



# Script generated by TTT

Title: Einf\_HF (28.04.2014)  
 Date: Mon Apr 28 14:14:45 CEST 2014  
 Duration: 90:16 min  
 Pages: 30

Entity-Relationship-Diagramm: logisches Modell einer Datenbank. Für Implementierung in DBS: physikalisches Modell nötig.

Beispiel: relationales Datenbankmodell.

### Relationales Modell

### Tabellendarstellung

### Normalisierung

### Umsetzung des ER-Modells

### Sichten

Verwendung von Sichten zur Auswahl einer Teilmenge der Tabellendaten.

Beispiel für eine Sicht: Vorname und Nachname von Kunden, die in München wohnen.

Views sind damit nichts anderes als benannte Such-Abfragen (z.B. SQL-Anfragen in MS ACCESS mit Namen).

### Abfragesprache SQL

### Beispielsysteme

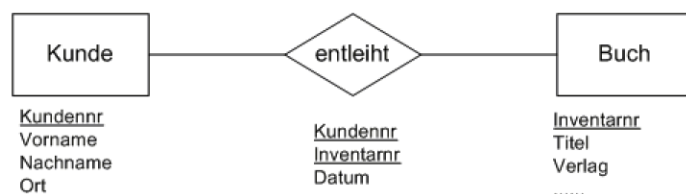
mySQL (Open Source), Microsoft Access, Microsoft SQL Server, Oracle, DB2 (IBM), Sybase, Informix

### Microsoft Access

Generated by Targeseam



## Beispiel



### 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



## Normalisierung



Mehrfache Speicherung derselben Information in gleicher oder mehreren Tabellen kann zu Konsistenzproblemen bei Änderungen führen.

Name	Vorname	PLZ	Ort
Huber	Franz	83022	Rosenheim

die selbe Information ist mehrfach dargestellt: 83022 ist die PLZ von Rosenheim

⇒ Lösung: Normalisierung

### Redundanz - Anomalien

#### 1. Normalform

#### 2. Normalform

#### 3. Normalform

### Zusammenfassung

Bei der Normalisierung werden Tabellen so zerlegt, dass keine Redundanzen mehr auftreten

1. Normalform: sämtliche Attributwerte sind atomar, d.h. nicht aus mehreren Elementen zusammengesetzt.
2. Normalform: 1. Normalform und jedes nicht zum Primärschlüssel gehörige Attribut ist von diesem voll funktional abhängig.
3. Normalform: 2. Normalform und es existieren keine transitiven Abhängigkeiten.

Generated by Targeseam



Unter Redundanz versteht man die überflüssige Mehrfachspeicherung von Daten: verschwendet Speicherplatz

kann bei Änderungen zu Inkonsistenzen oder Fehler führen (Anomalien).

Nr	Name	Vorname	PLZ	Ort	Ware	Preis
1	Huber	Franz	83022	Rosenheim	Hemd	22,50
2	Huber	Franz	83022	Rosenheim	Hose	78,90
3	Meyer	Christa	86321	Ganselham	Bluse	45,30

Update-Anomalie: bei einer Änderung von Daten werden Datensätze übersehen.

Beispiel: Adresse von F. Huber ist mehrfach in Datenbank gespeichert; sie wird jedoch nicht an allen Stellen geändert.

Delete-Anomalie: beim Löschen gehen Informationen verloren, die man später eventuell wieder benötigt.

Beispiel: Einträge von Huber werden gelöscht => Information über die Postleitzahl von Rosenheim geht verloren.

Insert-Anomalie: neue Datensätze lassen sich nicht eintragen, da Teile des Primärschlüssels fehlen.

Generated by Targeteam



Mehrfache Speicherung derselben Information in gleicher oder mehreren Tabellen kann zu Konsistenzproblemen bei Änderungen führen.

Name	Vorname	PLZ	Ort
Huber	Franz	83022	Rosenheim

dieselbe Information ist mehrfach dargestellt: 83022 ist die PLZ von Rosenheim

=> Lösung: Normalisierung

**Redundanz - Anomalien**

**1. Normalform**

**2. Normalform**

**3. Normalform**

**Zusammenfassung**

Bei der Normalisierung werden Tabellen so zerlegt, dass keine Redundanzen mehr auftreten

1. Normalform: sämtliche Attributwerte sind atomar, d.h. nicht aus mehreren Elementen zusammengesetzt.

2. Normalform: 1. Normalform und jedes nicht zum Primärschlüssel gehörige Attribut ist von diesem voll funktional abhängig.

3. Normalform: 2. Normalform und es existieren keine transitiven Abhängigkeiten.

Generated by Targeteam



Eine Tabelle ist in 1. Normalform, falls alle Attribute nur atomare Werte annehmen können. Keine Mengen, Aufzählungen.

Name	Vorname	Adresse
Müller	Anna	Sudentenstr. 18, 83022 Rosenheim
Huber	Karl	Hauptstr. 4, 86321 Ganselham

Normalisierung

Name	Vorname	Straße	PLZ	Ort
Müller	Anna	Sudentenstr. 18	83022	Rosenheim
Huber	Karl	Hauptstr. 4	86321	Ganselham

Generated by Targeteam



Unter Redundanz versteht man die überflüssige Mehrfachspeicherung von Daten: verschwendet Speicherplatz

kann bei Änderungen zu Inkonsistenzen oder Fehler führen (Anomalien).

Nr	Name	Vorname	PLZ	Ort	Ware	Preis
1	Huber	Franz	83022	Rosenheim	Hemd	22,50
2	Huber	Franz	83022	Rosenheim	Hose	78,90
3	Meyer	Christa	86321	Ganselham	Bluse	45,30

Update-Anomalie: bei einer Änderung von Daten werden Datensätze übersehen.

Beispiel: Adresse von F. Huber ist mehrfach in Datenbank gespeichert; sie wird jedoch nicht an allen Stellen geändert.

Delete-Anomalie: beim Löschen gehen Informationen verloren, die man später eventuell wieder benötigt.

Beispiel: Einträge von Huber werden gelöscht => Information über die Postleitzahl von Rosenheim geht verloren.

Insert-Anomalie: neue Datensätze lassen sich nicht eintragen, da Teile des Primärschlüssels fehlen.

Generated by Targeteam



Funktionale Abhängigkeit von Attributen

<u>TitelNr</u>	Titelname	Autor	<u>ExNr</u>	Zustand	Verliehen
1222	Fräulein Smilla	Peter Hoeg	01	Gut	ja
1222	Fräulein Smilla	Peter Hoeg	02	mittel	nein
1222	Fräulein Smilla	Peter Hoeg	03	schlecht	nein

Primärschlüssel ist das Paar (TitelNr, ExNr).

Mit jedem Exemplar werden Informationen zum Titel erneut eingetragen, z.B. Titelname, Autor.

Titelname und Autor sind nur von TitelNr abhängig, nicht jedoch von ExNr.

[Umwandlung in 2. Normalform](#)

Generated by Targteam

Eine Tabelle befindet sich in 2. Normalform

wenn sie sich in 1. Normalform befindet, und

jedes Attribut, das nicht zum Schlüssel gehört, nur vom gesamten Schlüssel und nicht bereits von einem Teil des Schlüssels abhängt

<u>TitelNr</u>	Titelname	Autor	<u>ExNr</u>	Zustand	Verliehen
1222	Fräulein Smilla	Peter Hoeg	01	Gut	ja
1222	Fräulein Smilla	Peter Hoeg	02	mittel	nein
1222	Fräulein Smilla	Peter Hoeg	03	schlecht	nein

Normalisierung

<u>TitelNr</u>	Titelname	Autor
1222	Fräulein Smilla	Peter Hoeg

<u>TitelNr</u>	<u>ExNr</u>	Zustand	Verliehen
1222	01	Gut	ja
1222	02	mittel	nein
1222	03	schlecht	nein



jedes Attribut, das nicht zum Schlüssel gehört, nur vom gesamten Schlüssel und nicht bereits von einem Teil des Schlüssels abhängt

<u>TitelNr</u>	Titelname	Autor	<u>ExNr</u>	Zustand	Verliehen
1222	Fräulein Smilla	Peter Hoeg	01	Gut	ja
1222	Fräulein Smilla	Peter Hoeg	02	mittel	nein
1222	Fräulein Smilla	Peter Hoeg	03	schlecht	nein

Normalisierung

<u>TitelNr</u>	Titelname	Autor
1222	Fräulein Smilla	Peter Hoeg

<u>TitelNr</u>	<u>ExNr</u>	Zustand	Verliehen
1222	01	Gut	ja
1222	02	mittel	nein
1222	03	schlecht	nein

Generated by Targteam

Trotz Einhaltung der 2. Normalfall treten noch Redundanzen auf,

nämlich transitive funktionale Abhängigkeiten

Kundennr → PLZ → Ort

Attribut Ort enthält redundante Daten, da bereits PLZ den Wert von Ort eindeutig festlegt

<u>Kundennr</u>	Name	Vorname	Straße	PLZ	Ort
00012	Müller	Anna	Sudentenstr. 18	83022	Rosenheim
00013	Huber	Karl	Hauptstr. 8	86321	Ganselham
00014	Meier	Kurt	Körberweg 11	83022	Rosenheim

[Umwandlung in 3. Normalform](#)

Generated by Targteam



Eine Tabelle befindet sich in 3. Normalform  
wenn sie sich in 2. Normalform befindetet, und  
kein Nichtschlüssel-Attribut transitiv abhängig von einem Schlüsselattribut ist

<u>Kundennr</u>	Name	Vorname	Straße	PLZ	Ort
00012	Müller	Anna	Sudentenstr. 18	83022	Rosenheim
00013	Huber	Karl	Hauptstr. 8	86321	Ganselham
00014	Meier	Kurt	Körperweg 11	83022	Rosenheim

Normalisierung

<u>Kundennr</u>	Name	Vorname	Straße	PLZ	PLZ	Ort
00012	Müller	Anna	Sudentenstr. 18	83022	83022	Rosenheim
00013	Huber	Karl	Hauptstr. 8	86321	86321	Ganselham
00014	Meier	Kurt	Körperweg 11	83022		



Jeder Entity-Typ wird zu einer eigenen Tabelle  
die Attribute des Entity-Typs werden zu den Spalten.  
ein Attribut (oder eine Kombination von Attribute) wird als **Primärschlüssel** definiert.  
eine Tabellenzeile repräsentiert eine Instanz des Entity-Typs (Objektinstanz, Entität)

- [Umsetzung von Relationship-Typ: 1:1](#)
- [Umsetzung von Relationship-Typ: 1:n](#)
- [Umsetzung von Relationship-Typ: n:m](#)

Generated by Targeteam



Mehrfache Speicherung derselben Information in gleicher oder mehreren Tabellen kann zu Konsistenzproblemen bei Änderungen führen.

Name	Vorname	PLZ	Ort
Huber	Franz	83022	Rosenheim

dieselbe Information ist mehrfach dargestellt: 83022 ist die PLZ von Rosenheim

⇒ Lösung: Normalisierung

**Redundanz - Anomalien**

- 1. Normalform
- 2. Normalform
- 3. Normalform

**Zusammenfassung**

Bei der Normalisierung werden Tabellen so zerlegt, dass keine Redundanzen mehr auftreten

1. Normalform: sämtliche Attributwerte sind atomar, d.h. nicht aus mehreren Elementen zusammengesetzt.
2. Normalform: 1. Normalform und jedes nicht zum Primärschlüssel gehörige Attribut ist von diesem voll funktional abhängig.
3. Normalform: 2. Normalform und es existieren keine transitiven Abhängigkeiten.

Generated by Targeteam



Integration in eine er beiden Tabellen der beteiligten Entity-Typen

Schlüsselattribut einer Tabelle wird als Attribut in die zweite Tabelle aufgenommen (**Fremdschlüssel**)



<u>Kundennr</u>	Name	Vorname
00012	Müller	Anna
00013	Huber	Karl
00014	Meier	Kurt

<u>Sitznr</u>	Lage	Kundennr
15A	Fenster	00012
15B	Mitte	00013
15C	Gang	00014

Fremdschlüssel

Generated by Targeteam



Jeder Entity-Typ wird zu einer eigenen Tabelle  
 die Attribute des Entity-Typs werden zu den Spalten.  
 ein Attribut (oder eine Kombination von Attribute) wird als **Primärschlüssel** definiert.  
 eine Tabellenzeile repräsentiert eine Instanz des Entity-Typs (Objektinstanz, Entität)

- Umsetzung von Relationship-Typ: 1:1
- Umsetzung von Relationship-Typ: 1:n
- Umsetzung von Relationship-Typ: n:m

Generated by Targeteam

direkte Umwandlung  
 das Schlüsselattribut der 1-Seite wird als Fremdschlüssel in die Tabelle der n-Seite aufgenommen.



<u>Kundenr</u>	Name	Vorname	Straße	<u>PLZ</u>	<u>PLZ</u>	Ort
00012	Müller	Anna	Sudentenstr. 18	83022	83022	Rosenheim
00013	Huber	Karl	Hauptstr. 8	86321	86321	Ganselham
00014	Meier	Kurt	Körperweg 11	83022		

Fremdschlüssel

Generated by Targeteam



Darstellung in einer eigenen Tabelle  
 aufgebaut aus den Schlüsselattributen der Tabellen der beteiligten Entity-Typen



<u>Autornr</u>	Name	<u>Autornr</u>	<u>Titelnr</u>	<u>Titelnr</u>	Buchtitel	Jahr
00012	Karl Müller	00012	12340	12340	Informatik	2002
00013	Peter Huber	00013	12340	22300	Physik	1991
00014	Ernst Meier	00013	55521	55521	Geschichte	1978
		00014	22300			

Generated by Targeteam

Jeder Entity-Typ wird zu einer eigenen Tabelle  
 die Attribute des Entity-Typs werden zu den Spalten.  
 ein Attribut (oder eine Kombination von Attribute) wird als **Primärschlüssel** definiert.  
 eine Tabellenzeile repräsentiert eine Instanz des Entity-Typs (Objektinstanz, Entität)

- Umsetzung von Relationship-Typ: 1:1
- Umsetzung von Relationship-Typ: 1:n
- Umsetzung von Relationship-Typ: n:m

Generated by Targeteam



Entity-Relationship-Diagramm: logisches Modell einer Datenbank. Für Implementierung in DBS: physikalisches Modell nötig.

Beispiel: relationales Datenbankmodell.

**Relationales Modell**

**Tabellendarstellung**

**Normalisierung**

**Umsetzung des ER-Modells**

**Sichten**

Verwendung von Sichten zur Auswahl einer Teilmenge der Tabellendaten.

Beispiel für eine Sicht: Vorname und Nachname von Kunden, die in München wohnen.

Views sind damit nichts anderes als benannte Such-Abfragen (z.B. SQL-Anfragen in MS ACCESS mit Namen).

**Abfragesprache SQL**

**Beispielsysteme**

mySQL (Open Source), Microsoft Access, Microsoft SQL Server, Oracle, DB2 (IBM), Sybase, Informix

**Microsoft Access**

*Generated by Targeteam*

"Structured Query Language": Suche und Ändern von Tabelleneinträgen; seit 1989 international genormt; für fast alle relationalen Datenbanken verfügbar; Abfrage liefert als Ergebnis alle gefundenen Lösungen (d.h. mengenorientiert).

**Elementare Operationen bei Abfragen**

**Relationen**

**INSERT (Einfügen)**

INSERT INTO Entleihe VALUES (300, 100, '01/12/97')

**UPDATE (Aktualisierung)**

UPDATE Kunde SET PLZ = "80330" WHERE Strasse = 'Arcisstrasse'

**SELECT (Abfrage)**

Finde alle Kunden, die in der Arcisstraße in München wohnen:

SELECT Vorname, Nachname, Straße FROM Kunde WHERE Ort = 'München' AND Straße = 'Arcisstraße' ORDER BY Nachname

**Aggregatfunktionen und Gruppierung**

*Generated by Targeteam*



Bei der Auswertung von Datenbanken will man

bestimmte Attributwerte

aus bestimmten Datensätzen

einer Tabelle oder einer Kombination von Tabellen

Beispiele:

Liste der Wohnorte aller Kunden,

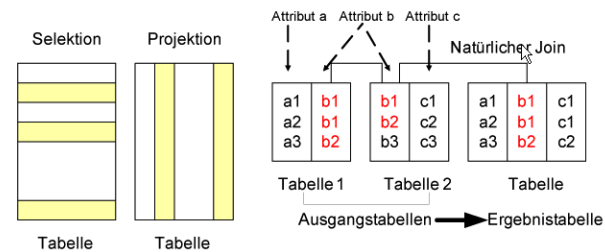
alle Kontonummern mit aktuellem Kontostand und Kundennummer

ISBN und Titelname aller Werke eines bestimmten Autors, etc.

*Generated by Targeteam*

Tabellen entsprechen Relationen, Mengenoperationen anwendbar.

Mengenoperationen als Basis für Zugriffsoperationen.



Selektion: Auswahl von Zeilen einer Tabelle über Prädikate, z.B. allen Zeilen mit Attribut Ort = 'München'.

Projektion: Auswahl der Spalten einer Tabelle, z.B. aus der Tabelle Kunde die Spalte mit den Nachnamen.

Natürlicher Join: Gleichverbund über alle gleichen Attribute und Projektion über die verschiedenen Attribute; Attribute sind durch Übereinstimmungsbedingung gegeben.

Bestimmung der Zeilen, in denen die Werte für Attribut b in Tabelle 1 und Tabelle 2 identisch sind.

Abfragen in der Datenbank lassen sich in die Teilfunktionen Selektion und Projektion zerlegen => eine Abfrage ist eine Verkettung von Selektion und Projektion.

*Generated by Targeteam*



"Structured Query Language": Suche und Ändern von Tabelleneinträgen; seit 1989 international genormt; für fast alle relationalen Datenbanken verfügbar; Abfrage liefert als Ergebnis alle gefundenen Lösungen (d.h. mengenorientiert).

### Elementare Operationen bei Abfragen

#### Relationen

##### INSERT (Einfügen)

```
INSERT INTO Entleihe VALUES (300, 100, '01/12/97')
```

##### UPDATE (Aktualisierung)

```
UPDATE Kunde SET PLZ = "80330" WHERE Strasse = 'Arcisstrasse'
```

##### SELECT (Abfrage)

Finde alle Kunden, die in der Arcisstraße in München wohnen:

```
SELECT Vorname, Nachname, Straße FROM Kunde WHERE Ort = 'München' AND Straße = 'Arcisstraße' ORDER BY Nachname
```

### Aggregatfunktionen und Gruppierung

Generated by Targeteam



Aggregatfunktionen führen Berechnungen durch und liefern als Ergebnis einen einzelnen Wert

```
SELECT AVG(GebJahr) FROM Mitarbeiter; # liefert das Durchschnitts-Geburtsjahr
```

```
SELECT SUM(GebJahr) FROM Mitarbeiter; # liefert die Summe aller Geburtsjahre
```

```
SELECT COUNT(GebJahr) FROM Mitarbeiter; # liefert die Anzahl der Mitarbeiter, die Geburtsjahr angegeben haben
```

```
SELECT MIN(GebJahr) FROM Mitarbeiter; # liefert den jüngsten Mitarbeiter
```

```
SELECT MAX(GebJahr) FROM Mitarbeiter; # liefert den ältesten Mitarbeiter
```

```
SELECT AVG(GebJahr) FROM Mitarbeiter WHERE PersNr > 2000;
```

# liefert das Durchschnitts-Geburtsjahr mit Personalnummern höher als 2000

```
SELECT AbtID, AVG(GebJahr) FROM Mitarbeiter GROUP BY AbtID;
```

# liefert das Durchschnitts-Geburtsjahr in den einzelnen Abteilungen

```
SELECT AbtID, AVG(GebJahr) FROM Mitarbeiter GROUP BY AbtID HAVING COUNT(*) > 1;
```

# liefert das Durchschnitts-Geburtsjahr zu den Abteilungen, die mehr als einen Mitarbeiter haben

Generated by Targeteam



Entity-Relationship-Diagramm: logisches Modell einer Datenbank. Für Implementierung in DBS: physikalisches Modell nötig.

Beispiel: relationales Datenbankmodell.

### Relationales Modell

#### Tabellendarstellung

#### Normalisierung

#### Umsetzung des ER-Modells

##### Sichten

Verwendung von Sichten zur Auswahl einer Teilmenge der Tabellendaten.

Beispiel für eine Sicht: Vorname und Nachname von Kunden, die in München wohnen.

Views sind damit nichts anderes als benannte Such-Abfragen (z.B. SQL-Anfragen in MS ACCESS mit Namen).

### Abfragesprache SQL

#### Beispielsysteme

mySQL (Open Source), Microsoft Access, Microsoft SQL Server, Oracle, DB2 (IBM), Sybase, Informix

#### Microsoft Access

Generated by Targeteam



MS ACCESS stellt zum Erstellen von Tabellen, Abfragen usw. im Allgemeinen Assistenten zur Verfügung. Trotzdem ist auch das direkte Arbeiten mit SQL möglich!

#### Ansichten in MS Access

MS ACCESS unterscheidet bei Tabellen und Anfragen grundsätzlich:

die **Datenblattansicht** : Anzeige der Instanz einer Basistabelle bzw. der Ergebnistabelle einer Anfrage; erlaubt auch die Eingabe, die Änderung und das Löschen von Daten in einer Basistabelle.

die **Entwurfsansicht** : interaktives Erzeugen von Tabellen bzw. Anfragen "ohne" SQL.

die **SQL-Ansicht** : direktes Arbeiten mit SQL.

#### SQL-Ansicht

Generated by Targeteam



Die SQL-Ansicht erlaubt

- die Definition von Tabellen (CREATE TABLE),
- das Ändern von Tabellen (ALTER TABLE),
- das Löschen von Tabellen (DROP TABLE),
- das Einfügen von Daten in Tabellen (INSERT INTO),
- das Ändern von Daten in Tabellen (UPDATE),
- das Löschen von Daten in Tabellen (DELETE) und
- die Erstellung von SQL-Anfragen (SELECT).

*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*



Das World Wide Web (WWW bzw. Web) ist ein über das Internet abrufbares Informationssystem, basierend auf dem Hypertext-Ansatz.

[Hypertext](#)

[Einführung in HTML](#)

[Cascading Style-Sheets \(CSS\)](#)

*Generated by Targeteam*