auf Ebene der Datenbank einführen.
- Wir haben ja schon Regular Expressions kennengelernt, also Text-Strings, die ein Format beschreiben, mit dem andere Text-Strings verglichen werden können.
- Wir speichern also zu jedem ID-Type auch ein regex „Validation RegEx“ als Attribut.
- Wenn die Administatorin eine ID eines bestimmten Typs eintippt, dann soll diese mit diesem Format verglichen werden.

Verbesserte Lösungsidee

- Wenn ein neuer ID-Wert eingegeben wird, dann sollte der zumindest rudimentär geprüft werden.
- Mobiltelefonnummern und Reisepassnummern sind sehr verschieden.
- Trotzdem müssen wir verhindern, dass irgendwie Buchstaben in einer Telefonnummer vorkommen.
- Nun, wir können das der App überlassen, die später für die Dateneingabe benutzt werden wird.
- Wir können aber auch ein paar grundlegende Tests gleich auf Ebene der Datenbank einführen.
- Wir haben ja schon Regular Expressions kennengelernt, also Text-Strings, die ein Format beschreiben, mit dem andere Text-Strings verglichen werden können.
- Wir speichern also zu jedem ID-Type auch ein regex „Validation RegEx“ als Attribut.
- Wenn die Administatorin eine ID eines bestimmten Typs eintippt, dann soll diese mit diesem Format verglichen werden.
- Wir können so zwar keine sehr genauen Überprüfungen durchführen, aber zumindest ein paar einfache Fehler verhindern.

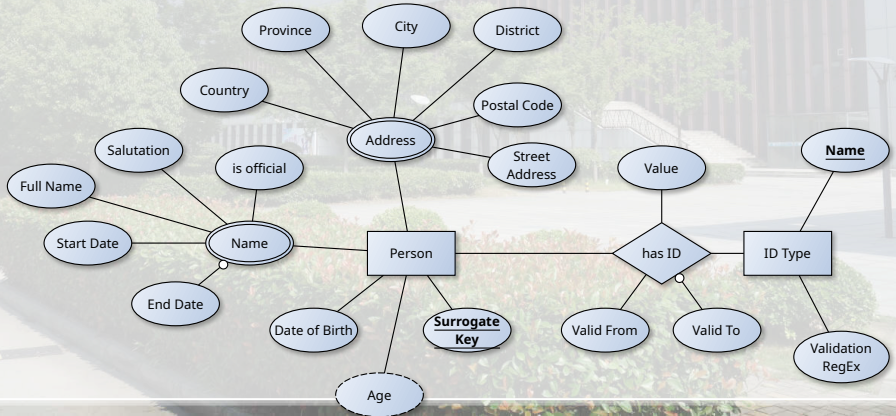
Neues ERD

- Wir können nun einen neuen Beziehungstyp zwischen den Entitätstypen *Person* und *ID Type* einführen.



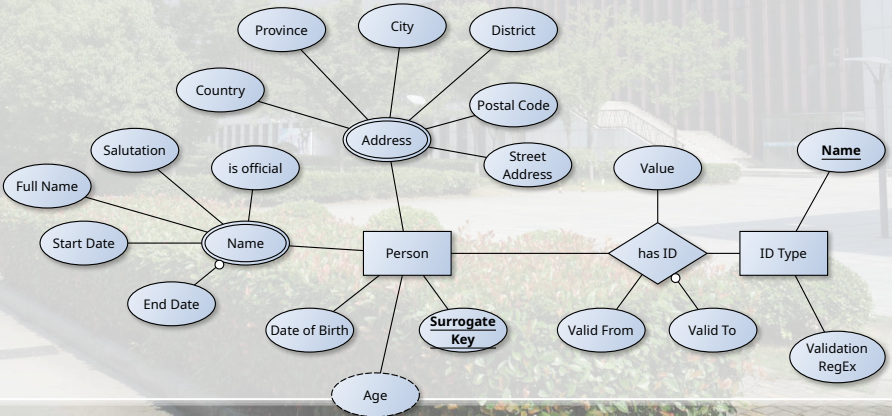
Neues ERD

- Wir können nun einen neuen Beziehungstyp zwischen den Entitätstypen *Person* und *ID Type* einführen.
- Eine Person hat IDs.



Neues ERD

- Wir können nun einen neuen Beziehungstyp zwischen den Entitätstypen *Person* und *ID Type* einführen.
- Eine Person hat IDs.
- Jede ID hat ein Attribut *Valid From* und optional ein *Valid To*-Datum.



Zusammenfassung

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.
- Ohne Beziehungen könnten wir unsere Daten genauso gut in Gruppen einfacher Dateien speichern.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.
- Ohne Beziehungen könnten wir unsere Daten genauso gut in Gruppen einfacher Dateien speichern.
- Aber so bald Entitäten in Beziehung stehen, geht das nicht mehr.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.
- Ohne Beziehungen könnten wir unsere Daten genauso gut in Gruppen einfacher Dateien speichern.
- Aber so bald Entitäten in Beziehung stehen, geht das nicht mehr.
- Man kann nicht erlauben, dass eine Entität, die in einer Beziehung teilnimmt, gelöscht wird bevor die Beziehung gelöscht wird.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.
- Ohne Beziehungen könnten wir unsere Daten genauso gut in Gruppen einfacher Dateien speichern.
- Aber so bald Entitäten in Beziehung stehen, geht das nicht mehr.
- Man kann nicht erlauben, dass eine Entität, die in einer Beziehung teilnimmt, gelöscht wird bevor die Beziehung gelöscht wird.
- Beziehungen können nur zwischen Entitäten erzeugt werden, die auch existieren.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.
- Ohne Beziehungen könnten wir unsere Daten genauso gut in Gruppen einfacher Dateien speichern.
- Aber so bald Entitäten in Beziehung stehen, geht das nicht mehr.
- Man kann nicht erlauben, dass eine Entität, die in einer Beziehung teilnimmt, gelöscht wird bevor die Beziehung gelöscht wird.
- Beziehungen können nur zwischen Entitäten erzeugt werden, die auch existieren.
- Deshalb erfordern unsere Modelle nun, das die Technologie, mit denen sie irgendwann implementiert werden, das auch unterstützen.

Zusammenfassung

- Jetzt können wir Beziehungen modellieren.
- Beziehungen verbinden zwei oder mehr Entitäten.
- Ohne Beziehungen könnten wir unsere Daten genauso gut in Gruppen einfacher Dateien speichern.
- Aber so bald Entitäten in Beziehung stehen, geht das nicht mehr.
- Man kann nicht erlauben, dass eine Entität, die in einer Beziehung teilnimmt, gelöscht wird bevor die Beziehung gelöscht wird.
- Beziehungen können nur zwischen Entitäten erzeugt werden, die auch existieren.
- Deshalb erfordern unsere Modelle nun, das die Technologie, mit denen sie irgendwann implementiert werden, das auch unterstützen.
- Relationale Datenbanken tuen das.

谢谢您们！

Thank you!

Vielen Dank!

References I

[1] 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. 197).

[2] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen und Eve Maler, Hrsg. *Extensible Markup Language (XML) 1.0 (Fifth Edition)*. W3C Recommendation. Wakefield, MA, USA: World Wide Web Consortium (W3C), 26. Nov. 2008–7. Feb. 2013. URL: http://www.w3.org/TR/2008/REC-xml-20081126 (besucht am 2024-12-15) (siehe S. 5–10, 198).

[3] 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. 197).

[4] Peter Pin-Shan Chen. "English, Chinese and ER Diagrams". *Data & Knowledge Engineering (DKE)* 23(1):5–16, Juni 1997. Amsterdam, The Netherlands: Elsevier B.V. ISSN: 0169-023X. doi:10.1016/S0169-023X(97)00017-7. URL: https://www.csc.lsu.edu/~chen/pdf/ER_C.pdf (besucht am 2025-04-06) (siehe S. 12–17).

[5] 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. 197).

[6] 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. 192, 197).

[7] 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[6] for a more comprehensive introduction. (Siehe S. 197).

[8] Timothy W. Cole und Myung-Ja K. Han. *XML for Catalogers and Metadata Librarians (Third Millennium Cataloging)*. 1. Aufl. Dublin, OH, USA: Libraries Unlimited, 23. Mai 2013. ISBN: 978-1-59884-519-8 (siehe S. 5–10, 198).

References II

[9] "csv – CSV File Reading and Writing". In: *Python 3 Documentation. The Python Standard Library*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: https://docs.python.org/3/library/csv.html (besucht am 2024-11-14) (siehe S. 197).

[10] ."transitive (adjective)". In: *Merriam-Webster: America's Most Trusted Dictionary*. Hrsg. von Editors of Merriam-Webster. 20. Feb. 2025. URL: https://www.merriam-webster.com/dictionary/transitive (besucht am 2025-04-06) (siehe S. 12–17).

[11] 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. 197).

[12] Todd J. Green. "Conceptual Modeling using the Entity-Relationship Model". In: *ECS 165A Winter 2011 – Introduction to Database Systems*. Hrsg. von Todd J. Green. Davis, CA, USA: University of California, Davis, Winter 2011. Kap. 2. URL: https://web.cs.ucdavis.edu/~green/courses/ecs165a-w11/2-er.pdf (besucht am 2025-03-27) (siehe S. 27–32).

[13] Christian Heimes. "defusedxml 0.7.1: XML Bomb Protection for Python stdlib Modules". In: 8. März 2021. URL: https://pypi.org/project/defusedxml (besucht am 2024-12-15) (siehe S. 198).

[14] 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. 197).

[15] *IEEE Standard for Information Technology--Portable Operating System Interfaces (POSIX(TM))--Part 2: Shell and Utilities*. IEEE Std 1003.2-1992. New York, NY, USA: Institute of Electrical and Electronics Engineers (IEEE), 23. Juni 1993. URL: https://mirror.math.princeton.edu/pub/oldlinux/Linux.old/Ref-docs/POSIX/all.pdf (besucht am 2025-03-27). Board Approved: 1992-09-17, ANSI Approved: 1993-04-05. See unapproved draft IEEE P1003.2 Draft 11.2 of 9 1991 at the url (siehe S. 197).

[16] 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. 197).

[17] Katie Kodes. *Intro to XML, JSON, & YAML*. London, England, UK: Payhip, 2019–4. Sep. 2020 (siehe S. 5–10, 198).

[18] Andrew M. Kuchling. *Python 3 Documentation. Regular Expression HOWTO*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: https://docs.python.org/3/howto/regex.html (besucht am 2024-11-01) (siehe S. 197).

References III

[19] 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. 197).

[20] Mark Lutz. *Learning Python*. 6. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., März 2025. ISBN: 978-1-0981-7130-8 (siehe S. 197).

[21] *Machine Readable Travel Documents. Part 3: Specifications Common to all MRTDs. Eighth Edition*. Techn. Ber. Doc 9303. Montreal, QC, Canada: International Civil Aviation Organization (ICAO), 2021. URL: https://www.icao.int/publications/documents/9303_p3_cons_en.pdf (besucht am 2025-04-03) (siehe S. 145–157).

[22] Zsolt Nagy. *Regex Quick Syntax Reference: Understanding and Using Regular Expressions*. New York, NY, USA: Apress Media, LLC, Aug. 2018. ISBN: 978-1-4842-3876-9 (siehe S. 197).

[23] Thomas Nield. *An Introduction to Regular Expressions*. Sebastopol, CA, USA: O'Reilly Media, Inc., Juni 2019. ISBN: 978-1-4920-8255-2 (siehe S. 197).

[24] 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. 197).

[25] "POSIX Regular Expressions". In: *PostgreSQL Documentation*. 17.4. The PostgreSQL Global Development Group (PGDG), 20. Feb. 2025. Kap. 9.7.3. URL: https://www.postgresql.org/docs/17/functions-matching.html#FUNCTIONS-POSIX-REGEXP (besucht am 2025-02-27) (siehe S. 197).

[26] *PostgreSQL Essentials: Leveling Up Your Data Work*. Sebastopol, CA, USA: O'Reilly Media, Inc., März 2024 (siehe S. 197).

[27] "`re` – Regular Expression Operations". In: *Python 3 Documentation. The Python Standard Library*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: https://docs.python.org/3/library/re.html#module-re (besucht am 2024-11-01) (siehe S. 197).

[28] Yakov Shafranovich. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. Request for Comments (RFC) 4180. Wilmington, DE, USA: Internet Engineering Task Force (IETF), Okt. 2005. URL: https://www.ietf.org/rfc/rfc4180.txt (besucht am 2025-02-05) (siehe S. 5–10, 197).

[29] 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. 197).

References IV

[30] 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. 197).

[31] *Python 3 Documentation. The Python Standard Library*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: https://docs.python.org/3/library (besucht am 2025-04-27).

[32] 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. 27–32, 41–53, 197).

[33] 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. 12–17, 197).

[34] 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. 197).

[35] 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. 197).

[36] "中华人民共和国外国人工作许可证 (Work Permit for Foreigners of the People's Republic of China)". In: 百度百科 *(Baidu Baike)*. China, Beijing (中国北京市): Baidu (百度公司), 19.–22. Jan. 2025. URL: https://baike.baidu.com/item/%E4%B8%AD%E5%8D%8E%E4%BA%BA%E6%B0%91%E5%85%B1%E5%92%8C%E5%9B%BD%E5%A4%96%E5%9B%BD%E4%BA%BA%E5%B7%A5%E4%BD%9C%E8%AE%B8%E5%8F%AF%E8%AF%81 (besucht am 2025-04-03) (siehe S. 145–157).

References V

[37] 公民身份号码 *(Citizen Identification Number)*. 中华人民共和国国家标准 (National Standard of the People's Republic of China, GB) GB11643-1999. China, Beijing (中国北京市): 中华人民共和国国家质量监督检验检疫总局 (General Administration of Quality Supervision, Inspection and Quarantine of the People's Republic of China), 中国国家标准化管理委员会 (Standardization Administration of the People's Republic of China, SAC) und 中国标准出版社 (Standards Press of China), 19. Jan.–3. Nov. 1999. URL: https://openstd.samr.gov.cn/bzgk/gb/newGbInfo?hcno=080D6FBF2BB468F9007657F26D60013E (besucht am 2024-07-26) (siehe S. 145–155).

[38] 来华签证简介 *(Introduction to Chinese Visa)*. China, Beijing (中国北京市): 中华人民共和国外交部 (Ministry of Foreign Affairs of the People's Republic of China), 20. Nov. 2019. URL: http://cs.mfa.gov.cn/wgrlh/lhqz/lhqzjj_660596 (besucht am 2025-04-03) (siehe S. 145–157).

Glossary (in English) I

CSV *Comma-Separated Values* is a very common and simple text format for exchanging tabular or matrix data[28]. Each row in the text file represents one row in the table or matrix. The elements in the row are separated by a fixed delimiter, usually a comma („,“), sometimes a semicolon („;“). Python offers some out-of-the-box CSV support in the `csv` module[9].

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*[32].

DBMS A *database management system* is the software layer located between the user or application and the database (DB). The DBMS allows the user/application to create, read, write, update, delete, and otherwise manipulate the data in the DB[35].

ERD Entity relationship diagrams show the relationships between objects, e.g., between the tables in a DB and how they reference each other[1,3,5–7,16,29,34]

PostgreSQL An open source object-relational database management system (DBMS)[11,24,26,30]. See https://postgresql.org for more information.

Python The Python programming language[14,19,20,33], i.e., what you will learn about in our book[33]. Learn more at https://python.org.

regex A *Regular Expression*, often called „regex“ for short, is a sequence of characters that defines a search pattern for text strings[15,18,22,23]. In Python, the `re` module offers functionality work with regular expressions[18,27]. In PostgreSQL, regex-based pattern matching is supported as well[25].

WeChat 微信, produced by Tencent (腾讯公司) is the most-used messenger application in China. It integrates payment capabilities, video and voice chats, as well as social plugins and location-based services. See also https://weixin.qq.com/.

Glossary (in English) II

XML The *Extensible Markup Language* is a text-based language for storing and transporting of data[2,8,17]. It allows you to define elements in the form `<myElement myAttr="x">...text..</myElement>`. Different from comma-separated values (CSV), elements in XML can be hierarchically nested, like `<a><b><c>test</c></b><b>bla</b></a>`, and thus easily represent tree structures. XML is one of most-used data interchange formats. To process XML in Python, use the `defusedxml` library[13], as it protects against several security issues.