Prof. DI Dr. Erich Gams

Datenbanken

ER Modell Basics

informationssysteme htl-wels

Übersicht ☛ Was lernen wir?

- Kurze Wiederholung
- Entity-Relationship-Modell und Erweiterungen
- Beziehungen mit den verschiedenen Kardinalitäten
- Notationen
- …und natürlich Übungen

ER-Modell

- Konzeptionelles Modell – Abstrakte Modellierung der Daten der realen Welt (oder eines Ausschnitts)
- Zur Modellierung der konzeptuellen Ebene verwendet man das Entity-Relationship-Modell, welches einen Ausschnitt der Realwelt unter Verwendung von Entities und Relationships beschreibt :
- Das ER-Modell wurde 1976 von Peter Chen in seiner Veröffentlichung *The Entity-Relationship Model** vorgestellt.
- = Konzeptionelles Modell, das gegen Veränderungen der Funktionalität weitgehend stabil ist.

* (Peter Pin-Shan Chen: *The Entity-Relationship Model--Toward a Unified View of Data*. In: ACM Transactions on Database Systems 1/1/1976 ACM-Press ISSN 0362-5915, S. 9–36)

Beispiel Universität

- Datensammlungen vermitteln Informationen.
- Beispiele für Informationen aus der Universitätswelt:
  - Welche Vorlesungen werden für das 2. oder 4. Semester angeboten?
  - Welcher Prof liest im Wintersemester 20/21 die Rhetorik?
  - Wieviele Studierenden hören die Vorlesung Rhetorik?
  - Welche Studierenden haben die Mathematik-Prüfung abgelegt?
  - Auf welchen Vorlesungen baut die Datenbankvorlesung auf?

Beispiel Universität

- Information für Studenten: Welcher Professor hält welche Vorlesung?
- Grundlegende Daten:
  - eine Menge von Vorlesungen für das zweite Semester
  - eine Menge von Professoren



- Beziehung zwischen diesen beiden Mengen mit den folgenden Charakteristika:
  - eine Vorlesung wird von genau einem Professor gehalten
  - ein Professor kann mehrere Vorlesungen halten
  - jede Vorlesung muss einen verantwortlichen Professor haben
  - ein Professor kann, muss aber keine Vorlesung halten

Ausprägung und Modell

Echte Daten: Ausprägungen:

Eine einzelne Ausprägung:

Modell: Abstraktion des Zusammenhangs:

für alle Ausprägungen gilt ...

Entity und Relationship

- Entity
  - Repräsentiert ein Objekt der realen Anwendungswelt,
  - z.B. GDI ist eine Vorlesung
  - Graphische Darstellung:

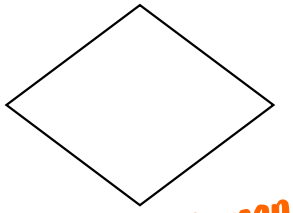
- Relationship
  - Repräsentiert eine Beziehung zwischen Entitäten
  - z.B. liest ist eine Beziehung zwischen Entitäten Rhetorik und Sokrates
  - Graphische Darstellung:

Wo findet man diese Begriffe
in der objektorientierten Welt wieder?

Entitätstyp und Relationstyp

- Entity Type
  - Definiert die Menge aller Entitäten einer Art
  - z.B. Vorlesung ist der Entity Type für alle Vorlesungs-Entitäten
  - Graphische Darstellung:

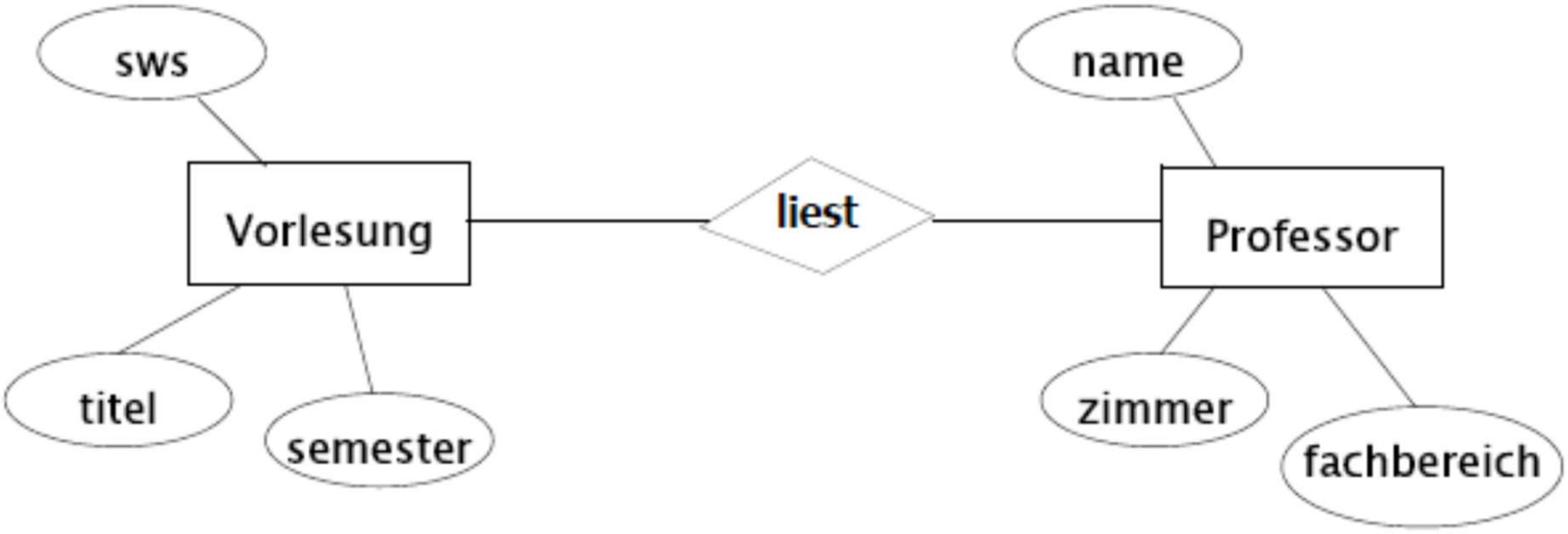
- Relationship Type
  - Definiert die Menge aller Beziehungen einer Art
  - z.B. liest ist der Relationship Type für eine Beziehung zwischen Vorlesungen und Professoren
  - Graphische Darstellung:



Wo findet man diese Begriffe in der objektorientierten Welt wieder?

Attribute von Entities

- Attribute
  - beschreiben die Objekte und Beziehungen genauer:



Attribute von Entities

- Beispiele (als Tabellen)

Vorlesung:

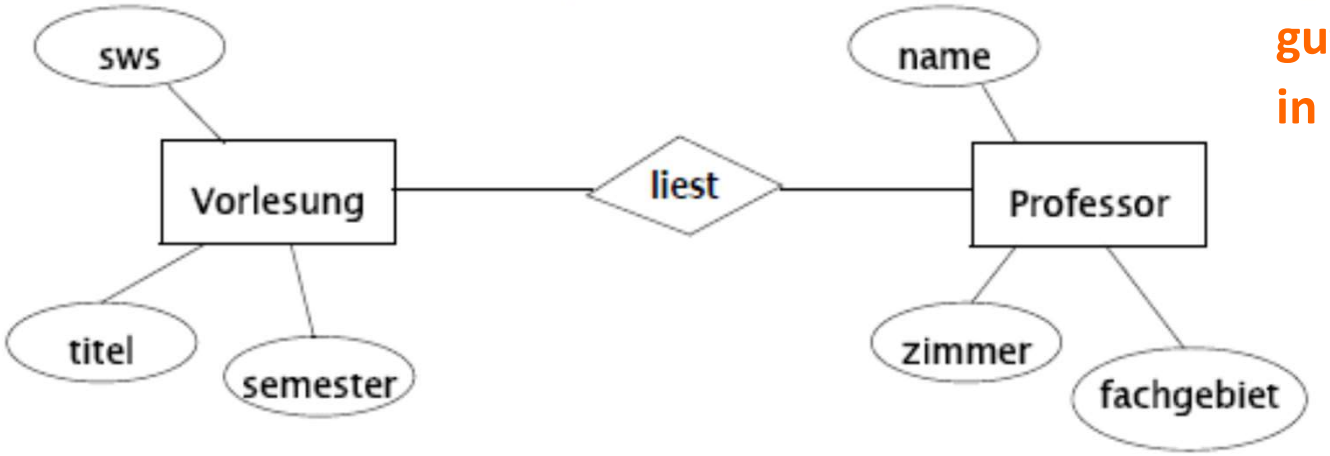
titel	semester	sws
PS1	SE2	6
Rhetorik	SE2	2

Professor:

name	zimmer	fachgebiet
Nitra	A417	Mathematik
Jaeger	E215	Informatik

Identität vs. Gleichheit: Schlüsselattribute

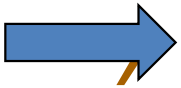
- Die einzelnen Entitäten sind eindeutig durch ihre Eigenschaften voneinander unterscheidbar.
- Es gibt immer eine Menge von Attributen, deren Wert jede Entität eindeutig bestimmen.
- Diese Attribute heißen Schlüsselattribute.



Welche Attribute sind gute Schlüsselkandidaten in unserem Beispiel?

Schlüsselkandidaten und Primärschlüssel

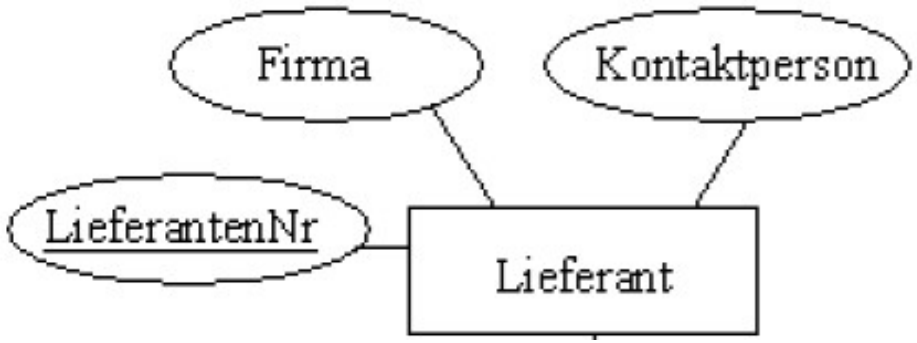
- Eine minimale(!) Menge von Attributen, welche das zugeordnete Entity eindeutig innerhalb aller Entities seines Typs identifiziert, nennt man Schlüsselkandidaten.
- Gibt es mehrere solcher Schlüsselkandidaten, wird einer als Primärschlüssel (Identifikationsschlüssel) ausgewählt.



- Mein gewählter Schlüsselkandidat heißt Primärschlüssel

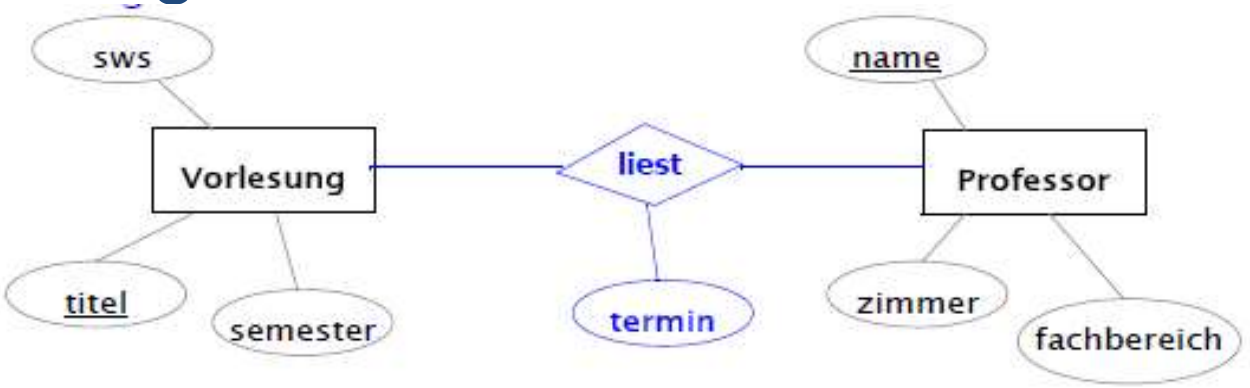
Primärschlüssel

- Tipp: Oft ist es sinnvoll künstlich eingeführte Attribute, wie z.B. Personalnummer (PersNr), als Primärschlüssel zu verwenden.
- Attribute des Primärschlüssel werden durch Unterstreichung gekennzeichnet.



Attribute von Relationships(types)

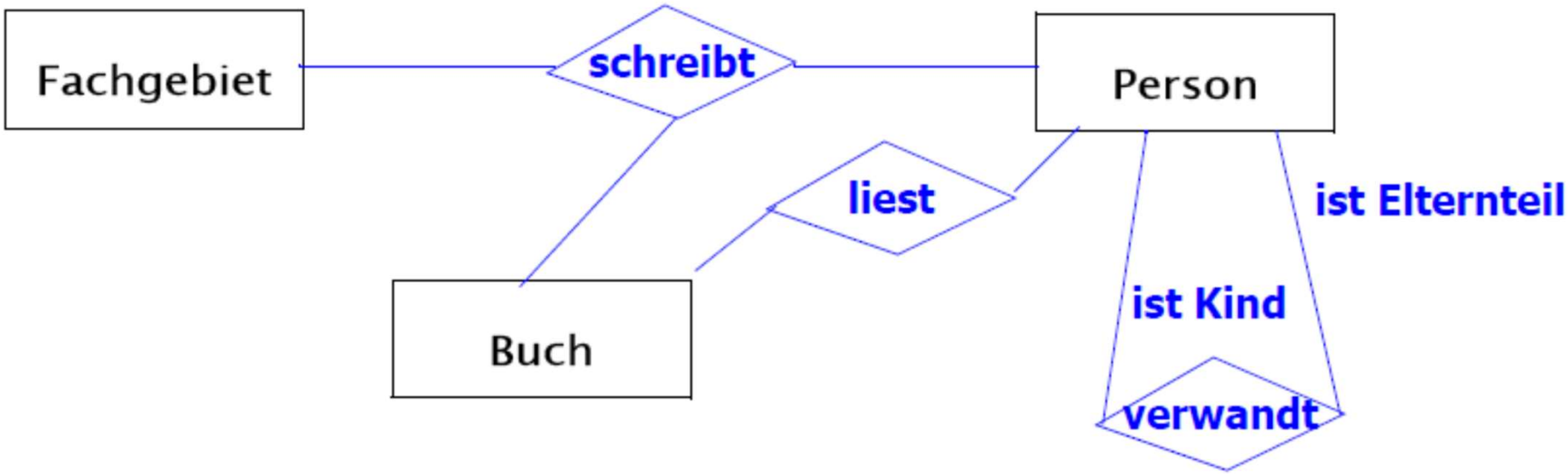
- Beziehungen haben auch Attribute



Beispiele als Tabelle liest:

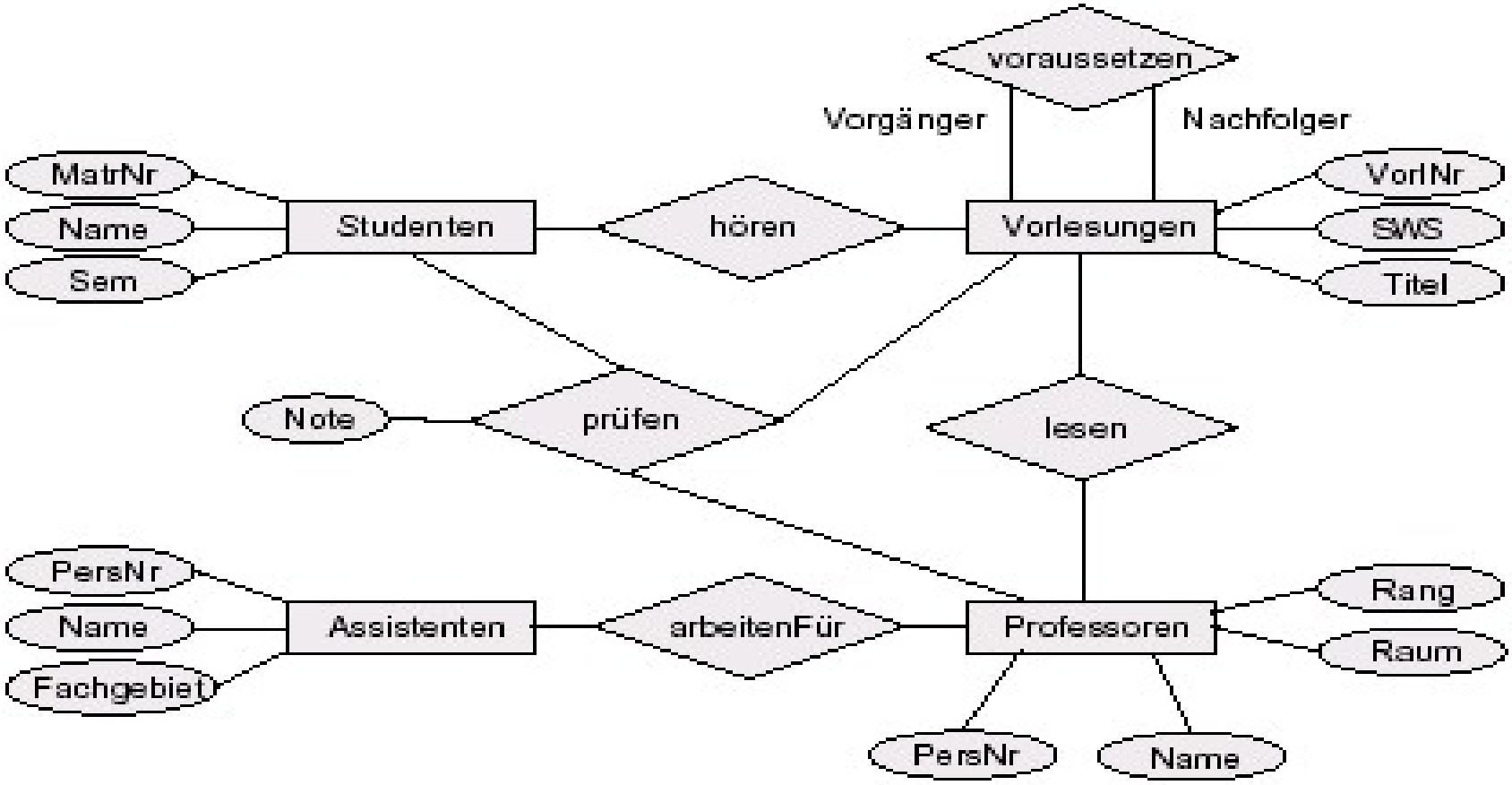
vorl	prof	termin
GDI	Winckler	WS01/02
DB1	Jaeger	WS02/03h
DB1	Sokrates	SS01
Rhetorik	Sokrates	SS02

Stelligkeit



- Beziehung schreibt ist eine dreistellige (ternäre) Beziehung
- Beziehung verwandt ist eine einstellige (unäre) Beziehung
- Beziehung liest ist eine zweistellige (binäre) Beziehung
- Häufigste Form: binäre Beziehungen

Ein einfaches Beispiel



Entnommen aus Kemper: Datenbanksysteme

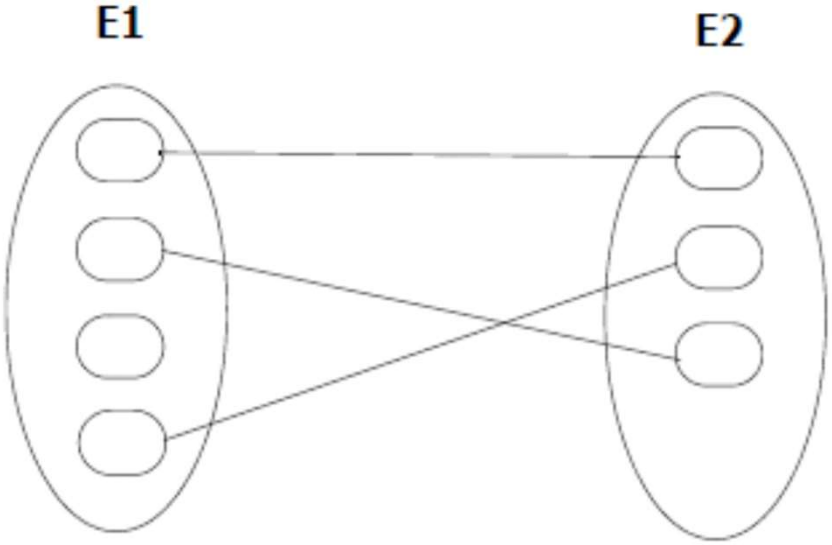
Charakteristika von Beziehungen

Kardinalität

- Beziehung liest zwischen den Mengen V und P ist eine N:1 Beziehung
  - liest (DB1) liefert {Jaeger}, stets genau ein Element.
  - liest (Jaeger) liefert {DB1, PS1}, mehrere Elemente.

Kardinalität

- Wir unterscheiden 1:1, 1:N, N:M Beziehungen:

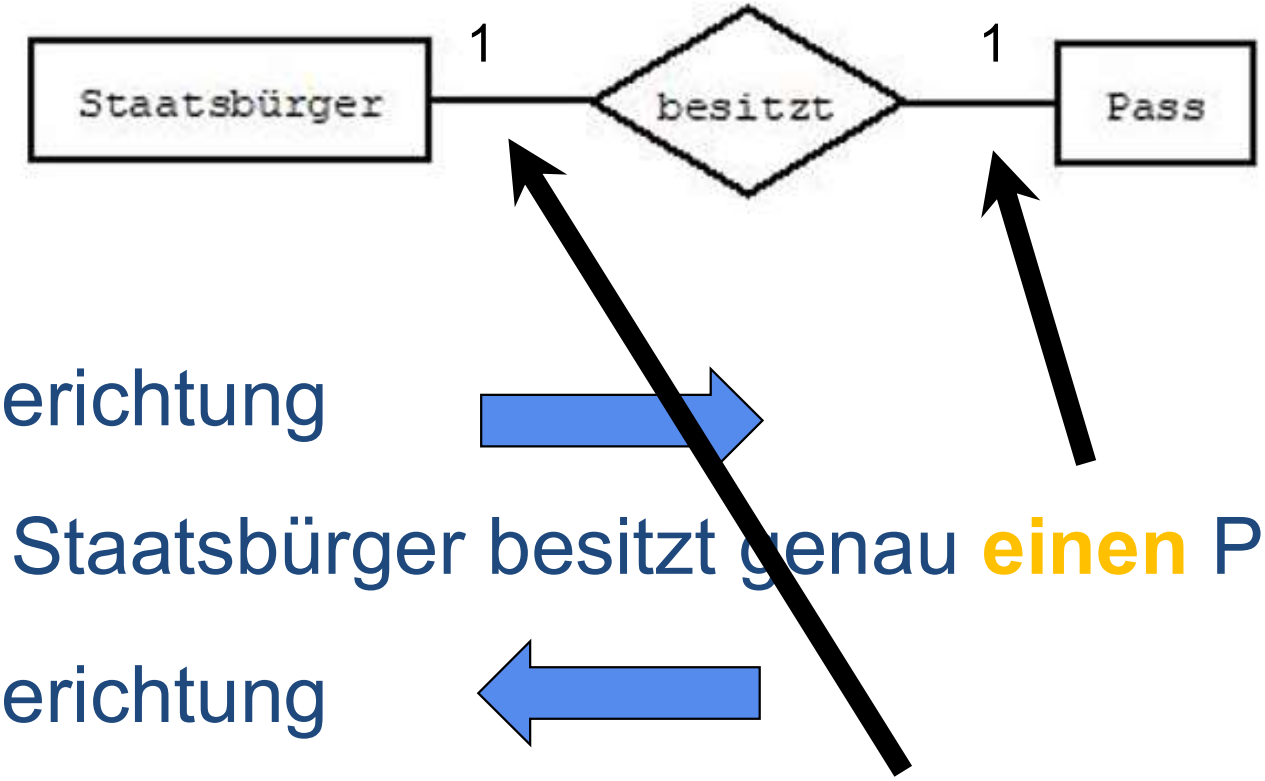


1.) Welche Art von Beziehung ist das?

2.) Zeichnen Sie die anderen Beziehungen nach demselben Muster

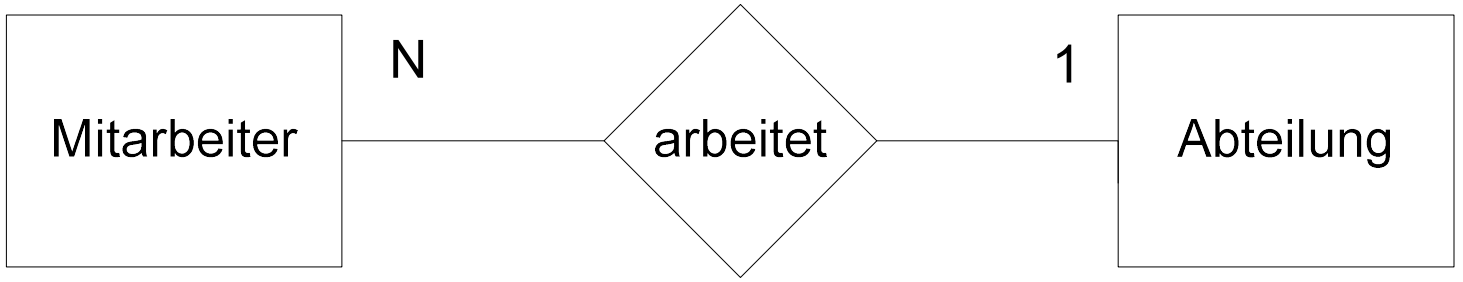
3.) Was ist der Unterschied zwischen N:1 und 1:N Beziehungen?

1:1 Beziehung



- Leserichtung
- Ein Staatsbürger besitzt genau **einen** Pass.
- Leserichtung
- Ein Pass gehört zu genau **einem** Staatsbürger.

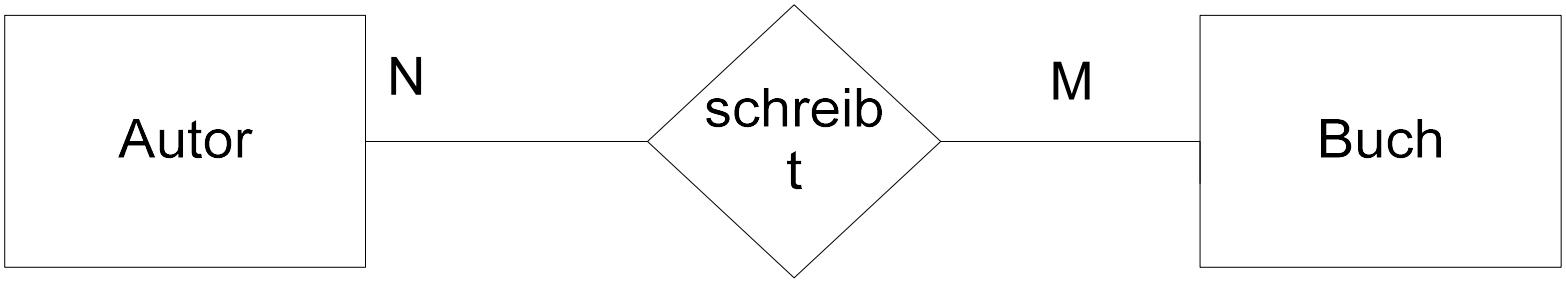
1:N bzw.N:1 Beziehung



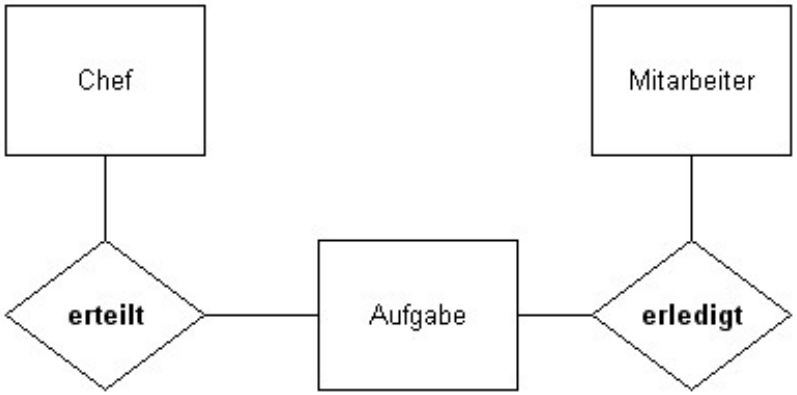
- Jeder Mitarbeiter gehört zu **einer** Abteilung.
- Eine Abteilung kann **mehrere** Mitarbeiter haben.

- Manche Autoren (z.b.: Zehnder) lassen sowohl bei der 1:N als auch bei der N:M Beziehung die Raute weg.

N:M



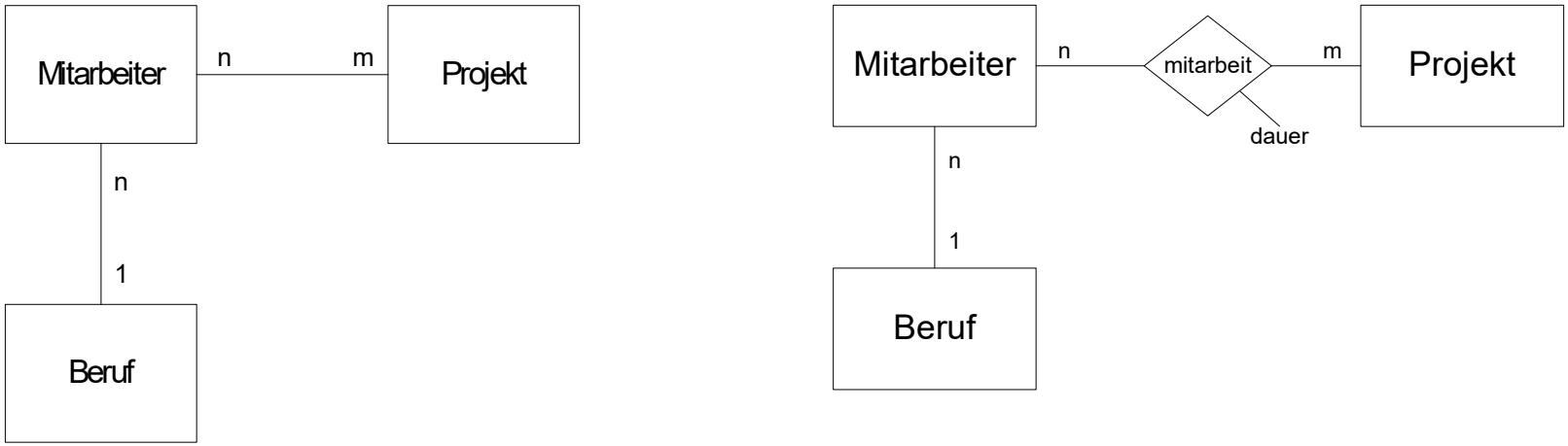
- Autoren können **mehrere (M)** Bücher schreiben.
- Bücher können **mehrere (N)** Autoren haben.



Setze die Kardinalitäten im folgenden Beispiel

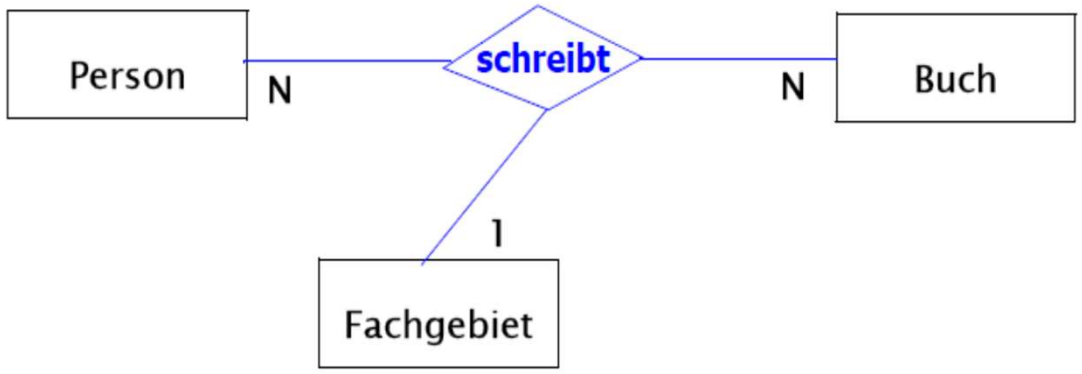
Beispiel: Mitarbeit

Beschreibe in Worten folgende Beziehungen!

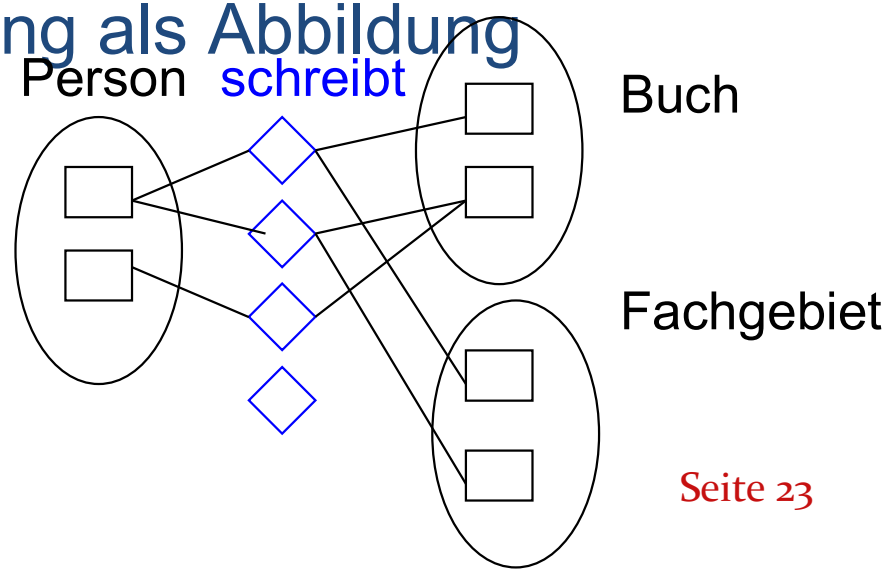


- Ein Mitarbeiter übt genau eine Funktion („Beruf“, z.b. Programmier) aus.
- Er kann an mehren Projekten beteiligt sein, wobei ein Projekt aus vielen Mitarbeitern bestehen kann.
- Er ist aber immer in seiner „Funktion“ an Projekten beteiligt.

Ternäre (dreistellige) Beziehungen



- Ein Buch zu einem bestimmten Fachgebiet wird von Personen geschrieben.
- Hilfreiche Vorstellung: Beziehung als Abbildung zwischen drei Mengen
- (Abbildung siehe Kemper)

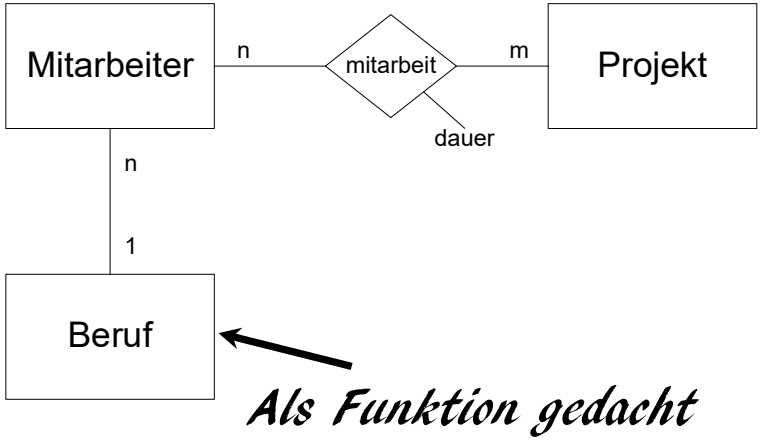


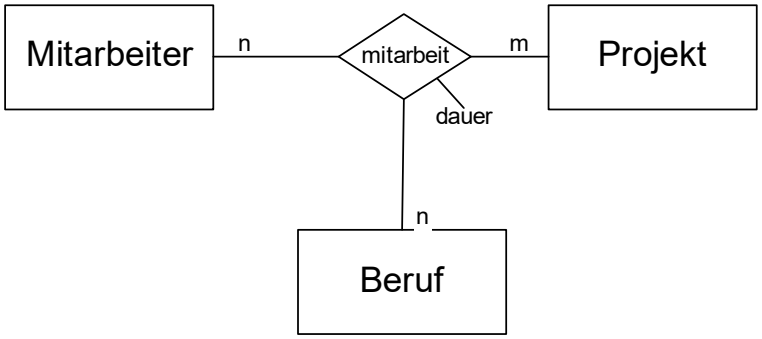
Ternäre (dreistellige) Beziehungen

- schreibt (Person, Fachgebiet): M Bücher
  - z.B. schreibt (Jaeger, Informatik) liefert zwei Bücher: {Java leicht gemacht, Dialogsteuerung}
- schreibt (Fachgebiet, Buch): N Autoren
  - z.B. schreibt (Informatik, Dialogsteuerung) liefert zwei Autoren: {Jaeger, Müller}
- schreibt (Person, Buch): 1 Fachgebiet
  - z.B. schreibt (Müller, Dialogsteuerung) liefert das Fachgebiet {Informatik}

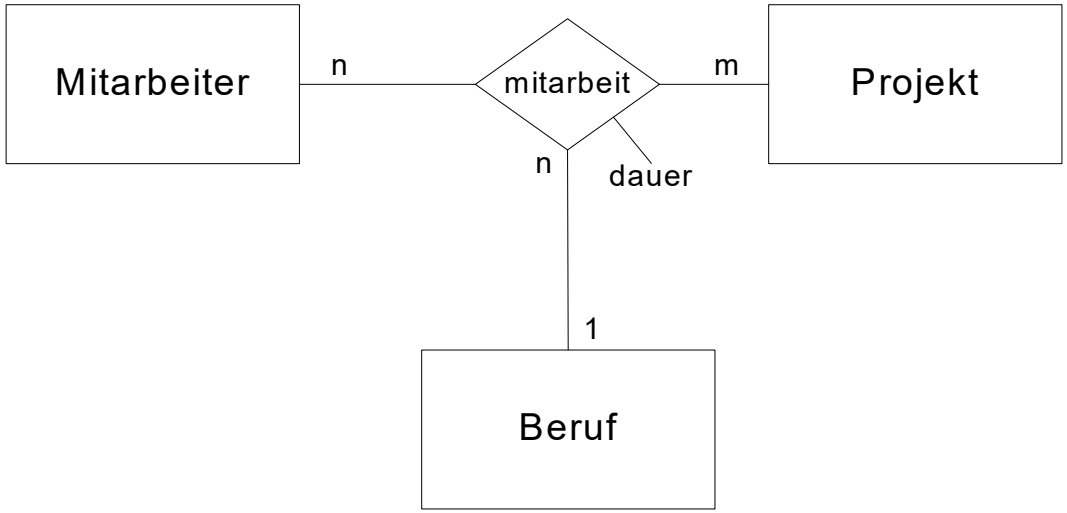
Ternäre (dreistellige) Beziehungen Beispiel

- Das Beispiel Mitarbeiter, soll derart geändert werden, dass ein Mitarbeiter in verschiedenen Projekten in **verschiedenen** Funktionen beteiligt ist. Er soll beispielsweise in einem Projekt sowohl als Programmierer als auch als Tester mitarbeiten können.



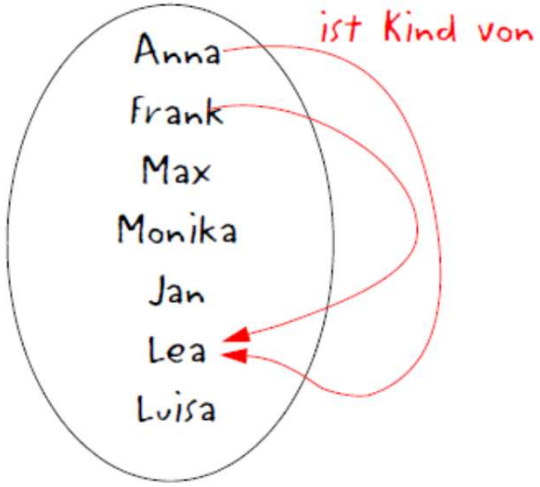


Ternäre (dreistellige) Beziehungen Beispiel



- Ein Mitarbeiter kann in einen Projekt nur einen „Beruf“ ausüben.
- Er kann aber in verschiedenen Projekten unterschiedliche Berufe(Funktionen) ausüben (d.h. Er hat mehrere „Berufe“, kann aber in einem Projekt immer nur einen ausüben)

Unäre Beziehungen



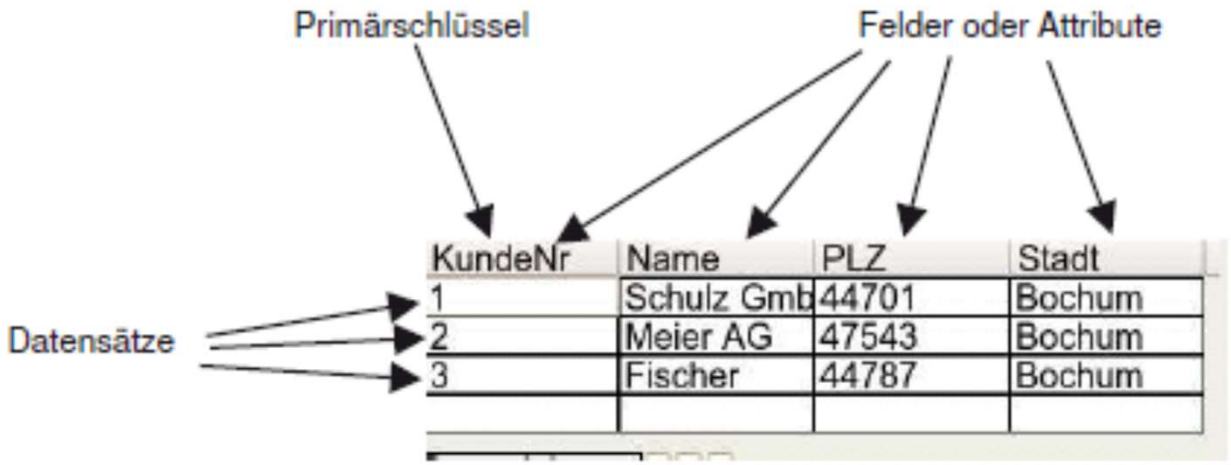
- Hier wichtig:Unterscheidung der Richtung (Rolle) der Beziehung:
- istElternteilVon(Anna) liefert: {Lea}
- istKindVon(Lea) liefert: {Anna, Frank}

Vorschau (Spoiler): Überführen in Tabellen

- Entitätstypen werden zu Tabellen:
  - Verschiedene Schreibweisen:

    Student: {[MatrNr:integer, *Name*: *string, Semester*: *integer*]}
    Student {[Matr#, *Name, Semester*]}
    Student {{Matr#, *Name, Semester*}}
  - Schlüssel werden unterstrichen
- Relationstypen werden auch zu Tabellen:
  - N:M Beziehung: Neue Tabelle mit den Schlüsseln der beteiligten Entitätstypen
  - 1:N bzw. N:1 Beziehung: Keine neue Tabelle, Schlüssel wandert von 1 zu N (wird als Fremdschlüssel bezeichnet)

Tabellen, Datensätze, Schlüssel



KundeNr	Name	PLZ	Stadt
1	Schulz Gmb	44701	Bochum
2	Meier AG	47543	Bochum
3	Fischer	44787	Bochum

Tabelle Kunde

Fremdschlüssel

Primärschlüssel

AuftragNr	AuftragDatum	AuftragBeschreibung	KundeNr
1	21.01.2005	Werkhalle streichen	1
2	22.01.2005	Teppichboden verlegen	2
3	24.01.2005	Laminat verlegen	1
4	27.01.2005	Tapezieren	3
5	01.02.2005	Werkhalle Boden verlegen	1
6	01.02.2005	Laminat verlegen	2

Tabelle Auftrag

Aufgabe

- Ergänze das Universitätsbeispiel um Kardinalitäten.
- Übungen in Moodle

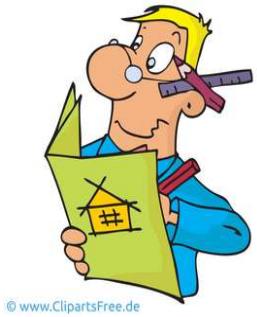
Auf los geht's los :-)

Aufgabe

- Modellierungsaufgabe:
  - Zeichne (händisch ein ER-Modell)
- Notationsarten:
  - Finde die verschiedenen Notationsarten und vergleiche sie?
- ER Modellierungstools
  - Welche Notation wird unterstützt?
  - Vor- Nachteile
  - Modelliere ein persönliches Beispiel

(CLIL)Task

- **Independent user view (data independence)**
  - 1) Try to describe the architecture of a database management system in your own words.
  - 2) What is data independence and how is it achieved?
  - 3) Complete your discussion with one or two examples from your „computer science life“ that demonstrate the benefits of data independence



Aufgabe

Übung zur Datenunabhängigkeit

- 1) Versuchen Sie in eigenen Worten die Architektur eines Datenbankmanagement-Systems zu beschreiben.
- 2) Was ist Datenunabhängigkeit und wie wird sie erreicht?
- 3) Ergänzen Sie Ihre Diskussion wenn möglich mit ein oder zwei Beispielen aus Ihrem bisherigen Informatikalltag, die den Nutzen der Datenunabhängigkeit aufzeigen.

Aufgaben eines Datenbankadministrators

- die **Gestaltung der konzeptionellen Ebene**. Man nennt eine einmal gestaltete konzeptionelle
- Ebene auch das **konzeptionelle Schema**. Eine andere Bezeichnung für
- diese Aufgabe, auf die Sie auch oft stoßen werden, ist das **logische Datenbankdesign**.
- Die **Gestaltung der internen Ebene** oder der Aufbau des **internen Schemas**. Hier
- ist die entsprechende Bezeichnung das **physikalische Datenbankdesign**.
- Die **Unterstützung der Benutzer** bei der Anlage der Datenbanken und Tabellen, die
- sie für ihre Zwecke benötigen. Oft wird diese Arbeit insgesamt von dem Datenbankadministrator
- übernommen.
- Die **Definition von Regeln**, die die **Integrität des Datenbestandes** sichern. Was
- dazu gehört und wie dieser Begriff genau zu verstehen ist, wird im Folgenden in den
- Kapiteln 5 und 8 noch erläutert.
- Die Festlegung und Durchführung von regelmäßigen **Sicherungsprozeduren** zur
- Sicherung des gesamten Datenbestandes (oft auf einem anderen Speichermedium).
- Regelmäßige **Reorganisation** des Datenbestandes, um zu verhindern, dass eine zu
- große physikalische Aufspaltung des Datenbestandes auf dem Speichermedium zu
- Performanceproblemen führt. (Schubert)

Eigenschaften von Datenbanken

- **Mehrfachnutzung**
  - Ein Datenbanksystem erlaubt mehreren Benutzern gleichzeitig den Zugriff auf eine Datenbank.
- **Datensichten (Views)**
  - Typischerweise hat eine Datenbank mehrere Benutzer, und jeder benötigt, je nach Zugangsrechten und Bedürfnissen, eine individuelle Ansicht der Daten (Inhalt und Form). Eine solche Datensicht kann aus einer Teilmenge der gespeicherten Daten oder aus daraus abgeleiteten (nicht explizit gespeicherten) Daten bestehen.
- **Strukturierte und beschriebene Daten**
  - Eine grundsätzliche Eigenschaft des Datenbankansatzes ist, dass ein Datenbanksystem nicht nur die Daten enthält, sondern auch eine komplette Definition oder Beschreibung der Daten (Metadaten). Diese Beschreibung besteht aus Angaben über den Umfang, die Struktur, die Art und das Format aller Daten.

Eigenschaften von Datenbanken

- **Trennung von Daten und Anwendungen**

- Wie in der Eigenschaft „Strukturierte Daten“ beschrieben, wird die Struktur einer Datenbank durch die Metadaten, die ebenfalls in der Datenbank abgelegt sind, beschrieben.

  Das Anwendungsprogamm benötigt keine Kenntnis über die physikalische Datenspeicherung (Codierung, Format, Speicherort etc.). Es kommuniziert mit dem Verwaltungssystem einer Datenbank (DBMS) über eine normierte Schnittstelle mittels einer standardisierten Sprache (z. B. SQL). Den Zugriff auf die eigentlichen Daten und die Metadaten übernimmt dabei das DBMS.
  Datenintegrität

- Datenintegrität ist ein Begriff für die Qualität und Zuverlässigkeit von Daten eines Datenbanksystems. Im weiteren Sinne zählt zur Integrität auch der Schutz der Datenbank vor unberechtigtem Zugriff (Vertraulichkeit) und Veränderungen.

  Daten widerspiegeln Sachverhalte der realen Welt. Logischerweise wird verlangt, dass sie dies korrekt tun. Ein DBMS soll Unterstützung bieten bei der Aufgabe, nur korrekte und widerspruchsfreie („konsistente“) Daten in die Datenbank gelangen zu lassen. Ausserdem wird mit korrekten Transaktionen die Konsistenz auch während des Systembetriebs aufrechterhalten.

Eigenschaften von Datenbanken

- **Transaktionen**
- Eine Transaktion ist ein Bündel von Aktionen, die in der Datenbank durchgeführt werden, um diese von einem konsistenten Zustand wieder in einen konsistenten (widerspruchsfreien) Zustand zu überführen. Dazwischen sind die Daten zum Teil zwangsläufig inkonsistent. Eine Transaktion ist atomar, d. h. nicht weiter zerlegbar. Innerhalb einer Transaktion werden entweder alle Aktionen oder keine durchgeführt. Nur ein Teil der Aktionen würde zu einem inkonsistenten Datenbankzustand führen.
- **Datenpersistenz**
- Datenpersistenz meint, dass in einem DBMS einzelne Daten solange aufbewahrt werden müssen, bis sie explizit gelöscht werden. Die Lebensdauer von Daten muss also von den Benutzern direkt oder indirekt bestimmbar sein und darf nicht von irgendwelchen Systemgegebenheiten abhängen. Ebensowenig dürfen einmal in die Datenbank aufgenommene Daten verloren gehen.

Eigenschaften von Datenbanken

- Aus (http://www.gitta.info/IntroToDBS/de/html/unit_DBApproaChar.html)
- Aufgabe:
- Suchen Sie selber eine weitere Datenbank-Anwendung (in Ihrem Umfeld, im Internet, ...) und versuchen Sie abzuschätzen, welches speziell wichtige / speziell geforderte Eigenschaften für diese Anwendung sind.
  Verfassen Sie eine kurze Zusammenstellung der Informationen

Vorteile DBMS

- In einem globalen, integrierten DBMS werden unkontrollierte Redundanz und Inkonsistenz vermieden (Achtung: kontrollierte Redundanz kann manchmal erwünscht sein)
- Zugriffsmöglichkeiten: Daten werden miteinander verknüpft und in Beziehung gebracht
- Mehrbenutzerbetrieb (Nebenläufigkeitskontrolle)
- Datensicherung
- Integritätsverletzungen (Transaktionen), Einschränkungen bez. Der Daten die gespeichert werden
- Sicherheit
- Entwicklungskosten (Flexibilität bei Veränderungen)

Vorteile DBMS

- **Redundanz**:
  Dieselben Informationen werden doppelt gespeichert.
- **Inkonsistenz**:
  Dieselben Informationen werden in unterschiedlichen Versionen gespeichert.
- **Integritätsverletzung**:
  Die Einhaltung komplexer Integritätsbedingungen fällt schwer.
- **Verknüpfungseinschränkung**:
  Logisch verwandte Daten sind schwer zu verknüpfen, wenn sie in isolierten Dateien liegen.
- **Mehrbenutzerprobleme**:
  Gleichzeitiges Editieren derselben Datei führt zu Anomalien (lost update).
- **Verlust von Daten**:
  Außer einem kompletten Backup ist kein Recoverymechanismus vorhanden.
- **Sicherheitsprobleme**:
  Abgestufte Zugriffsrechte k¨onnen nicht implementiert werden.
- **Hohe Entwicklungskosten**:
  Für jedes Anwendungsprogramm müssen die Fragen zur Dateiverwaltung erneut gelöst werden.

Auf los geht's los ;-)