



Script generated by TTT

Title: Einf_HF (14.04.2014)

Date: Mon Apr 14 14:14:31 CEST 2014

Duration: 92:31 min

Pages: 33

Informationsgesellschaft führt zu immer größeren Datenmengen ⇒ Speicherung der Daten in Datenbanksystemen.

Allgemeines

Anwendungsübergreifende Datenspeicherung und -organisation.

Datenbank = Datenmenge mit Strukturmodell (Datenbankschema, Datenmodell)

"Datenbanksystem" (DBS): Sammlung gespeicherter Daten (Datenbank), Software für Speicherung und zum Zugriff.

Operationen: Eintragen, Löschen, Suchen, Verknüpfen von Daten. Wichtig: Daten mit langer Lebensdauer, große Datenmengen (GBytes, TBytes).

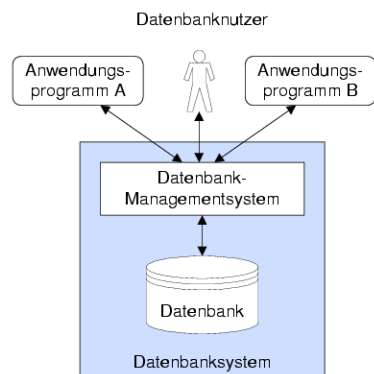
Logischer Aufbau

Datenbankmanagementsystem

Beispiele aus der Praxis

Anforderungen an ein DBS

Generated by Targeteam



Generated by Targeteam

Das Datenbankmanagementsystem (DBMS) ist die Gesamtheit aller Programme für den Umgang mit den Daten. Es ist verantwortlich für

- die sichere und einheitliche Verwaltung persistenter (langlebiger) Daten,
- den Datenaustausch zwischen Datenbank und Anwendungsprogrammen,
- die Verhinderung von unkontrollierten Zugriffen auf den Datenbestand und
- die effiziente Zugriffsmöglichkeit auf die in der Regel sehr großen Datenbestände.

Generated by Targeteam



Universitätsdatenbank

Sammlung aller für Verwaltungsaufgaben an einer Universität benötigten Daten. I.a. Gliederung in Fachbereiche mit zugeordneten Studenten, Professoren, Mitarbeitern. Studenten belegen Vorlesungen von Professoren, legen Prüfungen ab. Typische Anwendungen: Immatrikulation, Rückmeldung, Ausfertigen von Studentenausweisen, Studienbescheinigungen, Stundenplänen. Raumplanung, Ausstellen von Zeugnissen, Exmatrikulation, Statistiken.

Datenbank einer Fluggesellschaft

Fluggesellschaft fliegt verschiedene Flughäfen an. Flugstrecken, Flugzeugtypen, Personal. Piloten haben Flugscheine jeweils nur für einige wenige Flugzeugtypen. Typische Anwendungen: Flugbuchungen, Anfertigen Passagierlisten, Personaleinsatzplanung, Materialeinsatzplanung, Flugplanerstellung, Überwachung der Wartefristen, Gehaltsabrechnung.

Generated by Targeteam



Kontrolle über Daten

Kontrolle der Datenintegrität

Automatisierte Zugriffskontrollen (Datenschutz).

Erhaltung der logischen Datenintegrität, d.h. Überprüfung der Richtigkeit von Daten.

Notwendigkeit des kontrollierten Mehrbenutzer-Betriebs, keine gegenseitige Störung durch parallel zugreifenden Benutzer.

Leichte Handhabbarkeit der Daten

Einfache Beschreibung der logische Aspekte der Daten (Bedeutung, Zusammenhänge), leicht erlernbare Sprache zum Zugriff bzw. Speichern von Daten in der Datenbank (z.B. SQL).

Datenunabhängigkeit

Anwendungen unabhängig von Organisation und Speicherungen der Daten.

Generated by Targeteam



Alle Daten können/müssen gemeinsam benutzt werden

keine verstreuten privaten Daten

Entwicklung neuer Anwendungen auf der existierenden Datenbank

Querauswertungen aufgrund inhaltlicher Zusammenhänge

Erweiterung und Anpassung der Datenbank

Eliminierung der Redundanz, damit Vermeidung von Inkonsistenzen

Datenbankadministrator hat zentrale Verantwortung für Daten

Generated by Targeteam

Kontrolle über Daten

Kontrolle der Datenintegrität

Automatisierte Zugriffskontrollen (Datenschutz).

Erhaltung der logischen Datenintegrität, d.h. Überprüfung der Richtigkeit von Daten.

Notwendigkeit des kontrollierten Mehrbenutzer-Betriebs, keine gegenseitige Störung durch parallel zugreifenden Benutzer.

Leichte Handhabbarkeit der Daten

Einfache Beschreibung der logische Aspekte der Daten (Bedeutung, Zusammenhänge), leicht erlernbare Sprache zum Zugriff bzw. Speichern von Daten in der Datenbank (z.B. SQL).

Datenunabhängigkeit

Anwendungen unabhängig von Organisation und Speicherungen der Daten.

Generated by Targeteam



- Fragestellungen des Abschnitts:
 - Was unterscheidet Dateisysteme von Datenbanksystemen?
 - Wie kann die Struktur der Daten in einem Datenbanksystem dargestellt werden?
 - Was sind relationale Datenbanksysteme?
 - Was sind die grundlegenden Konstrukte von HTML?

[Dateisysteme](#)

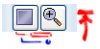
[Datenbanksysteme](#)

[Datenbankentwurf](#)

[Relationale Datenbanksysteme](#)

[WWW - Informationssystem](#)

Generated by Targeteam



Die Daten in einem Datenbanksystem bilden ein abstrahiertes Spiegelbild einer Miniwelt. ⇒ Verwendung von Datenmodellen.

Definition: Ein **Datenmodell** ist ein (oft mathematischer) Formalismus

mit einer Notation zur Beschreibung und Definition der Datenobjekte und deren Struktur,
einer Menge von Operationen zur Manipulation der Daten.

Nicht beschrieben werden Abläufe, Interaktion zwischen Objekten oder zeitliches Verhalten.

Beispiele

Entity-Relationship-Modell (ER-Modell), Objektorientiertes Modell, Hierachisches Modell.

Generated by Targeteam



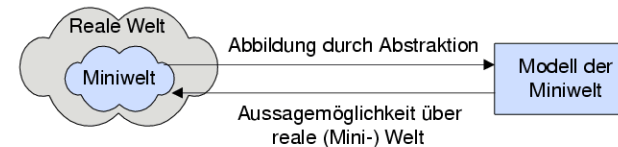
Man geht schrittweise vor

1. Erstellung eines Modells der Daten und der Beziehungen zwischen diesen Daten ⇒ Datenmodell, z.B. ER-Modell
2. Abbildung des Datenmodells auf eine Menge von Tabellen (relationales Modell)
3. Normalisierung: Entfernen aller Redundanzen. Eine Redundanz liegt vor, wenn dieselbe Information an mehreren Stellen eingetragen ist, z.B. die PLZ in verschiedenen Tabellen der Datenbank. Bei einer Änderung der PLZ müssen alle Stellen in der Datenbank geändert werden; bei Vergessen einer Änderungsstelle können Fehler auftreten.
4. Eintragen der Daten in die Tabellen.
5. Realisierung der möglichen Abfragen ⇒ Sichten auf die Daten in den Tabellen.

Generated by Targeteam



Daten in einem Datenbanksystem sind das Spiegelbild einer Situation der realen Welt ⇒ Vereinfachung durch Abstraktion.



Miniwelt

In der Regel reicht es, nur einen Ausschnitt der realen Welt, die sogenannte Miniwelt, zu betrachten.

Durch Vernachlässigung unnötiger Informationen wird diese Miniwelt abstrahiert (vereinfacht).

[Wie entwirft man eine Datenbank?](#)

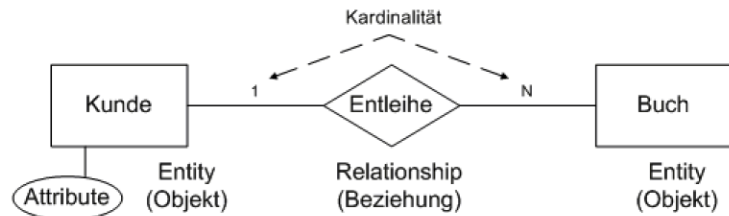
[Datenmodell](#)

[ER-Modell](#)

Generated by Targeteam



Entity-Relationship-Modell eignet sich zur Darstellung des Datenbankschemas.



Graphisches Hilfsmittel zur semantischen Modellierung eines Anwendungsgebietes, d.h. zum Entwurf einer Datenbank, unabhängig vom konkreten DBS.

Grundidee: Reale Welt (Mini-Welt) lässt sich durch Objekte und Beziehungen zwischen Objekten beschreiben (Objekte: Entities, Beziehungen: Relationships).

Gleichartige Entities (Objektinstanzen) werden zu Entity-Typen (vergleichbar Klassen) bzw. Relationships zu Relationship-Typen zusammengefasst.

Beispiele: Entity-Typ: "Bibliotheksbenuer", "Buch", und Relationship-Typ: "entleiht".

Attribute bei Entities und Relationships, z.B. "Name" bei Entity "Bibliotheksbenuer", "Entleihdatum" bei Relationship "leiht aus".

Entity-Typ

Relationship-Typ

Erweiterung auf n-stellige Beziehungen

Modellierung einer Datenbank findet auf der Ebene von Entity-Typen, Relationship-Typen und Attributen



Wahl eines oder mehrerer Attribute, die eine Entity (Objektinstanz) **eindeutig** identifiziert, z.B.

Kundenr identifiziert eindeutig einen einzelnen Kunden.

⇒ Primärschlüssel.

Ein Primärschlüssel kann aus mehr als einem Attribut bestehen.

Die Entscheidung für einen bestimmten Primärschlüssel geschieht während der Modellierung des Anwendungsbereiches.

falls kein Attribut eindeutig ist, wird ein **künstliches** Attribut eingeführt, z.B. eine fortlaufende Nummer für jeden Eintrag.

Primärschlüssel werden im ER-Modell durch Unterstreichnung gekennzeichnet.

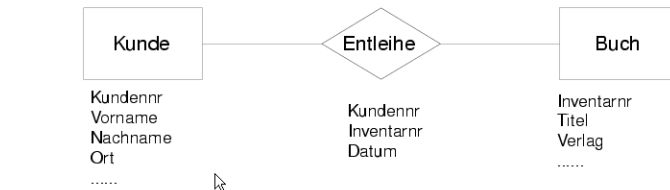
Generated by Targeteam



Ein Entity-Typ fasst eine Menge von gleichartigen Objektinstanzen, die durch gleiche Attribute charakterisiert sind, zusammen.



Abstraktion



Schlüssel

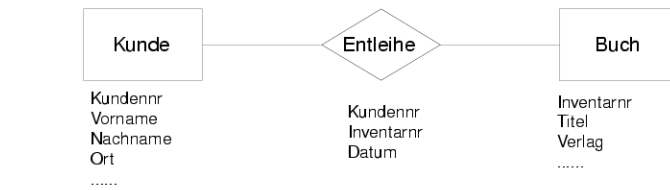
Generated by Targeteam



Ein Entity-Typ fasst eine Menge von gleichartigen Objektinstanzen, die durch gleiche Attribute charakterisiert sind, zusammen.



Abstraktion



Schlüssel

Generated by Targeteam



Schlüssel



Wahl eines oder mehrerer Attribute, die eine Entity (Objektinstanz) **eindeutig** identifiziert, z.B.

Kundenr identifiziert eindeutig einen einzelnen Kunden.

⇒ Primärschlüssel.

Ein Primärschlüssel kann aus mehr als einem Attribut bestehen.

Die Entscheidung für einen bestimmten Primärschlüssel geschieht während der Modellierung des Anwendungsbereiches.

falls kein Attribut eindeutig ist, wird ein **künstliches** Attribut eingeführt, z.B. eine fortlaufende Nummer für jeden Eintrag.

Primärschlüssel werden im ER-Modell durch Unterstreichung gekennzeichnet.

Generated by Targeteam



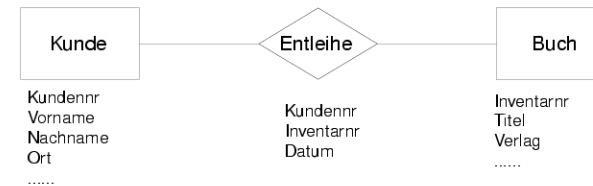
Entity-Typ



Ein Entity-Typ fasst eine Menge von gleichartigen Objektinstanzen, die durch gleiche Attribute charakterisiert sind, zusammen.



Abstraktion



Schlüssel

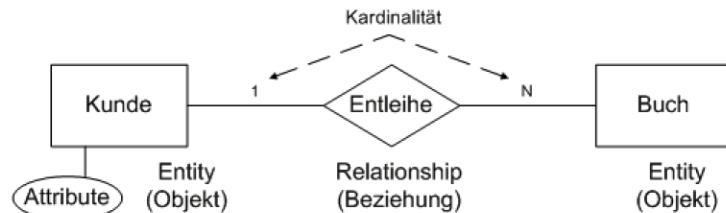
Generated by Targeteam



ER-Modell



Entity-Relationship-Modell eignet sich zur Darstellung des Datenbankschemas.



Graphisches Hilfsmittel zur semantischen Modellierung eines Anwendungsgebietes, d.h. zum Entwurf einer Datenbank, unabhängig vom konkreten DBS.

Grundidee: Reale Welt (Mini-Welt) lässt sich durch Objekte und Beziehungen zwischen Objekten beschreiben (Objekte: Entities, Beziehungen: Relationships).

Gleichartige Entities (Objektinstanzen) werden zu Entity-Typen (vergleichbar Klassen) bzw. Relationships zu Relationship-Typen zusammengefasst.

Beispiele: Entity-Typ: "Bibliotheksbenuer", "Buch", und Relationship-Typ: "entleiht".

Attribute bei Entities und Relationships, z.B. "Name" bei Entity "Bibliotheksbenuer", "Entleihdatum" bei Relationship "leiht aus".

Entity-Typ

Relationship-Typ

Erweiterung auf n-stellige Beziehungen

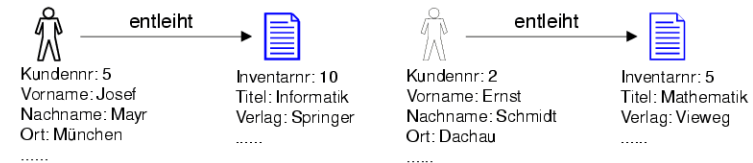
Modellierung einer Datenbank findet auf der Ebene von Entity-Typen, Relationship-Typen und Attributen



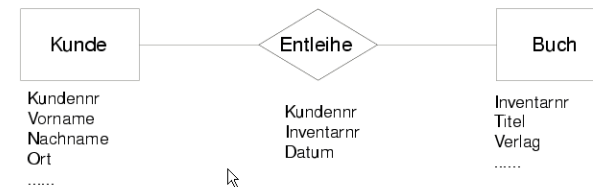
Entity-Typ



Ein Entity-Typ fasst eine Menge von gleichartigen Objektinstanzen, die durch gleiche Attribute charakterisiert sind, zusammen.



Abstraktion



Schlüssel

Generated by Targeteam



Ein Relationship-Typ umfasst die Menge gleichartiger Relationships. Ein Relationship-Typ R stellt die Beziehung zwischen Entity-Typen E1 und E2 her, d.h. $R \subseteq E1 * E2$.

Kardinalität von Relationship-Typen zur Spezifikation der Art der Beziehung zwischen Entities.

Man kann verschiedene Kardinalitätsarten unterscheiden

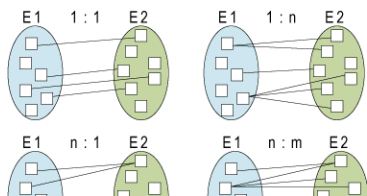
1:1, ein Entity aus E1 kann mit höchstens einem Entity aus E2 über R in Beziehung stehen kann und umgekehrt.

n:1, ein Entity aus E1 kann mit höchstens einem Entity aus E2, aber ein Entity aus E2 mit beliebig vielen Entities aus E1 über R in Beziehung stehen.

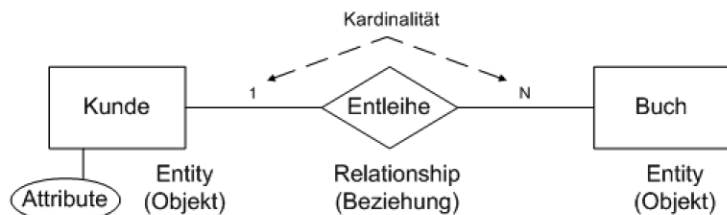
1:n, ein Entity aus E1 kann mit beliebig vielen Entities aus E2, aber ein Entity aus E2 mit höchstens einem Entity aus E1 über R in Beziehung stehen.

Beispiel: 1 Kunde kann n Bücher ausleihen; 1 Buch kann nur von einem Kunden ausgeliehen werden.

n:m, ein Entity aus E1 kann mit beliebig vielen Entities aus E2 über R in Beziehung stehen und umgekehrt.



Entity-Relationship-Modell eignet sich zur Darstellung des Datenbankschemas.



Graphisches Hilfsmittel zur semantischen Modellierung eines Anwendungsgebietes, d.h. zum Entwurf einer Datenbank, unabhängig vom konkreten DBS.

Grundidee: Reale Welt (Mini-Welt) lässt sich durch Objekte und Beziehungen zwischen Objekten beschreiben (Objekte: Entities, Beziehungen: Relationships).

Gleichartige Entities (Objektinstanzen) werden zu Entity-Typen (vergleichbar Klassen) bzw. Relationships zu Relationship-Typen zusammengefasst.

Beispiele: Entity-Typ: "Bibliotheksbenuzer", "Buch", und Relationship-Typ: "entleiht".

Attribute bei Entities und Relationships, z.B. "Name" bei Entity "Bibliotheksbenuzer", "Entleihdatum" bei Relationship "leiht aus".

Entity-Typ

Relationship-Typ

Erweiterung auf n-stellige Beziehungen

Modellierung einer Datenbank findet auf der Ebene von Entity-Typen, Relationship-Typen und Attributen



Kardinalität von Relationship-Typen zur Spezifikation der Art der Beziehung zwischen Entities.

Man kann verschiedene Kardinalitätsarten unterscheiden

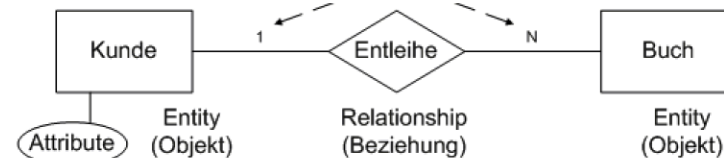
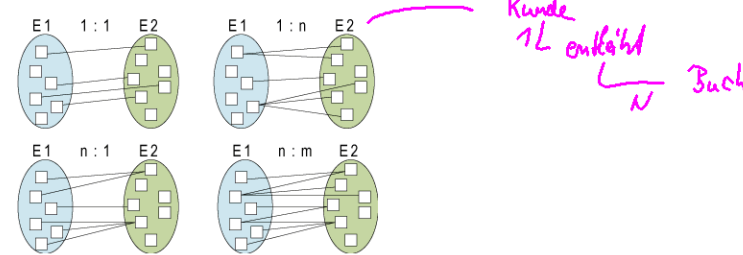
1:1, ein Entity aus E1 kann mit höchstens einem Entity aus E2 über R in Beziehung stehen kann und umgekehrt.

n:1, ein Entity aus E1 kann mit höchstens einem Entity aus E2, aber ein Entity aus E2 mit beliebig vielen Entities aus E1 über R in Beziehung stehen.

1:n, ein Entity aus E1 kann mit beliebig vielen Entities aus E2, aber ein Entity aus E2 mit höchstens einem Entity aus E1 über R in Beziehung stehen.

Beispiel: 1 Kunde kann n Bücher ausleihen; 1 Buch kann nur von einem Kunden ausgeliehen werden.

n:m, ein Entity aus E1 kann mit beliebig vielen Entities aus E2 über R in Beziehung stehen und umgekehrt.



Graphisches Hilfsmittel zur semantischen Modellierung eines Anwendungsgebietes, d.h. zum Entwurf einer Datenbank, unabhängig vom konkreten DBS.

Grundidee: Reale Welt (Mini-Welt) lässt sich durch Objekte und Beziehungen zwischen Objekten beschreiben (Objekte: Entities, Beziehungen: Relationships).

Gleichartige Entities (Objektinstanzen) werden zu Entity-Typen (vergleichbar Klassen) bzw. Relationships zu Relationship-Typen zusammengefasst.

Beispiele: Entity-Typ: "Bibliotheksbenuzer", "Buch", und Relationship-Typ: "entleiht".

Attribute bei Entities und Relationships, z.B. "Name" bei Entity "Bibliotheksbenuzer", "Entleihdatum" bei Relationship "leiht aus".

Entity-Typ

Relationship-Typ

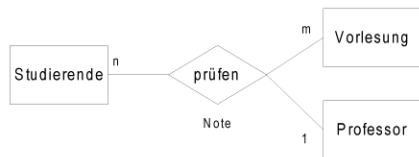
Erweiterung auf n-stellige Beziehungen

Modellierung einer Datenbank findet auf der Ebene von Entity-Typen, Relationship-Typen und Attributen statt.

Generalisierung



Relationship-Typen können nicht nur 2-stellig (binär), sondern auch n-stellig sein. Beispiel für einen 3-stellige Relationship-Typen



Darstellung der Funktionalität der Beziehung

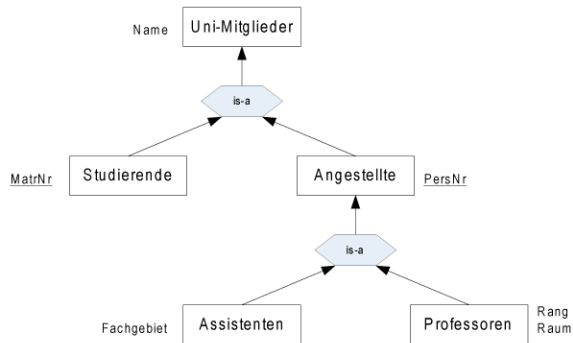
prüfen: Studierende * Vorlesung ⇒ Professor.

Mehrere Studierende können über den Stoff von mehreren Vorlesungen von einem Professor geprüft werden.

Generated by Targeteam



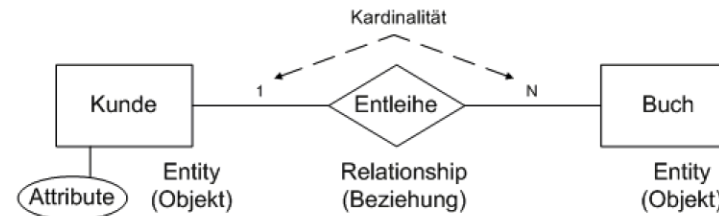
Generalisierung dient beim Konzeptentwurf zur übersichtlicheren Strukturierung der Entity-Typen



Generated by Targeteam



Entity-Relationship-Modell eignet sich zur Darstellung des Datenbankschemas.



Graphisches Hilfsmittel zur semantischen Modellierung eines Anwendungsgebietes, d.h. zum Entwurf einer Datenbank, unabhängig vom konkreten DBS.

Grundidee: Reale Welt (Mini-Welt) lässt sich durch Objekte und Beziehungen zwischen Objekten beschreiben (Objekte: Entities, Beziehungen: Relationships).

Gleichartige Entities (Objektinstanzen) werden zu Entity-Typen (vergleichbar Klassen) bzw. Relationships zu Relationship-Typen zusammengefasst.

Beispiele: Entity-Typ: "Bibliotheksbenuzer", "Buch", und Relationship-Typ: "entleiht".

Attribute bei Entities und Relationships, z.B. "Name" bei Entity "Bibliotheksbenuzer", "Entleihtdatum" bei Relationship "leiht aus".

Entity-Typ

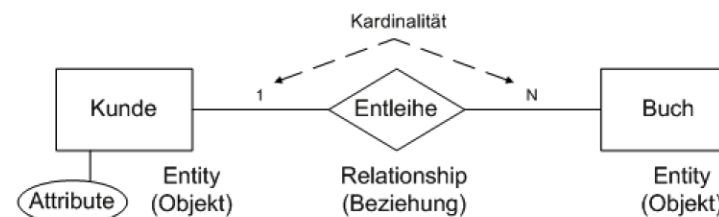
Relationship-Typ

Erweiterung auf n-stellige Beziehungen

Modellierung einer Datenbank findet auf der Ebene von Entity-Typen, Relationship-Typen und Attributen



Entity-Relationship-Modell eignet sich zur Darstellung des Datenbankschemas.



Graphisches Hilfsmittel zur semantischen Modellierung eines Anwendungsgebietes, d.h. zum Entwurf einer Datenbank, unabhängig vom konkreten DBS.

Grundidee: Reale Welt (Mini-Welt) lässt sich durch Objekte und Beziehungen zwischen Objekten beschreiben (Objekte: Entities, Beziehungen: Relationships).

Gleichartige Entities (Objektinstanzen) werden zu Entity-Typen (vergleichbar Klassen) bzw. Relationships zu Relationship-Typen zusammengefasst.

Beispiele: Entity-Typ: "Bibliotheksbenuzer", "Buch", und Relationship-Typ: "entleiht".

Attribute bei Entities und Relationships, z.B. "Name" bei Entity "Bibliotheksbenuzer", "Entleihtdatum" bei Relationship "leiht aus".

Entity-Typ

Relationship-Typ

Erweiterung auf n-stellige Beziehungen

Modellierung einer Datenbank findet auf der Ebene von Entity-Typen, Relationship-Typen und Attributen



- Fragestellungen des Abschnitts:
 - Was unterscheidet Dateisysteme von Datenbanksystemen?
 - Wie kann die Struktur der Daten in einem Datenbanksystem dargestellt werden?
 - Was sind relationale Datenbanksysteme?
 - Was sind die grundlegenden Konstrukte von HTML?

[Dateisysteme](#)

[Datenbanksysteme](#)

[Datenbankentwurf](#)

[Relationale Datenbanksysteme](#)

[WWW - Informationssystem](#)

Generated by Targeteam



Schema : legt die Struktur der in der Datenbank gespeicherten Daten fest. Darstellung mit Hilfe von Relationen
 $Relation R \subseteq Wertebereich (Attribut 1) * \dots * Wertebereich (Attribut n)$

- Beispiel der Relation Telefonbuch

Telefonbuch $\subseteq Text * Text * Zahl$

Darstellung als Schema

Telefonbuch: {[Name: Text, Adresse: Text, **Telefonnr: Zahl**] }

- Relationale Darstellung von Entity-Typen:

Kunde: {[**Kundennr: Zahl** , Vorname: Text, Nachname: Text, Ort: Text]}

Kundennr ist der Primärschlüssel zur Identifizierung der einzelnen Kunden.

- Relationale Darstellung von Relationship-Typen:

entleiht: {[**Kundennr: Zahl** , **Inventarnr: Zahl** , Datum: Datum]}

Kundennr und Inventarnr dient zur Identifizierung eines Entleihvorgangs.

Generated by Targeteam



In relationalen Datenbanken "sieht" der Benutzer die Information in Form von Tabellen (Relationen). Jede dieser Tabellen besteht aus Zeilen und Spalten; Spalten repräsentieren die Attribute von Entities.

Daten in Menge von Tabellen (Relationen) gespeichert. Meist eine Tabelle je Entity-Typ und eine Tabelle je Relationship-Typ.

Je Tabelle: Name, Spalten, Zeilen.

Je Zeile: ein zusammengehöriger Datensatz (Tupel einer Relation). Spalten als "Attribute" bezeichnet.

Jede Tabelle hat "Primärschlüssel": identifiziert Zeilen (Datensätze) eindeutig.

z.B. jeder einzelner Kunde wird durch Kundennr identifiziert.

Ordnung der Zeilen irrelevant.

Ordnung der Spalten irrelevant, da durch Namen bezeichnet.

Für Benutzer relevante Informationen: Datenwerte in den Tabellen.

Beziehungen zwischen zwei Tabellen

[Beispiel](#)

Generated by Targeteam



Kundennr
 Vorname
 Nachname
 Ort

Kundennr
Inventarnr
 Datum

Inventarnr
 Titel
 Verlag

Tabelle Kunde

Modelliert Entity-Typ "Kunde", je Zeile ein Kunde; Primärschlüssel Kundennummer.

Kunde: {[**Kundennr: Zahl** , Vorname: Text, Nachname: Text, PLZ: Zahl, Ort: Text, Straße: Text]}

<u>Kundennr</u>	Vorname	Nachname	PLZ	Ort	Straße

Tabelle Buch

Modelliert Entity-Typ "Buch", je Zeile ein Buch; Primärschlüssel Inventarnummer.

<u>Inventarnr</u>	Titel	Verlag	Preis	Erscheinungsdatum

.....





Kundenr
Vorname
Nachname
Ort
.....

Kundenr
Inventarnr
Datum

Inventarnr
Titel
Verlag
.....

Tabelle Kunde

Modelliert Entity-Typ "Kunde", je Zeile ein Kunde; Primärschlüssel Kundennummer.

Schem Kunde: {[**Kundenr: Zahl**, Vorname: Text, Nachname: Text, PLZ: Zahl, Ort: Text, Straße: Text]}

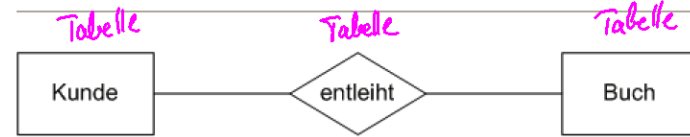
<u>Kundenr</u>	Vorname	Nachname	PLZ	Ort	Straße

Tabelle Buch

Modelliert Entity-Typ "Buch", je Zeile ein Buch; Primärschlüssel Inventarnummer.

<u>Inventarnr</u>	Titel	Verlag	Preis	Erscheinungsdatum

Tabelle Entleihe



Kundenr
Vorname
Nachname
Ort
.....

Kundenr
Inventarnr
Datum

Inventarnr
Titel
Verlag
.....

Tabelle Kunde

Modelliert Entity-Typ "Kunde", je Zeile ein Kunde; Primärschlüssel Kundennummer.

Schem Kunde: {[**Kundenr: Zahl**, Vorname: Text, Nachname: Text, PLZ: Zahl, Ort: Text, Straße: Text]}

<u>Kundenr</u>	Vorname	Nachname	PLZ	Ort	Straße

Tabelle Buch

Modelliert Entity-Typ "Buch", je Zeile ein Buch; Primärschlüssel Inventarnummer.

<u>Inventarnr</u>	Titel	Verlag	Preis	Erscheinungsdatum

{[Inventarnr: Zahl, -]}