BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Wirtschaftsinformatik 2
LE 12 – Klausurvorbereitung

Prof. Dr. Thomas Off

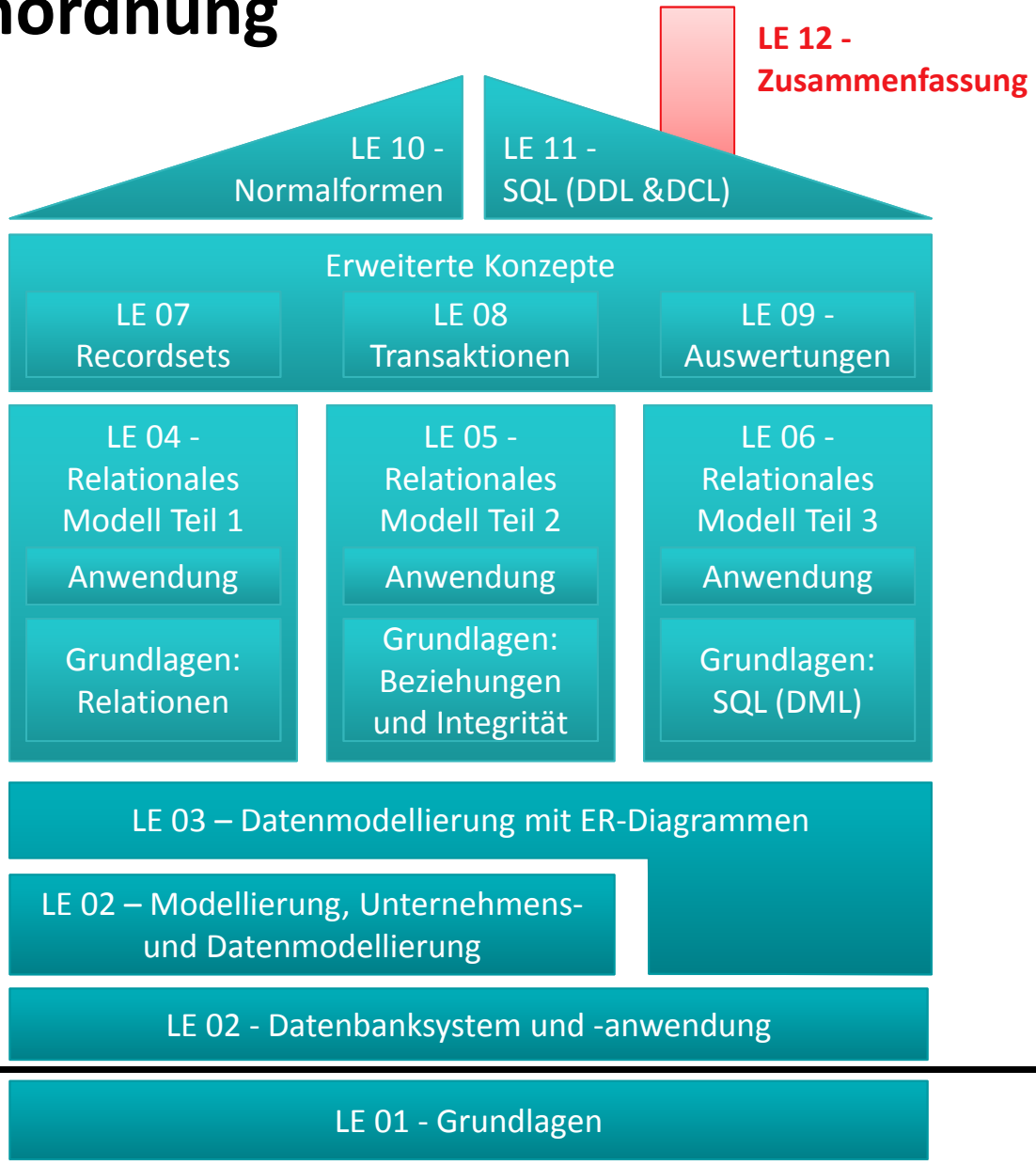
http://www.ThomasOff.de/lehre/beuth/wi2

Ziel

Ziel dieser Lehreinheit

- Überblick über die Inhalte der Lehreinheit
- Gemeinsame Wiederholung der Themen
- Einstieg in die individuellen Prüfungsvorbereitung

Einordnung


LE 12 - Zusammenfassung
LE 10 - Normalformen
LE 11 - SQL (DDL &DCL)
Erweiterte Konzepte
LE 07 Recordsets
LE 08 Transaktionen
LE 09 - Auswertungen
LE 04 - Relationales Modell Teil 1
Anwendung
Grundlagen: Relationen
LE 05 - Relationales Modell Teil 2
Anwendung
Grundlagen: Beziehungen und Integrität
LE 06 - Relationales Modell Teil 3
Anwendung
Grundlagen: SQL (DML)
LE 03 – Datenmodellierung mit ER-Diagrammen
LE 02 – Modellierung, Unternehmens- und Datenmodellierung
LE 02 - Datenbanksystem und -anwendung
LE 01 - Grundlagen

Inhalt

Ziel und Einordnung

Wiederholung

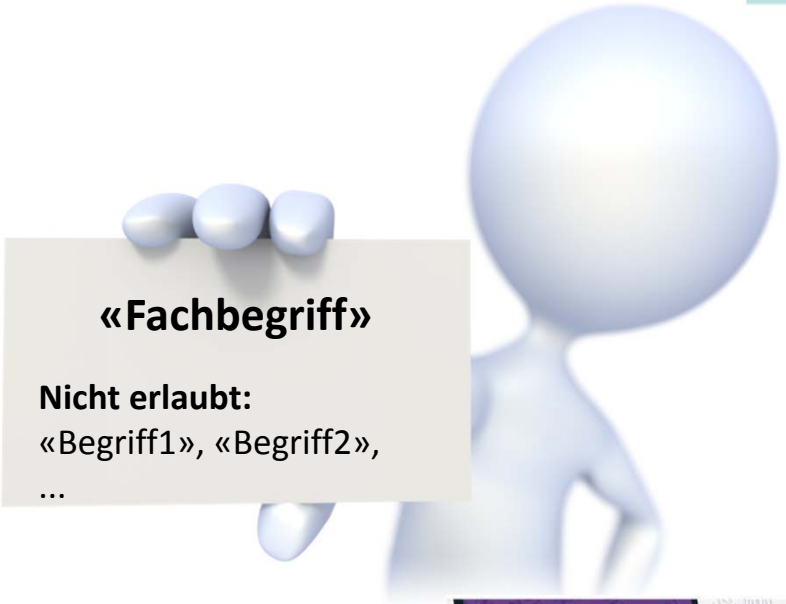
- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL DML inkl. Statistikfunktionen
- SQL DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Wiederholung zentraler Fachbegriffe

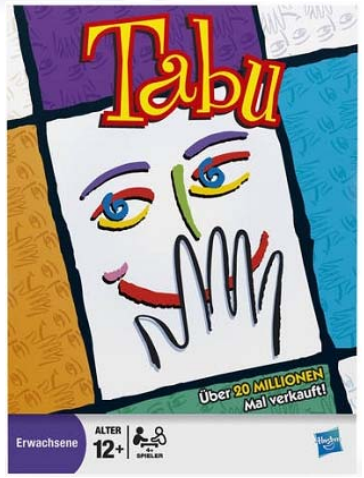
Karten mit Fachbegriffen, z.B.

- Konzepte
- Schlüsselworte
- ...
- und verbotenen Wörtern



Zwei Gruppen im Hörsaal

- je ein Gruppenmitglied zieht eine Begriffskarte
- erklärt die Karte ohne die verbotenen Begriffe zu benutzen (Mitglied der anderen Gruppe kontrolliert)
- die Gruppe, die den Begriff als erstes errät, bekommt einen Punkt
- es gewinnt die Mannschaft mit den meisten Punkten



Anwendungen und Datenbanksysteme

Oberfläche

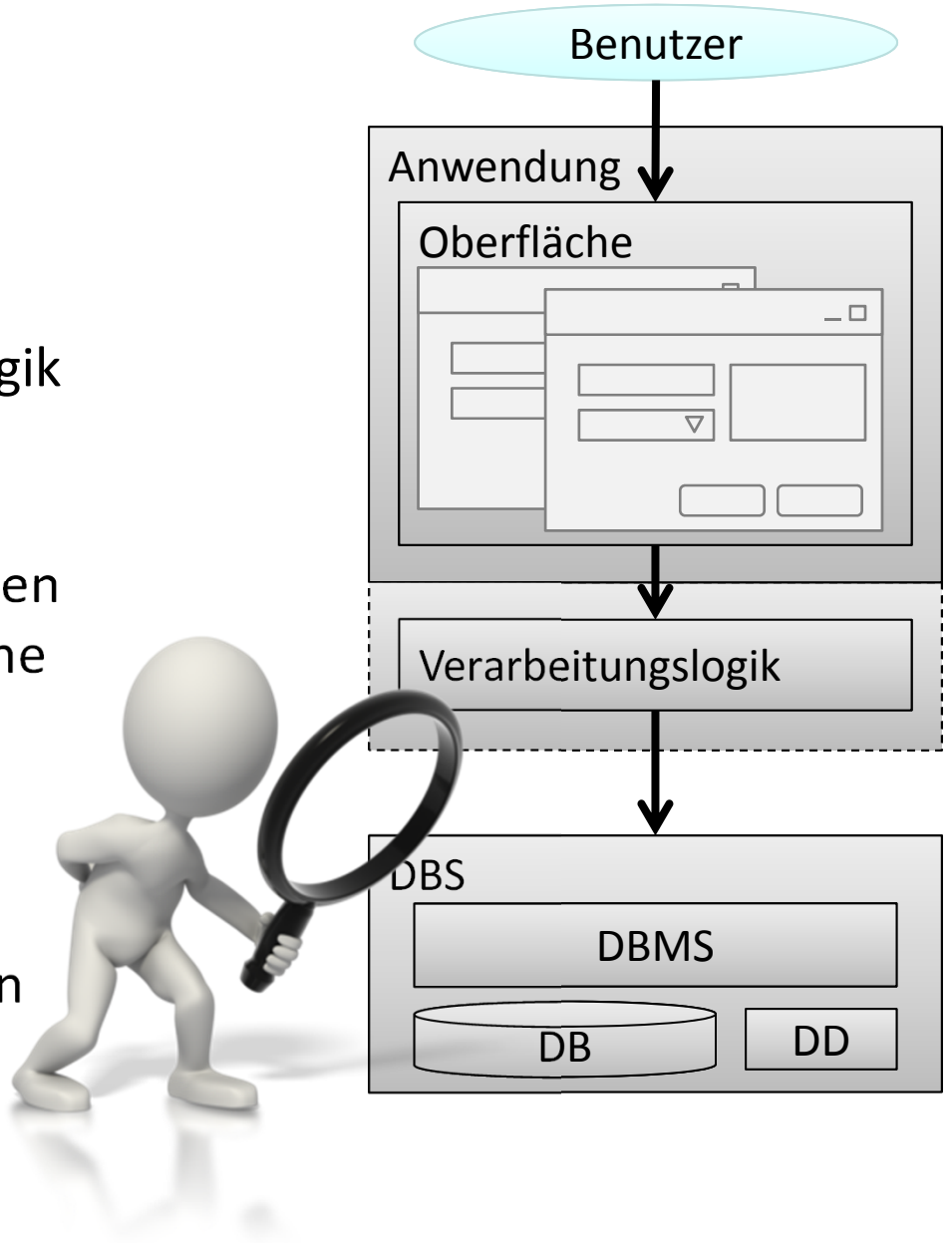
- Fenster, Dialoge mit Eingabefeldern (inkl. Steuerung)
- Visualisierung der Daten
- Zugriff auf Verarbeitungslogik

Verarbeitungslogik

- komplexe Berechnungen
- Operationen auf vielen Daten
- Zugriff auf das DBS über eine geeignete Schnittstellen zu

Datenbanksystem

- speichert die Daten in der Datenbank
- bietet Zugriffsmöglichkeiten auf Daten



Komponenten eines Datenbanksystems

Datenbankmanagementsystem (DBMS)

- bietet Anwendungsprogrammen Zugriffsmöglichkeiten i.d.R. über eine Datenbanksprache
- verwaltet und kontrolliert die abgelegten Datenbestände
- berücksichtigt dabei den Aufbau der Datenbank

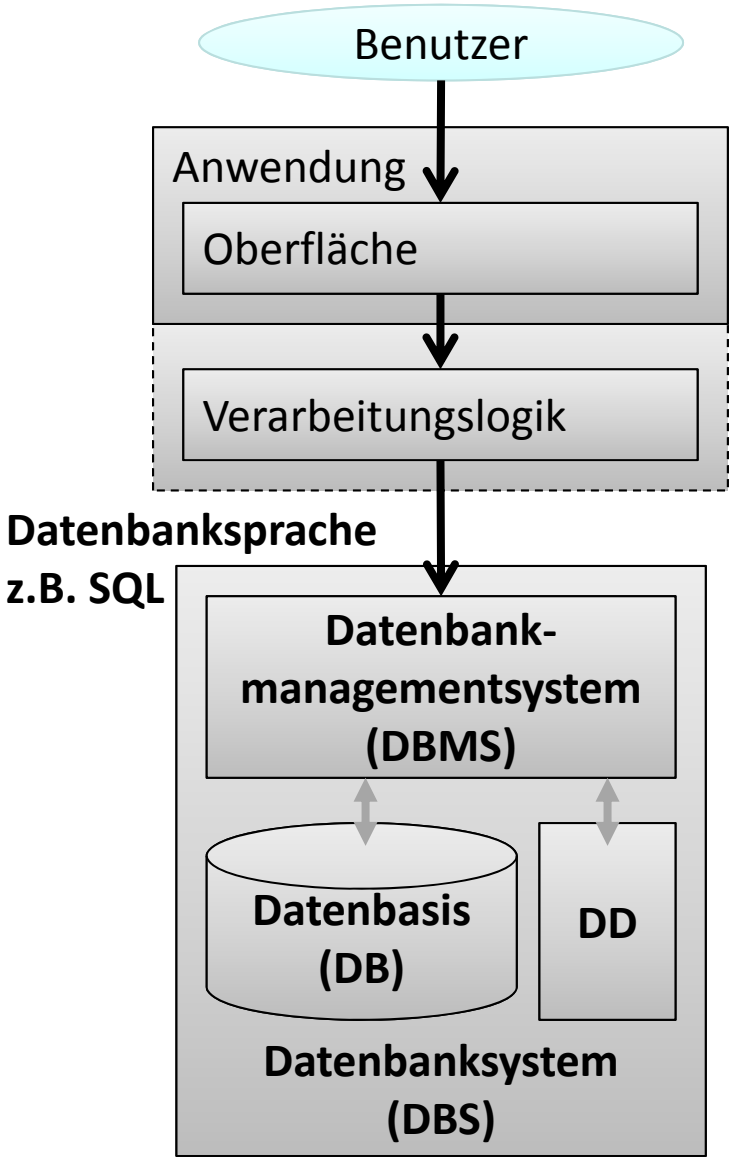
Datenbasis (syn. Datenbank, DB)

- speichert Gesamtheit aller Daten

Data Dictionary (DD)

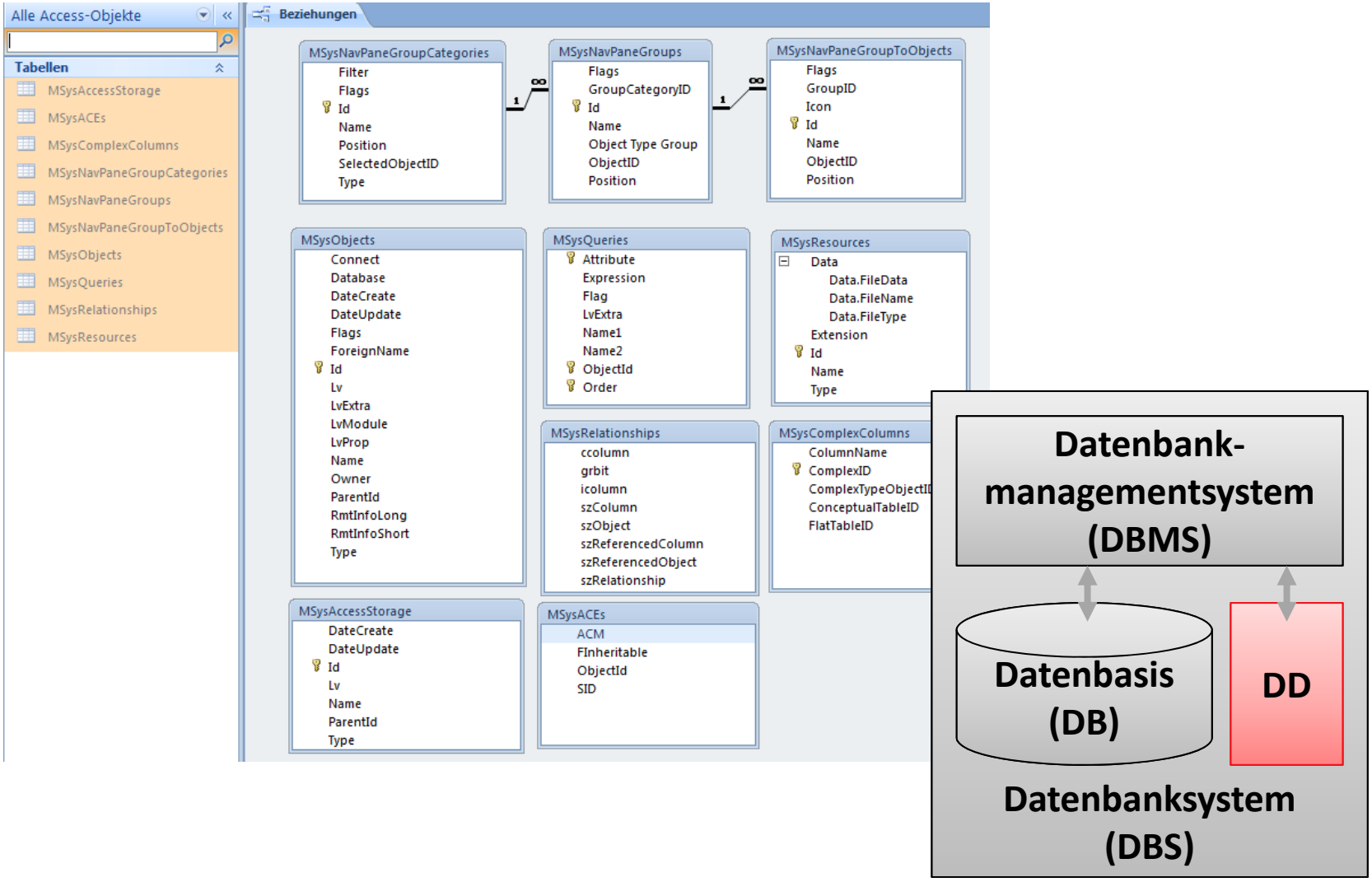
- speichert eine Beschreibung des Aufbaus der Datenbank

Datenbanksystem besteht aus DBMS + DD + mind. einer DB + Datenbanksprache



Exkurs: Data Dictionary in MS Access

Tabellen in MS Access für Data Dictionary



Prüfungsvorbereitung

Beispielhafte Aufgaben

- Was ist das DBMS und welche Aufgabe hat das es?
- Nennen Sie die Komponenten des DBS!
- Welche Aufgaben haben die Komponenten?

Inhalt

Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Modellierung betrieblicher Systeme

Modell

- "Ein Modell ist ein abstraktes System, das ein anderes (meist reales) System in vereinfachter Weise abbildet." [1, S. 12]
  - Vereinfachung/Abstraktion: weniger komplex, leichter zu überblicken
  - Abbildung: Elemente des Systems finden sich in Elementen des Modells wieder
  - Zweckgebunden: nur relevante Aspekte werden dargestellt/berücksichtigt

System

- Ein System ist eine Menge von Elementen, die durch eine Menge von Beziehungen (Relationen) miteinander verbunden sind. (Nach [1, S. 12].)
- Elemente sind nicht weiter zerlegbare Elemente. Sie können Input und/oder Output produzieren
- System durch die Systemgrenze von der Umwelt des Systems abgegrenzt

Modellierung betrieblicher Systeme

Modellierung

- Prozesse in dem ein System durch strukturähnliche Abbildung auf ein Modell abgebildet wird
  - System bestehend aus Elementen, deren Beziehungen, einer Systemgrenze und Input/Output-Beziehungen zu Umwelt
  - Modell bestehend aus Elementen und deren Beziehungen
- Ziel der Modellierung ist vereinfachtes Abbild des Systems für unterschiedliche Zwecke zu schaffen (z.B. für Erklärungen, Prognosen)
  - nur für den Zweck relevante Aspekte sind im Modell berücksichtigt
  - Vereinfachung und Abstraktion werden eingesetzt, um Komplexität des Systems zu reduzieren

Modellierung betrieblicher Systeme

- umfasst verschiedene Sichten (z.B. Organisation, Funktionen, Leistungen, Daten und deren Steuerung) auf das Unternehmen
- im Rahmen dieser Lehrveranstaltung nur Datenmodellierung relevant

Datenmodellierung

Datenmodellierung als Prozess, in dem

- die relevanten Informationsobjekte mit ihren Eigenschaften
- auf Modelle abgebildet werden

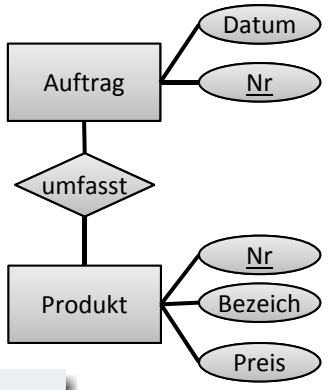
Prozessphasen

- Diskursbereich beschreiben
- Informationsmodell erstellen
- Datenmodell ableiten
- Schema implementieren

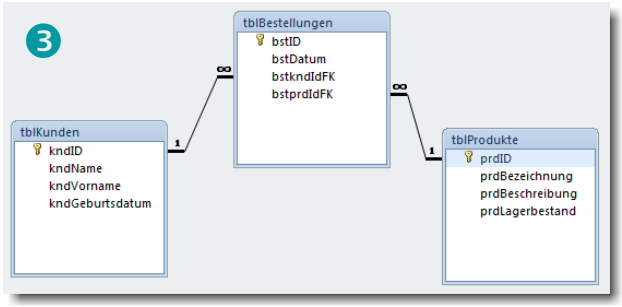
Schema wird anschließend in einem DBS umgesetzt und eine Anwendung zur Nutzung der Datenbank implementiert

1

2



3



4

```
CREATE TABLE [tblBestellungen]
([bstID] COUNTER, [bstDatum]
DATE, [bstkndIdFK] LONG,
[bstprdIdFK] LONG)

CREATE UNIQUE INDEX bstID ON
[tblBestellungen]([bstID])


-- ...
```

Entity-Relationship-Modell

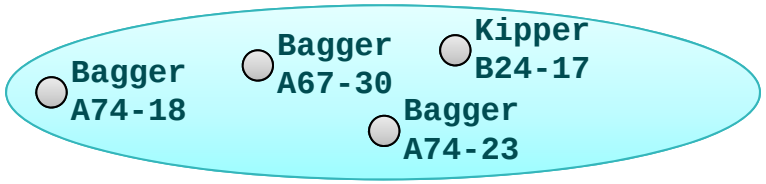
Entitäten

- Dinge der realen Welt oder der Vorstellungswelt
- zusammengefasst zu Gruppen (Mengen), die sich sehr ähnlich sind
- deshalb korrekte Bezeichnung
  - Entitätsmenge für die Zusammenfassung
  - Entität für ein Ding der Menge
- Bezeichnung durch ein Substantiv im Singular
- fachliche Beschreibung, die angibt
  - zu welchem Zweck sie im Modell existieren
  - welche Dinge der realen Welt sie repräsentieren
  - ...
- Darstellung: Rechteck mit Bezeichnung

Entity-Relationship-Modell

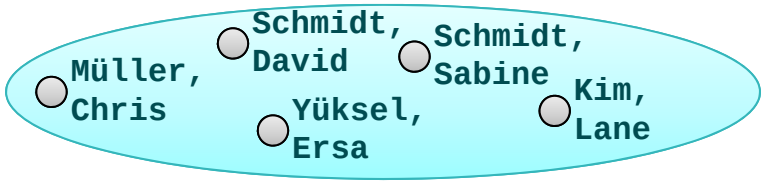
Entitätsmengen und Entitäten (Beispiele)

– Produkt (= "Menge alle Produkte des Unternehmens")



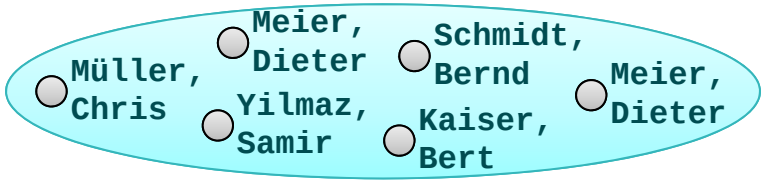
Entitätsmenge "Produkt"

– Mitarbeiter (= "Menge alle Beschäftigten im Unternehmen")



Entitätsmenge "Mitarbeiter"

– Kunde (= "Menge alle Personen, die Produkte des Unternehmens kauften")

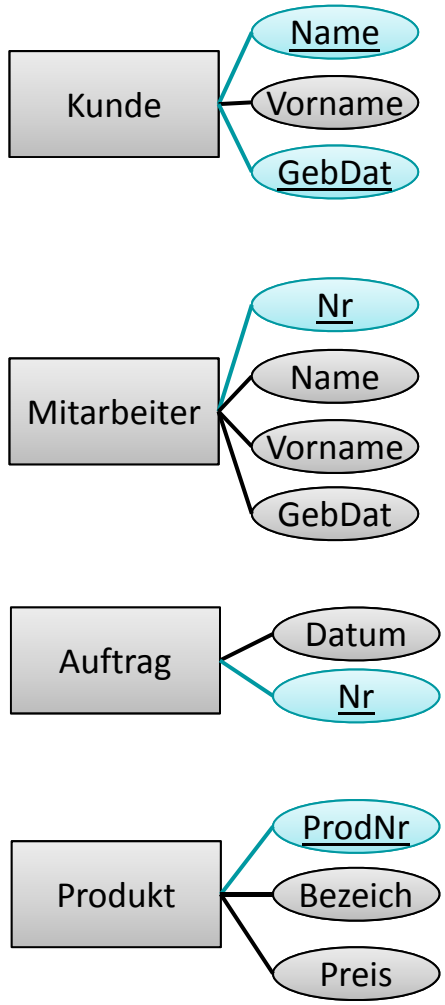


Entitätsmenge "Kunde"

Entity-Relationship-Modell

Attribute

- Schlüsselattribute
  - eindeutige Identifizierung einer Entität einer Entitätsmenge
  - ein oder mehrere Attribute können den Schlüssel bilden
- Beispiel: Entitätsmenge Kunden
  - Zwei Kunden mit Name "Sabine Müller". Wie kann man sie unterscheiden?
  - durch geeignete Schlüssel!
- Darstellung der Schlüsselattribute im ER-Modell: Bezeichnung wird unterstrichen



Entity-Relationship-Modell

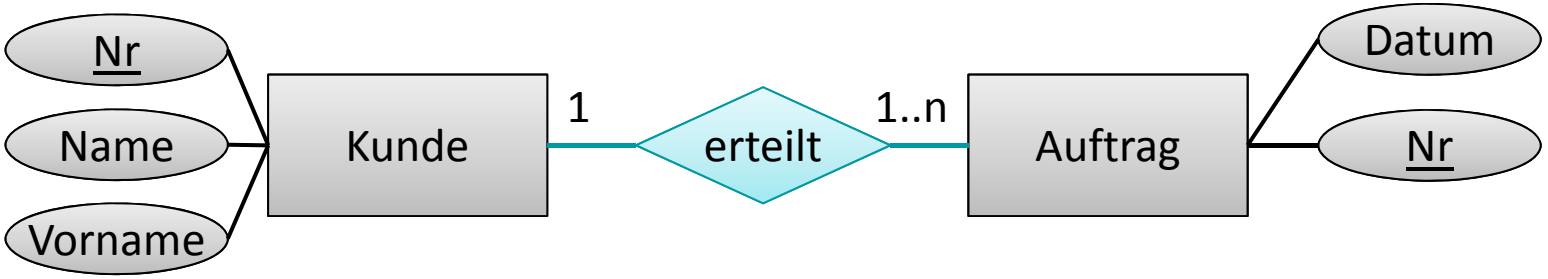
Beziehungen

- resultieren aus Abhängigkeiten, Zusammenhänge und Wechselwirkungen zwischen Entitäten
- Bezeichnung mit Verb im Singular
- fachliche Beschreibung, die angibt
  - zu welchem Zweck sie im Modell existieren
  - welche Beziehung in der realen Welt sie repräsentiert
  - ...

Entity-Relationship-Modell

Beziehungen

- haben eine Kardinalität
  - gibt an, mit wie vielen anderen Entitäten eine Entität in Beziehung stehen muss bzw. kann
  - Wird immer in beide Richtungen gelesen als: "Jede(r) ... steht in Beziehung mit ..."
- Darstellung
  - Raute mit Bezeichnung, die durch Linien (ungerichtete Kante) mit beteiligten Entitäten verbunden ist
  - Kardinalität als Ziffer/Buchstaben-Kombination auf der Linie in der Nähe der Entität



Entity-Relationship-Modell

Beziehungen (Beispiele)

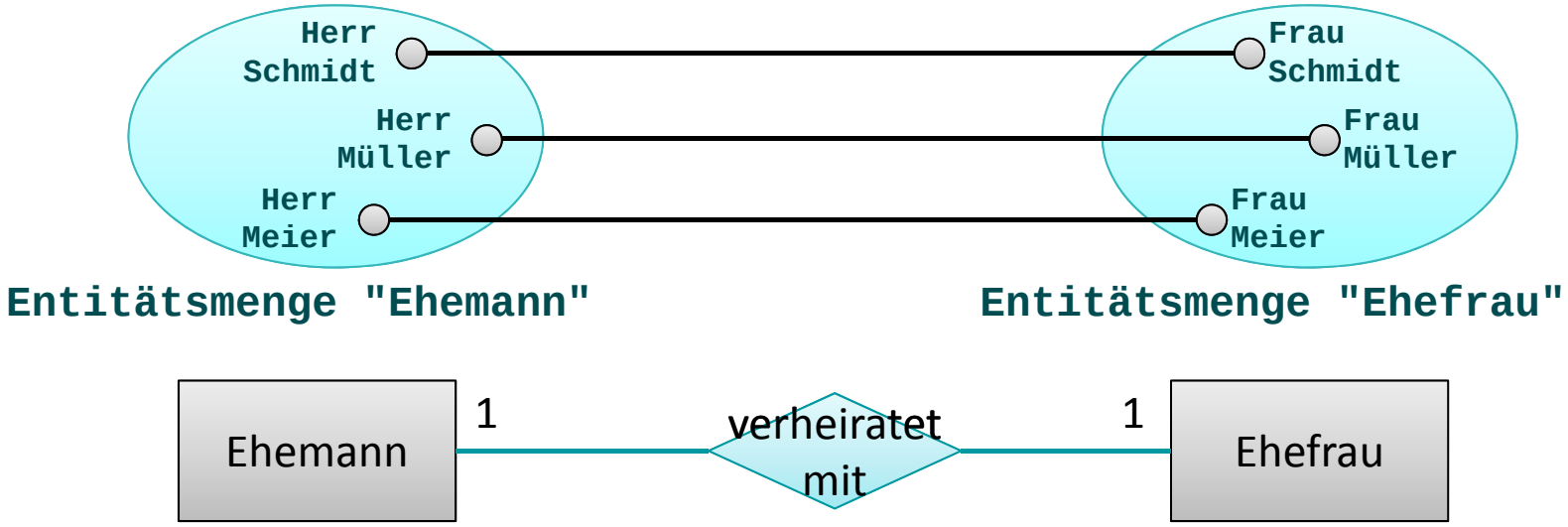
- Ein Ehemann ist verheiratet mit einer Ehefrau
- Ein Kunde erteilt einen oder mehrere Aufträge
- Kunden kaufen Produkte
- Mitarbeiter verkaufen Produkte

Entity-Relationship-Modell

Beziehungen (Beispiele)

– Ein Ehemann ist verheiratet mit einer Ehefrau

- westlichen Kulturkreis eine 1:1-Beziehung



– Ein Kunde erteilt einen oder mehrere Aufträge

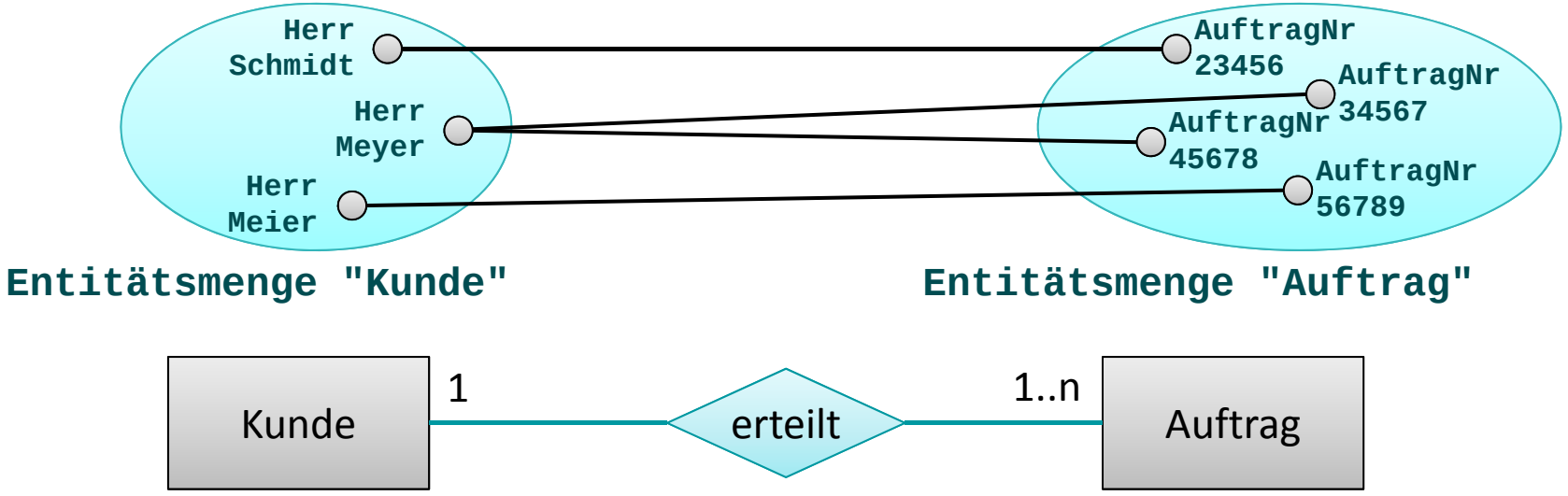
– Kunden kaufen Produkte

– Mitarbeiter beraten zu Produkten

Entity-Relationship-Modell

Beziehungen (Beispiele)

- Ein Ehemann ist verheiratet mit einer Ehefrau
- Ein Kunde erteilt einen oder mehrere Aufträge
  - Jeder Auftrag wurde von einem Kunden erteilt
  - Ein Kunde kann mehrere Aufträge erteilen

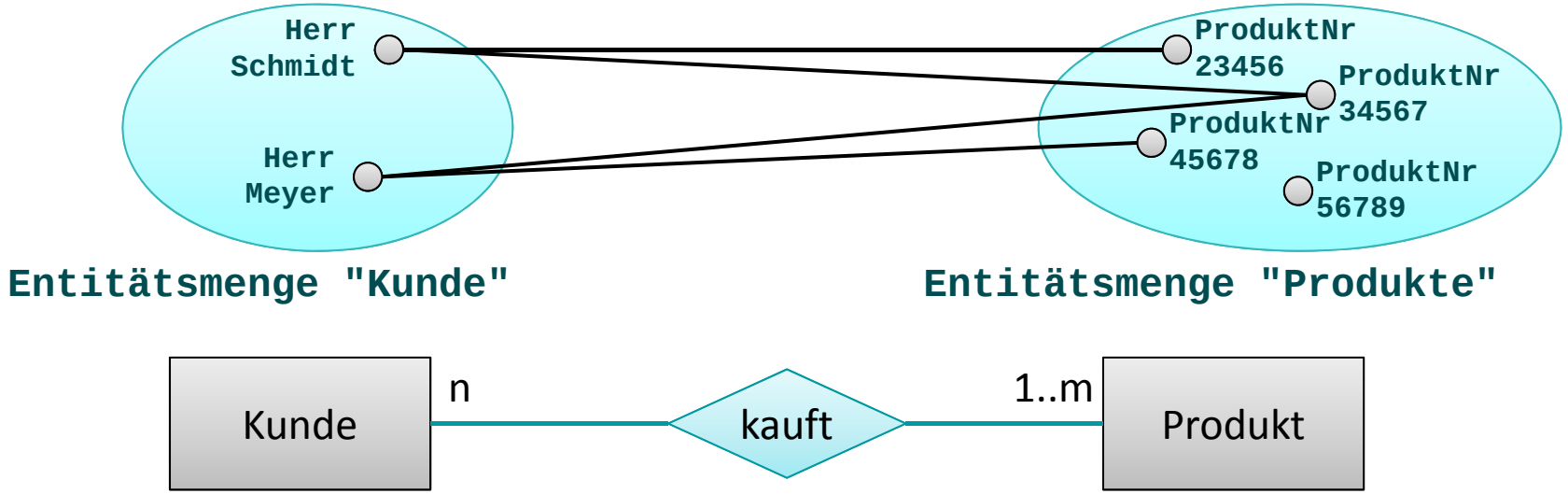


- Kunden kaufen Produkte
- Mitarbeiter beraten zu Produkten

Entity-Relationship-Modell

Beziehungen (Beispiele)

- Ein Ehemann ist verheiratet mit einer Ehefrau
- Ein Kunde erteilt einen oder mehrere Aufträge
- Kunden kaufen Produkte
  - Nicht alle Produkte werden gekauft
  - Aber jeder Kunde hat mindestens ein Produkt gekauft

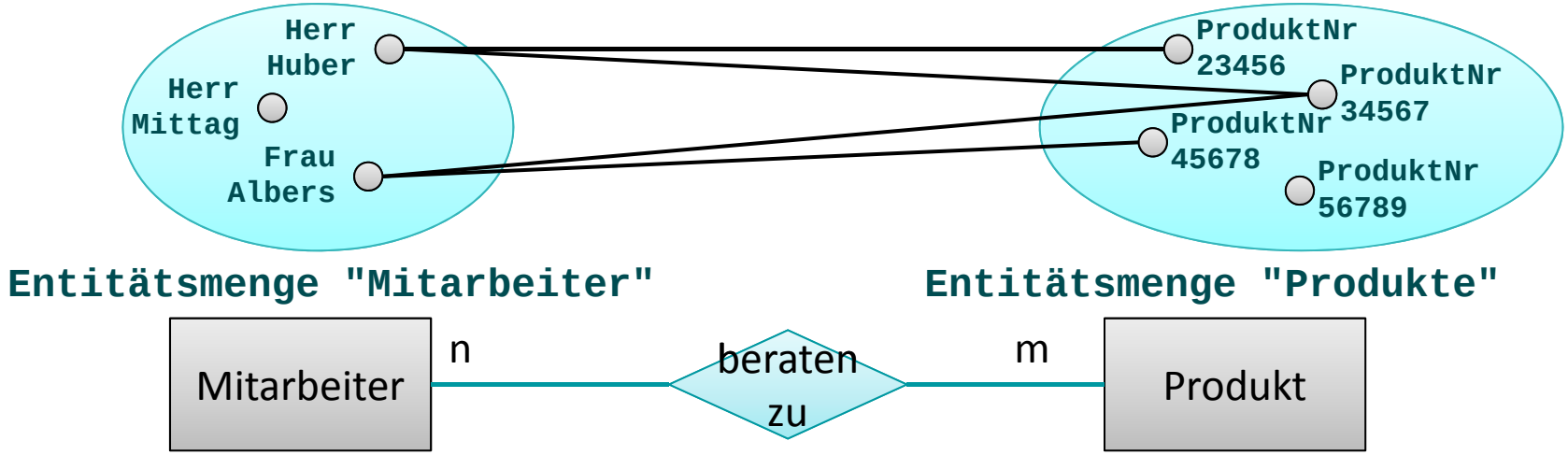


- Mitarbeiter beraten zu Produkten

Entity-Relationship-Modell

Beziehungen (Beispiele)

- Ein Ehemann ist verheiratet mit einer Ehefrau
- Ein Kunde erteilt einen oder mehrere Aufträge
- Kunden kaufen Produkte
- Mitarbeiter beraten zu Produkten
  - Zu Produkten aus dem Online-Geschäft wird keine Beratung durch Mitarbeiter angeboten (nur Online)
  - Nicht alle Mitarbeiter beraten Kunden (es gibt auch einen Chef)



Entity-Relationship-Modell

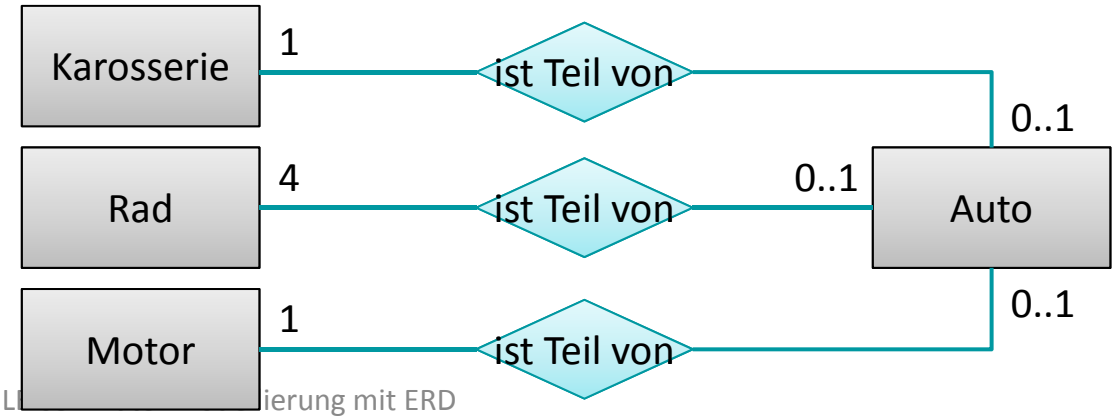
Beziehungen (Beispiele)

- Ein Ehemann ist verheiratet mit einer Ehefrau
- Ein Kunde erteilt einen oder mehrere Aufträge
- Kunden kaufen Produkte
- Mitarbeiter beraten zu Produkten

Entity-Relationship-Modell

Aggregationsbeziehung

- um die besondere Form der Beziehung
- drückt die Beziehung eines Ganzen zu seinen Bestandteilen aus
- Bezeichnung: standardisiert immer "ist Teil von" (engl. "part of")
- Beispiel
  - ein Auto besteht aus einem Motor, einer Karosserie und vier Rädern
  - der Motor, die Räder und die Karosserie können auch allein existieren (wenn sie noch nicht zusammengebaut sind)
  - jeder Motor, jedes Rad und jede Karrossserie kann nur höchstens zu einem Auto gehören



Entity-Relationship-Modell

Generalisierungsbeziehung

- ermöglicht es, gemeinsame Attribute verschiedener Entitätsmengen auf einer gemeinsamen, übergeordneten Entitätsmenge zuzuordnen
- Bezeichnung: standardisiert immer "ist ein" (engl. "is a")
- Keine Kardinalitäten!

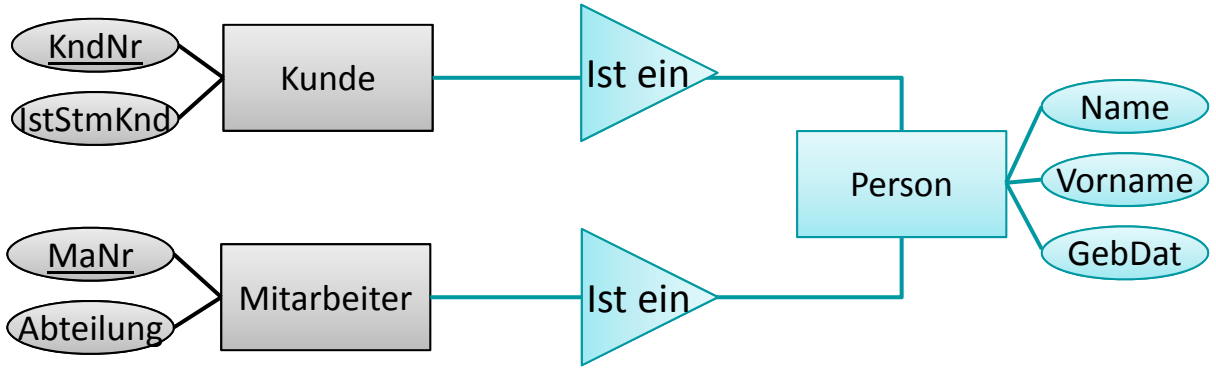
Entity-Relationship-Modell

Generalisierungsbeziehung

– Beispiel

- Ausgangssituation: Kunde und Mitarbeiter mit gleichen Attributen

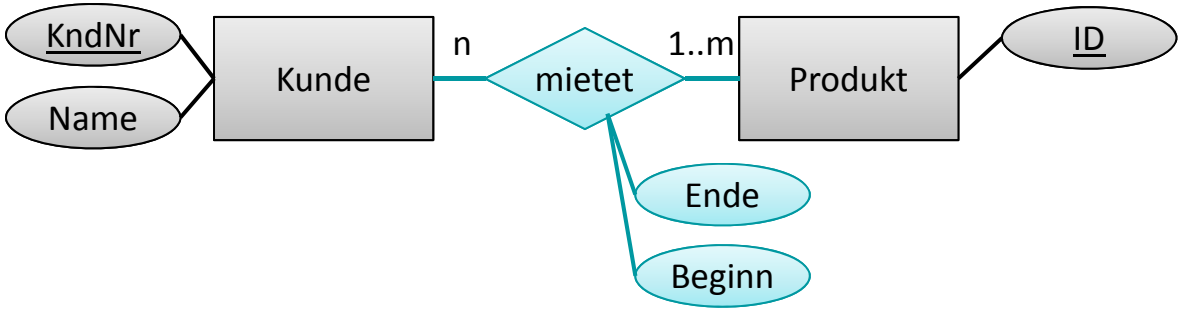
- Ergebnis der Generalisierung:



Entity-Relationship-Modell

Zusammenfassung

- ER-Modell
  - dient zur Darstellung einer abstrakten und vollständigen Beschreibung des Diskursbereichs in Form eines Informationsmodells (syn. konzeptionelles Datenmodell, semantisches Datenmodell)
  - beschreibt, **WAS** die Datenbank speichern soll
- Umfasst als Hauptbestandteile
  - Entitätsmengen,
  - Beziehungen,
  - Kardinalitäten
  - und Attribute.



- besondere Arten
  - von Attributen, die Schlüssel zur eindeutigen Identifikation von Entitäten
  - von Beziehungen, die Aggregation und die Generalisierung.

Prüfungsvorbereitung

Beispielhafte Aufgaben

- Was ist der Unterschied zwischen einem System und einem Modell?
- Definieren Sie den Begriff Modell, ...!
- In welchen Schritten verläuft der Modellierungsprozess einer Datenbank?

Prüfungsvorbereitung

Beispielhafte Aufgaben

- Stellen Sie folgenden Ausschnitt aus einem Diskursbereich in einem ER-Diagramm dar:
  - Kunden, die einen Vornamen, Namen und mehrere Adressen (mit Straße, Hausnummer, PLZ, Ort) haben, wobei jede Adresse immer zu einem Kunden gehört,
  - Kunden kaufen mind. ein Produkt mit einem Preis, einer eindeutigen Artikelnummer und einer Bezeichnung.
  - Jedes Produkt wird von beliebig vielen Kunden gekauft.

Inhalt

Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

Tabellenname	Attribut 1	...	Attribut n
	Wert		

Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

Tabellenname	Attribut 1	...	Attribut n
	Wert		

Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

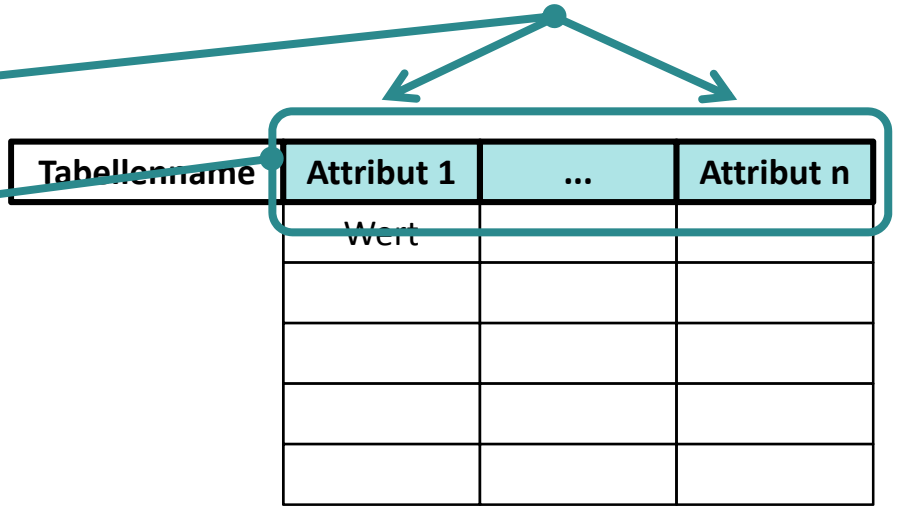


Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

Tabellenname	Attribut 1	...	Attribut n
	Wert ABC	...	Wert XYZ

Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

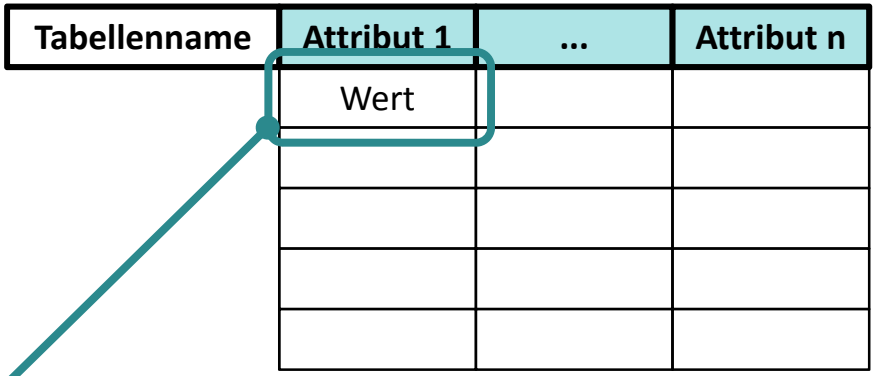


Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

Tabellenname	Attribut 1	...	Attribut n
	Wert		

Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Datenmodell (Grundkonzepte)

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

Tabellenname	Attribut 1	...	Attribut n
	Wert		

Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Schlüssel als Bestandteil von Relationen

Geht nicht!

Relation als Menge

- in der Menge sind gleiche Tupel nicht zulässig
- Tupel müssen voneinander unterschieden werden können
  - keine Tupel mit gleichen Attributwerten mehrmals vorkommen
  - ein oder mehrere Attributwerte eines Tupels müssen es von anderen unterscheidbar machen

Kunden	Name	Vorname
	Müller	Sophie
	Yilmaz	Ali
	Müller	Sophie
	Kaiser	Tina

Schlüssel

- sind Attribute, die ein Tupel eindeutig identifizierbar machen
- können ein Attribut sein oder
- aus mehrere Attributen zusammengesetzt

Kunden	KundeNr	Name	Vorname	GebDat
	123	Müller	Sophie	02.05.97
	234	Yilmaz	Ali	03.02.98
	345	Müller	Sophie	23.06.90
	456	Kaiser	Tina	03.02.98

Kunden	Name	Vorname	GebDat
	Müller	Sophie	02.05.97
	Yilmaz	Ali	03.02.98
	Müller	Sophie	23.06.90
	Kaiser	Tina	03.02.98

Schlüssel als Bestandteil von Relationen

Definition "Schlüsselkandidat"

- besteht aus einem oder mehreren Attributen, über die die zu speichernden Daten naturgemäß verfügen
- identifiziert eindeutig jedes Tupel (Datensatz)
- ist minimal, d.h. beim Weglassen eines Attributes geht Eindeutigkeit verloren
- es kann mehrere Schlüsselkandidaten geben

Definition "Stellvertreterschlüssel" (syn. "Surrogate Key")

- gibt es kein Attribut, das sich als Schlüsselkandidat eignet, wird ein künstlicher Schlüssel als Attribut hinzugefügt
- wird künstlich und ggf. automatisch erzeugt (z.B. als Autowert in MS Access)

Schlüssel als Bestandteil von Relationen

Definition "Primärschlüssel" (syn. "Primary Key")

- besitzt eine Relation
  - mehr als einen Schlüsselkandidaten, wird einer als Primärschlüssel ausgewählt
  - keinen Schlüsselkandidaten, wird ein Stellvertreterschüssel als Primärschlüssel verwendet
- identifiziert jedes Tupel der Relation eindeutig
- kann aus einem oder mehreren Attributen bestehen
- andere Schlüsselkandidaten sind "Alternativschlüssel"

Anforderungen an Primärschlüssel[1]

- Wert des Primärschlüssels soll sich im Laufe der Zeit nicht mehr ändern
- aus möglichst wenigen Attributen bestehen, um seine Verwendung zu vereinfachen
- Datentyp mit wenig Speicherplatzbedarf
- sollte Tabelle nicht komplizierter machen, wobei Stellvertreterschlüssel in der Praxis akzeptiert ist

1) vgl. [1], S. 82

Schlüssel als Bestandteil von Relationen

1. Integritätsregel (Entitätsintegrität):

Kein Bestandteil eines Primärschlüssels darf leer sein

- Primärschlüssel muss Datensätze eindeutig identifizieren, leerer Schlüssel macht keinen Sinn
- bei zusammengesetzten Primärschlüsseln darf auch nicht ein Teil leer sein

Relationales Modell im Überblick

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper

aber fundiert in Prädikaten-logik erster Ordnung (und Mengenlehre)

Tabellenname	Attribut 1	...	Attribut n
	Wert		

Abb. nach Wikipedia, http://en.wikipedia.org/wiki/Relational_model

Relationales Modell im Überblick

besteht aus wenigen, sehr einfachen Bestandteilen

- Tabelle (Relation)
- Name der Tabelle
- Tabellenspalten (Attribute) definiert durch den Tabellenkopf (unsortiert)
- Zusammengehörige Gruppen eindeutiger Attributwerte (Tupel, syn. Record, Datensatz) als unsortierte Tabellenzeilen
- Attributwerte als Zellen innerhalb der Tabelle
- Tabellenkörper
- **Schlüssel**
  - **Schlüsselkandidat**
  - **Stellvertreterschlüssel**
  - **Primärschlüssel (niemals leer)**

Tabellenname	Attribut 1	...	Attribut n
	Wert		

Relationales Datenmodell (Grundkonzepte)

Definition: Relation[1]

- Eine Relation ist eine Tabelle, die
  - aus Tabellenkopf und Tabellenkörper besteht,
  - einen Namen hat,
  - eine Menge zu speichernder Daten repräsentiert
- und die folgenden vier Eigenschaften aufweist:
  - Tupel als Zeilen im Tabellenkörper sind nicht geordnet.
  - Es gibt keine doppelten Tupel, weil sie durch einen Schlüssel eindeutig identifiziert werden können.
  - Die Attribute im Tabellenkopf sind nicht geordnet.
  - Alle Attribute sind atomar, d.h.
    - sie erlauben nur Werte eines einfachen Datentyps (z.B. Zahl, Text, Datum, Währung)
    - pro Tupel ist immer nur einen Wert für das Attribut möglich

1) nach [2], S. 67

Prüfungsvorbereitung

Beispielhafte Aufgaben

- Welchem Zweck dient der Primärschlüssel? Welchem Zweck dient der Fremdschlüssel?
- Was ist ein Schlüsselkandidat?
- Was ist ein Surrogate Key (Stellvertreterschlüssel) und wozu dient er?
- Bennen Sie die Bestandteile der dargestellten Tabelle mit den erlernten Fachbegriffen!

Inhalt

Ziel und Einordnung

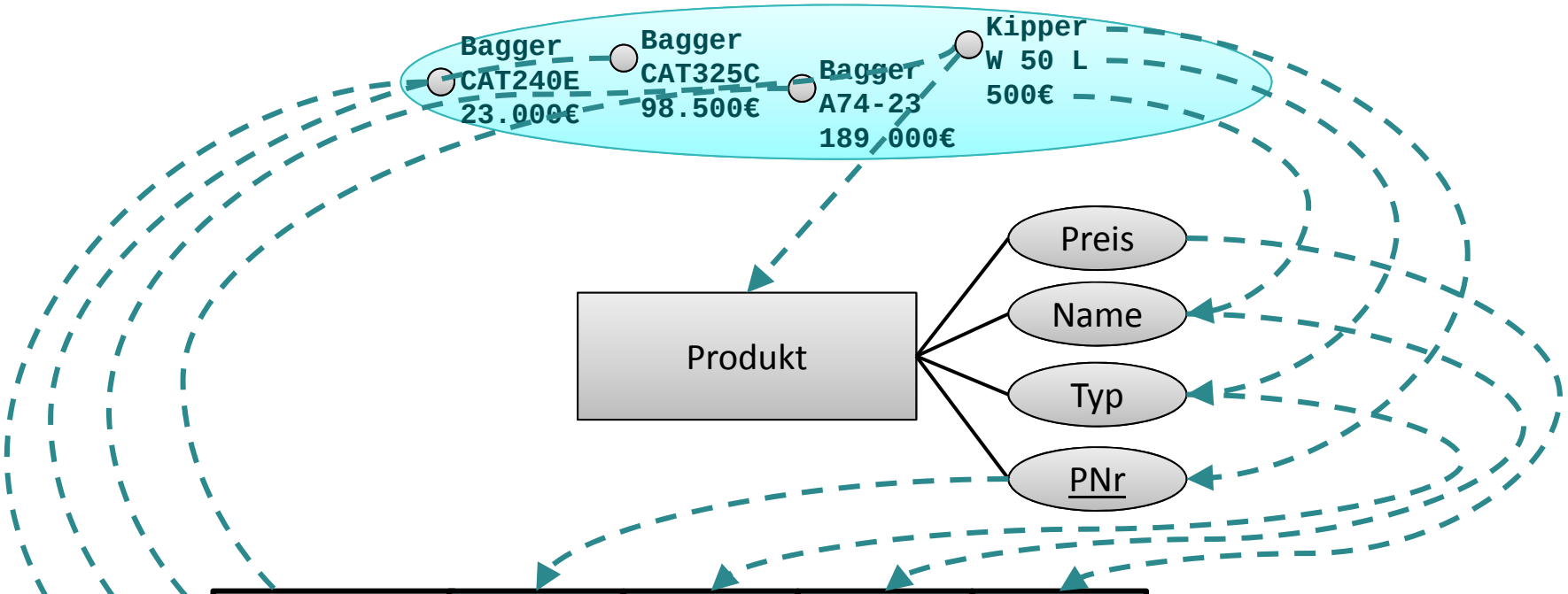
Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Entitäten auf Relationen



Produkte	PNr	Typ	Name	Preis
	4711	Bagger	A74-23	189.000€
	4712	Kipper	W 50 L	500€
	4718	Bagger	CAT325C	98.500 €
	4720	Bagger	CAT240E	23.000€

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Beziehungen auf Relationen

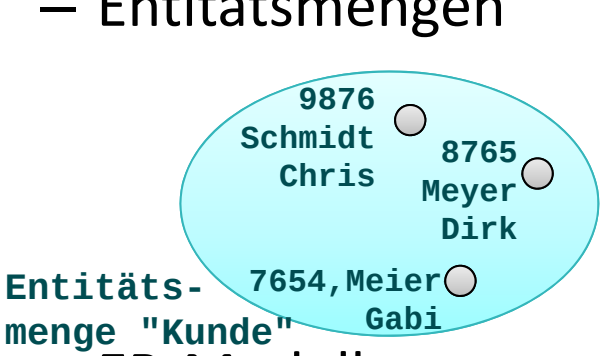
- 1:n Beziehung im relationalen Datenmodell
- n:m Beziehung im relationalen Datenmodell
- 1:1 Beziehung im relationalen Datenmodell
- Attribute von Beziehungen

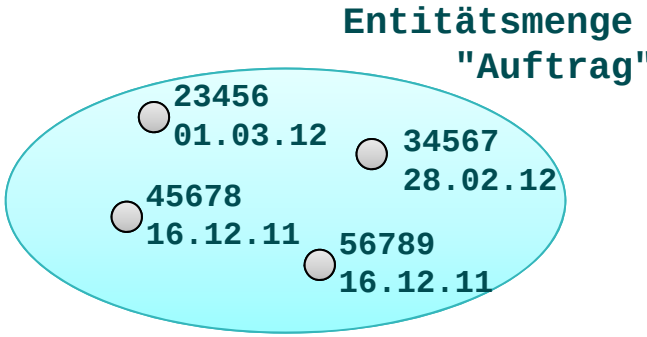
und das Konzept des Fremdschlüssel.

Übergang vom ER-Modell zum Relationalen Modell

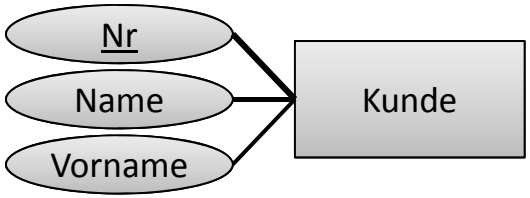
1:n Beziehungen am Beispiel "Kunde erteilt Auftrag"

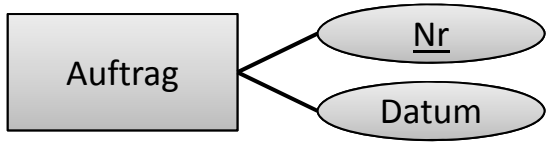
– Entitätsmengen





– ER-Modell





– Relationen

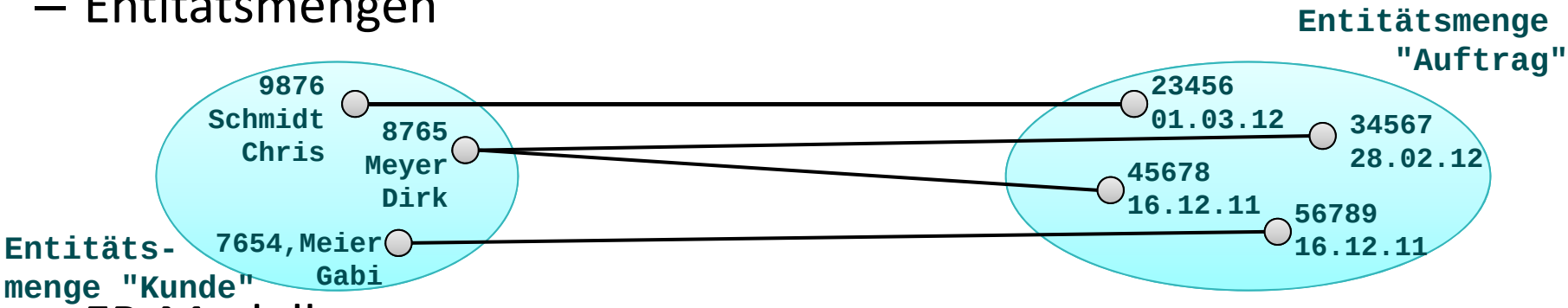
Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Auftrag	Nr	Datum
	23456	01.03.2012
	34567	28.02.12
	45678	16.12.2011
	56789	16.12.2011

Übergang vom ER-Modell zum Relationalen Modell

1:n Beziehungen am Beispiel "Kunde erteilt Auftrag"

– Entitätsmengen



– ER-Modell

– Relationen

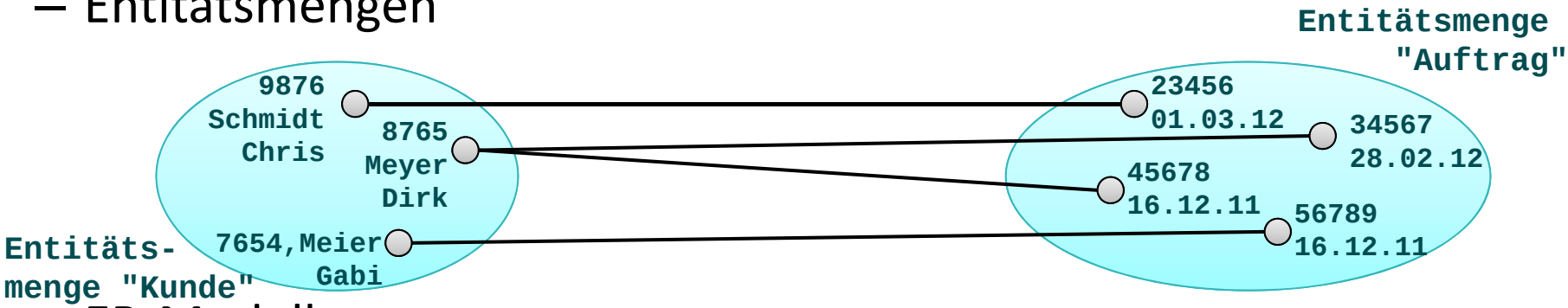
Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Auftrag	Nr	Datum
	23456	01.03.2012
	34567	28.02.12
	45678	16.12.2011
	56789	16.12.2011

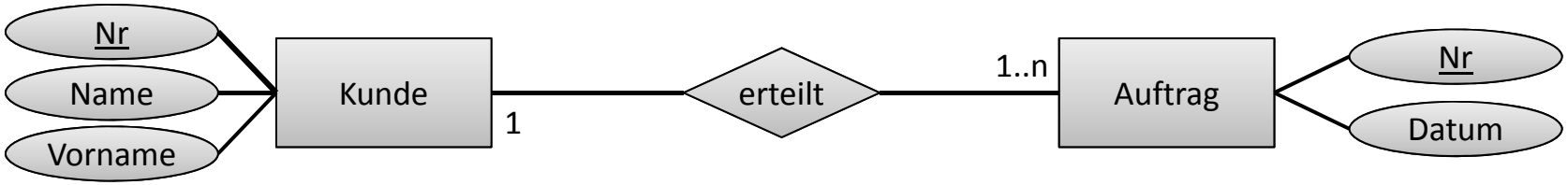
Übergang vom ER-Modell zum Relationalen Modell

1:n Beziehungen am Beispiel "Kunde erteilt Auftrag"

– Entitätsmengen



– ER-Modell



– Relationen

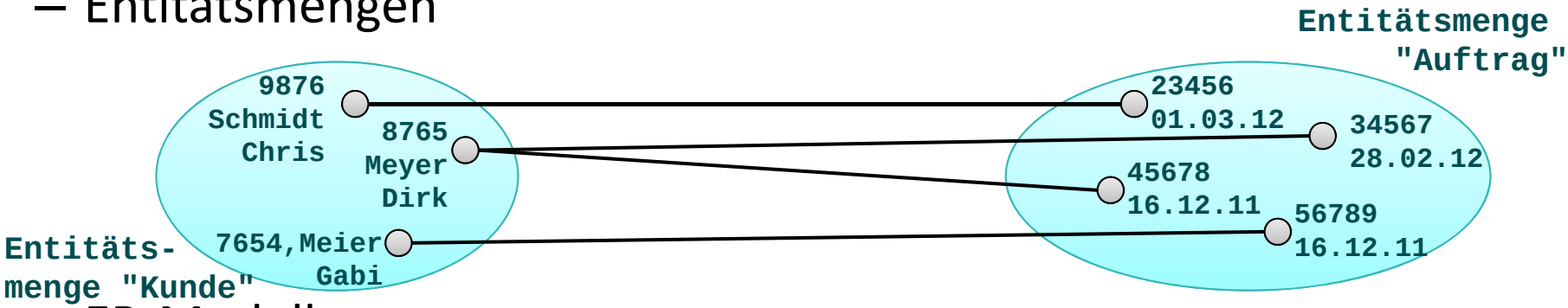
Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Auftrag	Nr	Datum
	23456	01.03.2012
	34567	28.02.12
	45678	16.12.2011
	56789	16.12.2011

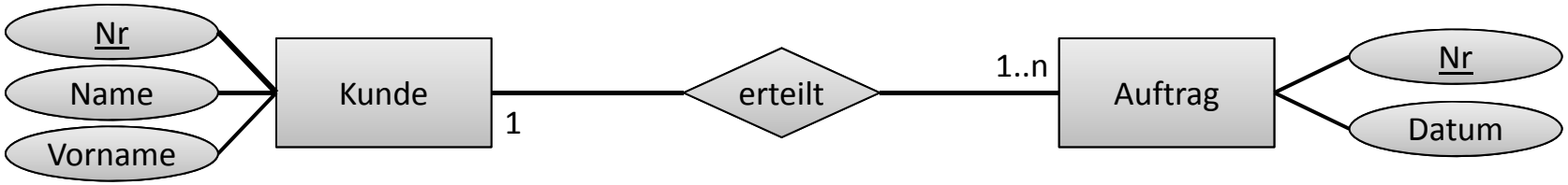
Übergang vom ER-Modell zum Relationalen Modell

1:n Beziehungen am Beispiel "Kunde erteilt Auftrag"

– Entitätsmengen



– ER-Modell



– Relationen

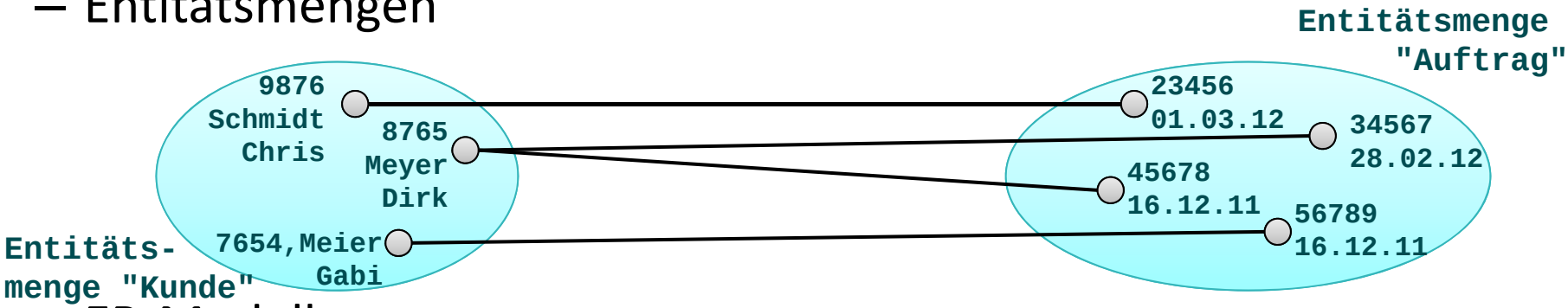
Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Auftrag	Nr	Datum
	23456	01.03.2012
	34567	28.02.12
	45678	16.12.2011
	56789	16.12.2011

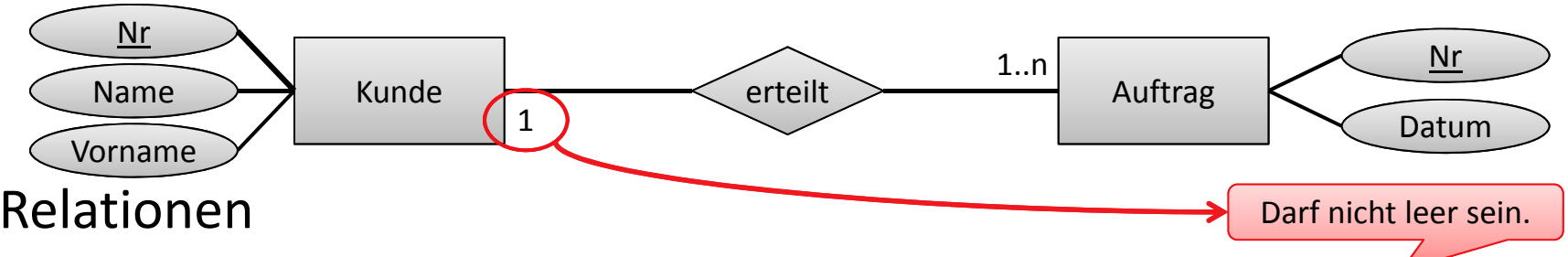
Übergang vom ER-Modell zum Relationalen Modell

1:n Beziehungen am Beispiel "Kunde erteilt Auftrag"

– Entitätsmengen



– ER-Modell



– Relationen

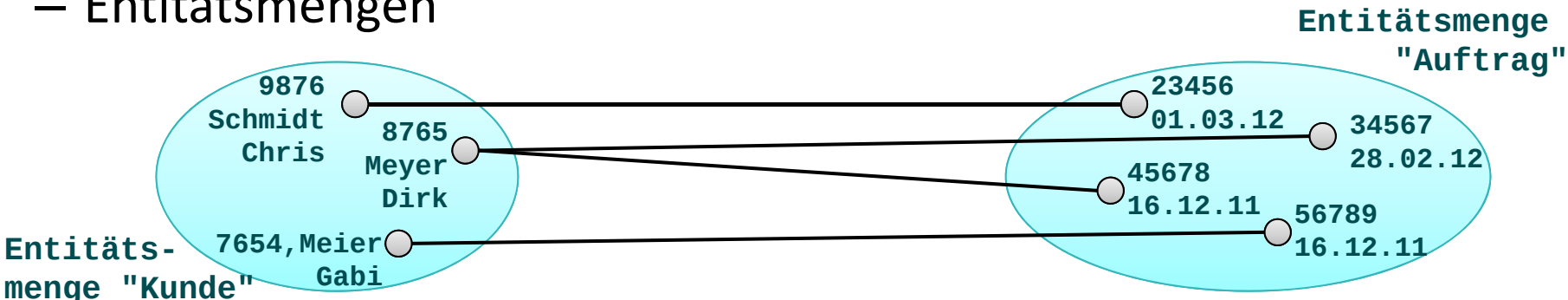
Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Auftrag	Nr	Datum	KndNr
	23456	01.03.2012	9876
	34567	28.02.12	8765
	45678	16.12.2011	8765
	56789	16.12.2011	7654

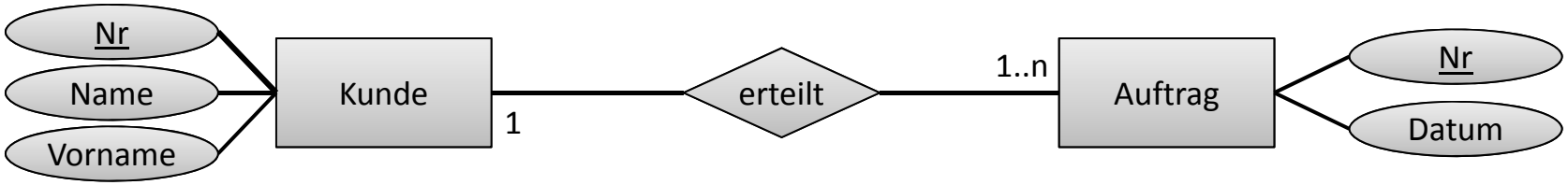
Übergang vom ER-Modell zum Relationalen Modell

1:n Beziehungen am Beispiel "Kunde erteilt Auftrag"

– Entitätsmengen



– ER-Modell



– Relationen

Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Auftrag	Nr	Datum	KndNr
	23456	01.03.2012	9876
	34567	28.02.12	8765
	45678	16.12.2011	8765
	56789	16.12.2011	7654

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Beziehungen auf Relationen

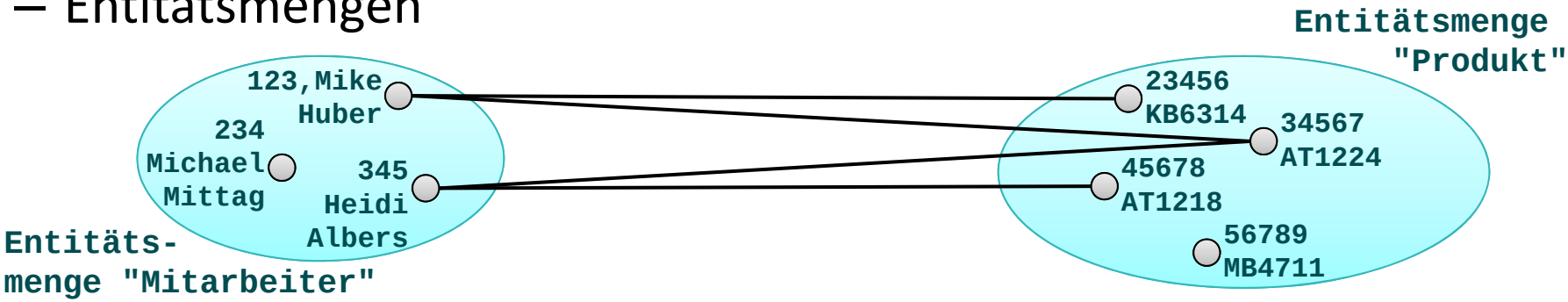
- 1:n Beziehung im relationalen Datenmodell
- n:m Beziehung im relationalen Datenmodell
- 1:1 Beziehung im relationalen Datenmodell
- Attribute von Beziehungen

und das Konzept des Fremdschlüssel.

Übergang vom ER-Modell zum Relationalen Modell

n:m Beziehungen am Beispiel "Mitarb. beraten zu Prod."

– Entitätsmengen



– ER-Modell

– Relationen

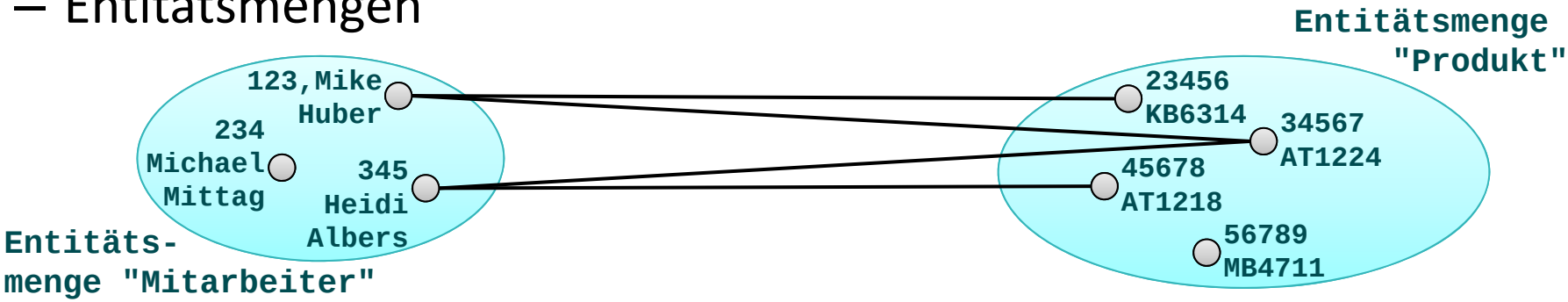
Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

n:m Beziehungen am Beispiel "Mitarb. beraten zu Prod."

– Entitätsmengen



– ER-Modell

– Relationen

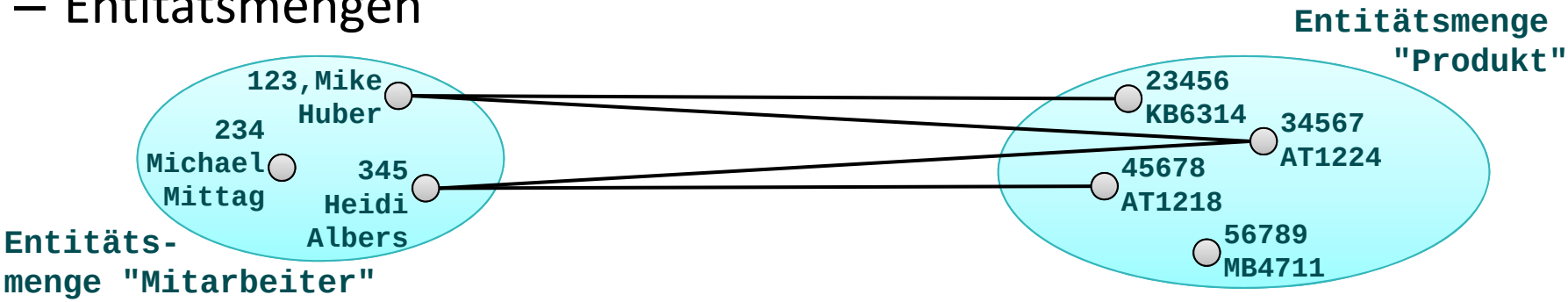
Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

n:m Beziehungen am Beispiel "Mitarb. beraten zu Prod."

– Entitätsmengen



– ER-Modell

– Relationen

Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

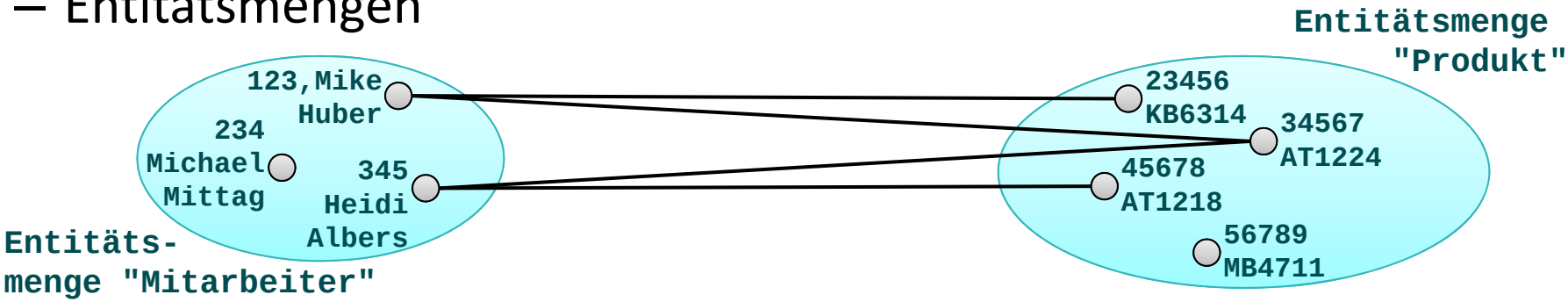
Beratung	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

n:m Beziehungen am Beispiel "Mitarb. beraten zu Prod."

– Entitätsmengen



– ER-Modell

– Relationen

Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

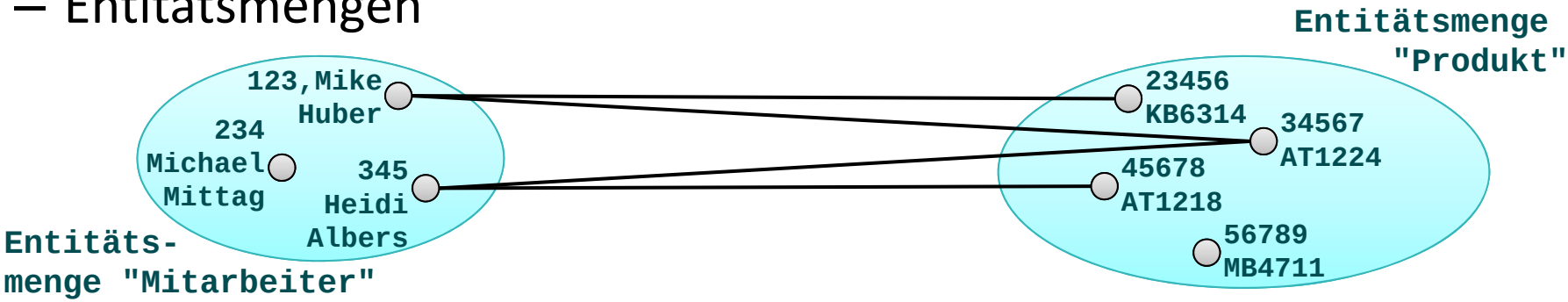
Beratung	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

n:m Beziehungen am Beispiel "Mitarb. beraten zu Prod."

– Entitätsmengen



– ER-Modell

– Relationen

Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

Beratung	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Beziehungen auf Relationen

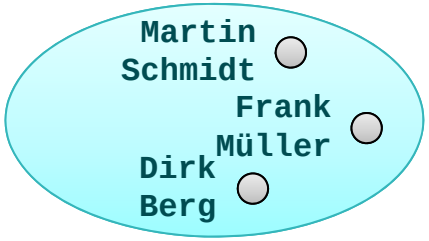
- 1:n Beziehung im relationalen Datenmodell
- n:m Beziehung im relationalen Datenmodell
- 1:1 Beziehung im relationalen Datenmodell
- Attribute von Beziehungen

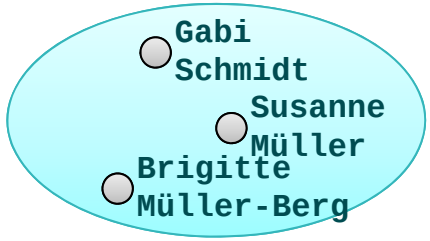
und das Konzept des Fremdschlüssel.

Übergang vom ER-Modell zum Relationalen Modell

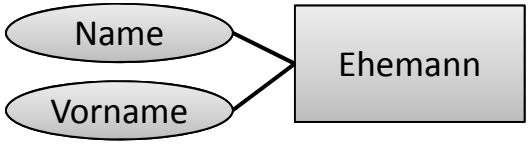
1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

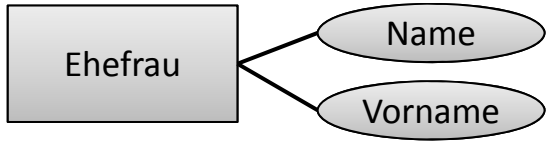
– Entitätsmengen





– ER-Modell





– Relationen

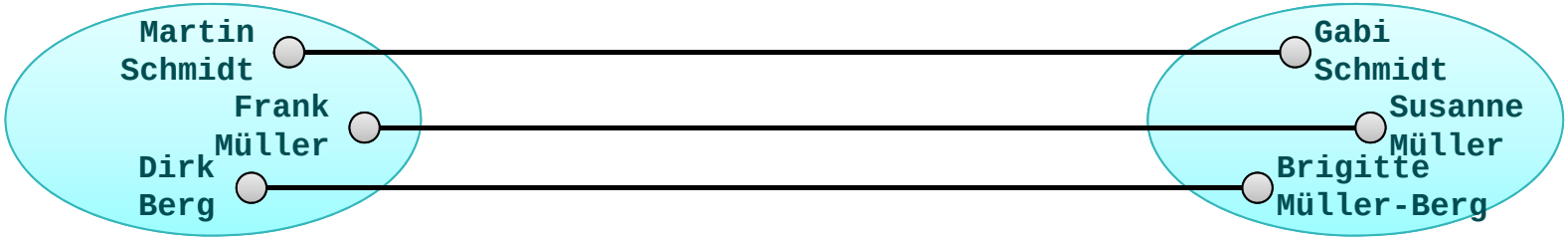
Ehemänner	Name	VName
	Schmidt	Martin
	Müller	Frank
	Berg	Dirk

Ehefrauen	Name	VName
	Schmidt	Gabi
	Müller	Susanne
	Müller-Berg	Brigitte

Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell

– Relationen

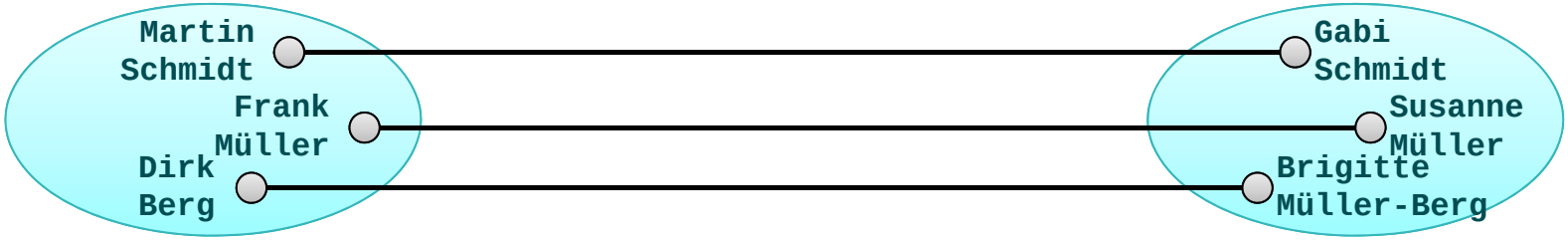
Ehemänner	Name	VName
	Schmidt	Martin
	Müller	Frank
	Berg	Dirk

Ehefrauen	Name	VName
	Schmidt	Gabi
	Müller	Susanne
	Müller-Berg	Brigitte

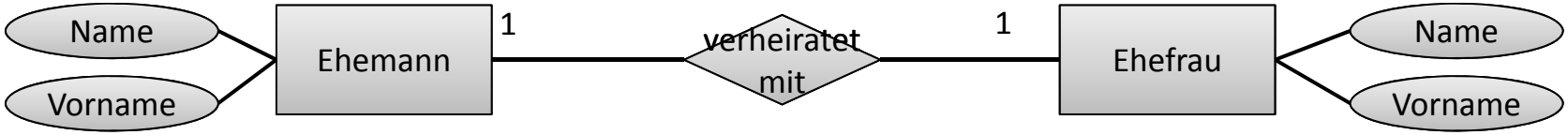
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen

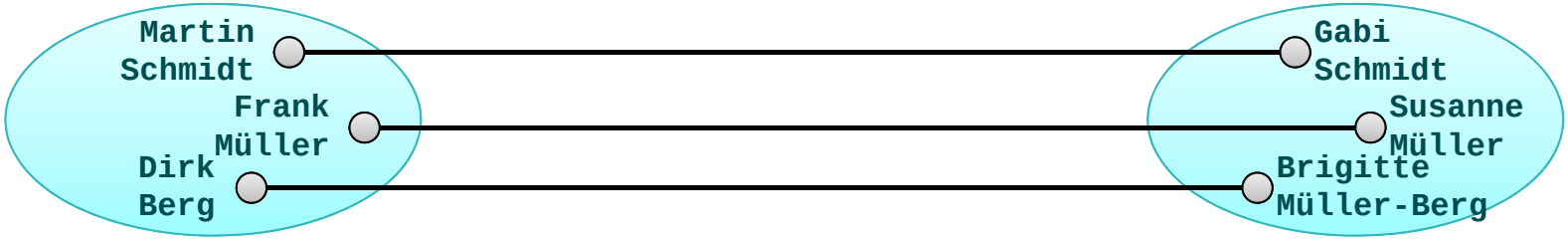
Ehemänner	Name	VName
	Schmidt	Martin
	Müller	Frank
	Berg	Dirk

Ehefrauen	Name	VName
	Schmidt	Gabi
	Müller	Susanne
	Müller-Berg	Brigitte

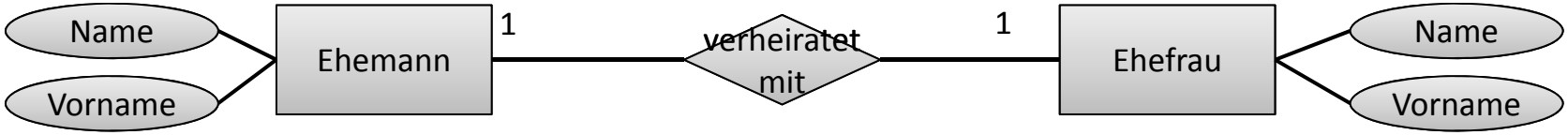
Übergang vom ER-Modell zum Relationalen Modells

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 1)

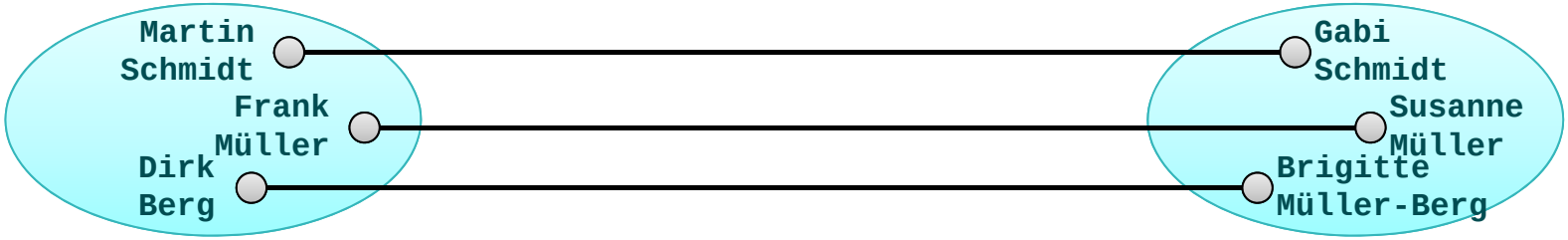
Ehema.	ID	Name	VName
	1	Schmidt	Martin
	2	Müller	Frank
	3	Berg	Dirk

Ehefr.	ID	Name	VName
	1	Schmidt	Gabi
	2	Müller	Susanne
	3	Müller-Berg	Brigitte

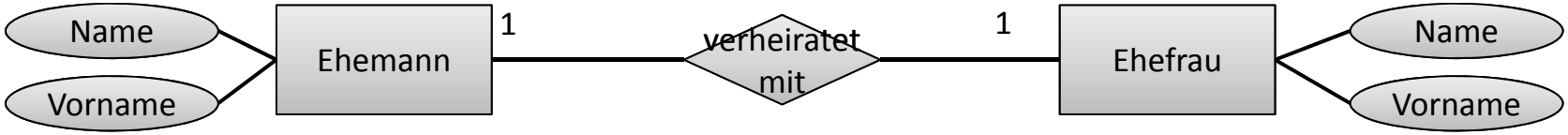
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 2a)

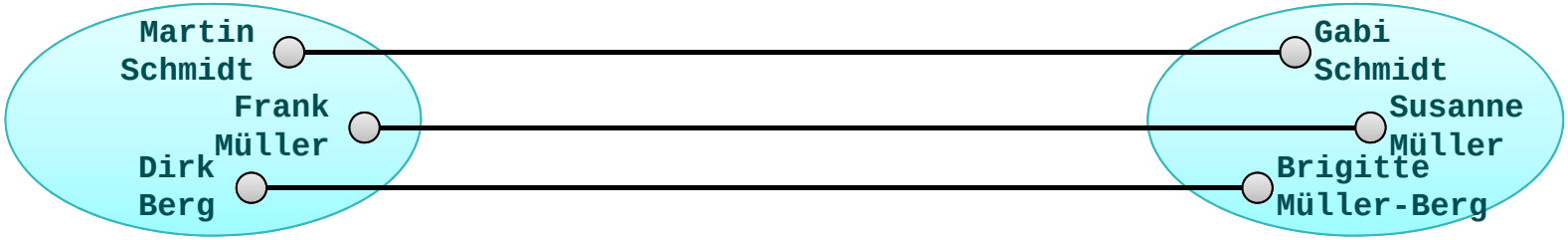
Ehema.	ID	Name	VName	Efr
	1	Schmidt	Martin	9
	2	Müller	Frank	8
	3	Berg	Dirk	7

Ehefr.	ID	Name	VName
	9	Schmidt	Gabi
	8	Müller	Susanne
	7	Müller-Berg	Brigitte

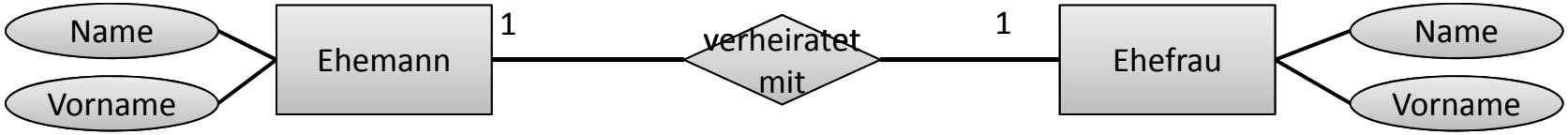
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 2b)

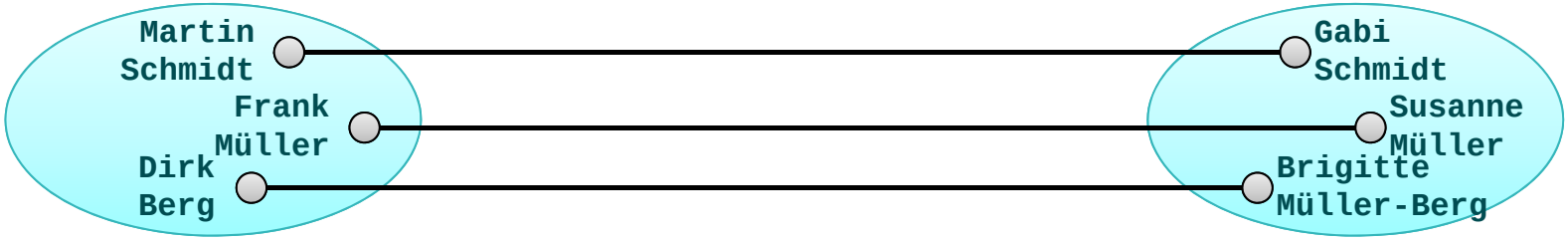
Ehema.	ID	Name	VName
	1	Schmidt	Martin
	2	Müller	Frank
	3	Berg	Dirk

Ehefr.	ID	Name	VName	Ema
	9	Schmidt	Gabi	1
	8	Müller	Susanne	2
	7	Müller-Berg	Brigitte	3

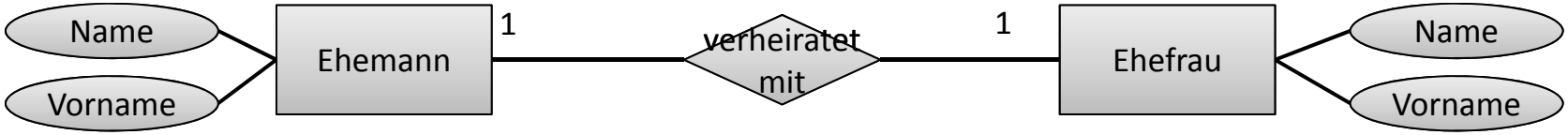
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 2c)

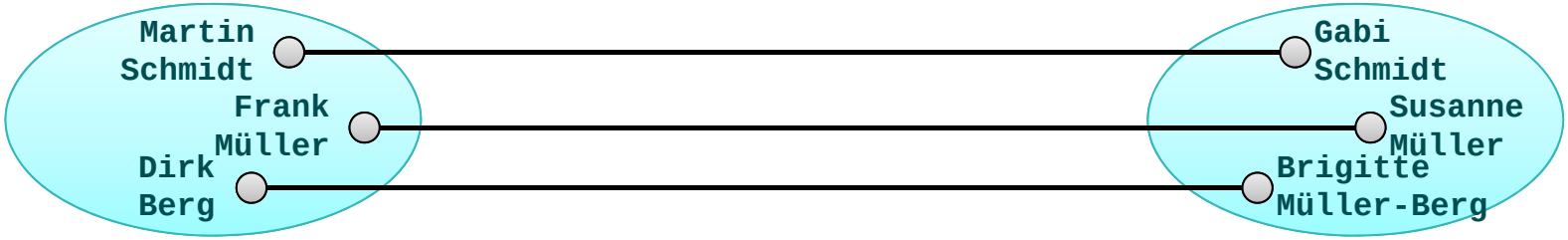
Ehema.	ID	Name	VName	Efr
	1	Schmidt	Martin	9
	2	Müller	Frank	8
	3	Berg	Dirk	7

Ehefr.	ID	Name	VName	Ema
	9	Schmidt	Gabi	1
	8	Müller	Susanne	2
	7	Müller-Berg	Brigitte	3

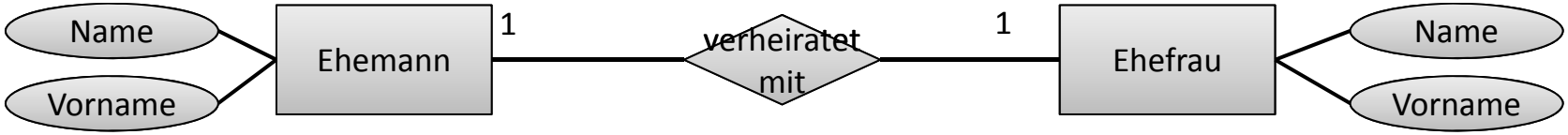
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 3)

Beide Spalten ohne Doppelte!

Ehema.	ID	Name	VName
	1	Schmidt	Martin
	2	Müller	Frank
	3	Berg	Dirk

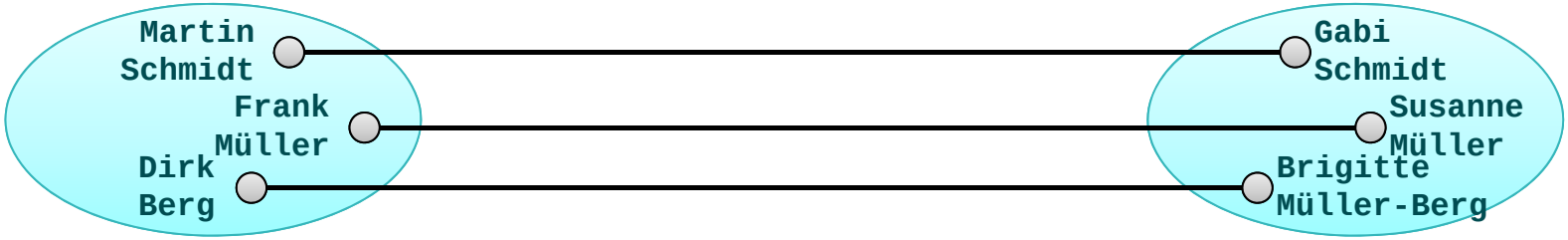
Ehe	Ema	Efr
	1	9
	2	8
	3	7

Ehefr.	ID	Name	VName
	9	Schmidt	Gabi
	8	Müller	Susanne
	7	Müller-Berg	Brigitte

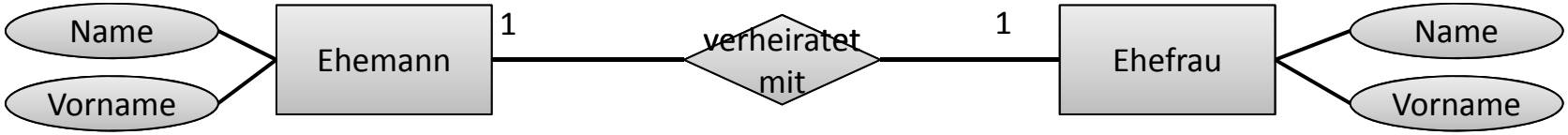
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 4)

Ehe	ID	MaName	MaVName	FrName	FrVName
	1	Schmidt	Martin	Schmidt	Gabi
	2	Müller	Frank	Müller	Susanne
	3	Berg	Dirk	Müller-Berg	Brigitte

Relationales Modell (Teil 2)

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen

– ER-Modell

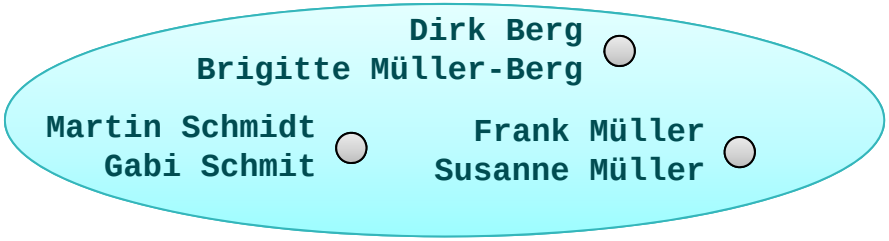
– Relationen (Variante 4)

Ehe	ID	MaName	MaVName	FrName	FrVName
	1	Schmidt	Martin	Schmidt	Gabi
	2	Müller	Frank	Müller	Susanne
	3	Berg	Dirk	Müller-Berg	Brigitte

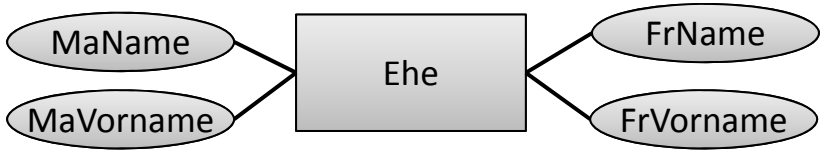
Übergang vom ER-Modell zum Relationalen Modell

1:1 Beziehungen am Beispiel "Ehemann und Ehefrau"

– Entitätsmengen



– ER-Modell



– Relationen (Variante 4)

Ehe	ID	MaName	MaVName	FrName	FrVName
	1	Schmidt	Martin	Schmidt	Gabi
	2	Müller	Frank	Müller	Susanne
	3	Berg	Dirk	Müller-Berg	Brigitte

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Beziehungen auf Relationen

- 1:n Beziehung im relationalen Datenmodell
- n:m Beziehung im relationalen Datenmodell
- 1:1 Beziehung im relationalen Datenmodell
- Attribute von Beziehungen

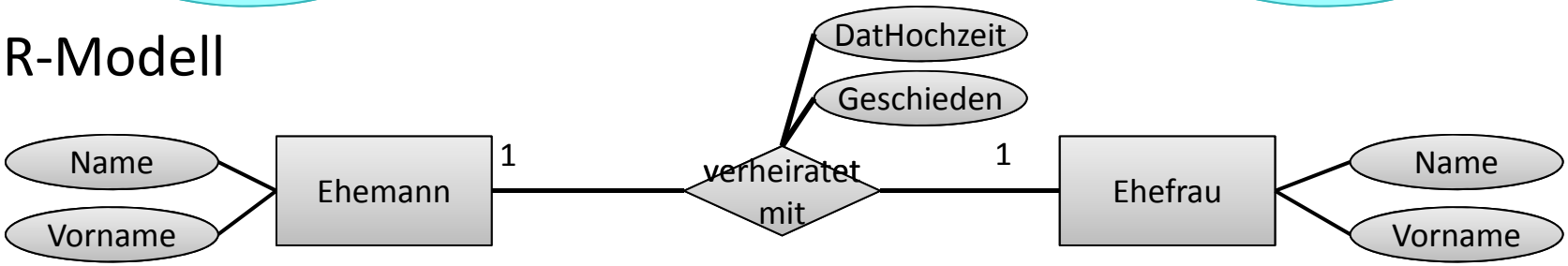
und das Konzept des Fremdschlüssel.

Übergang vom ER-Modell zum Relationalen Modell

Attribute von Beziehungen am Bsp "Ehemann&Ehefrau"

– Entitätsmengen

– ER-Modell



– Relationen

Ehema.	ID	Name	VName
	1	Schmidt	Martin
	2	Müller	Frank
	3	Berg	Dirk

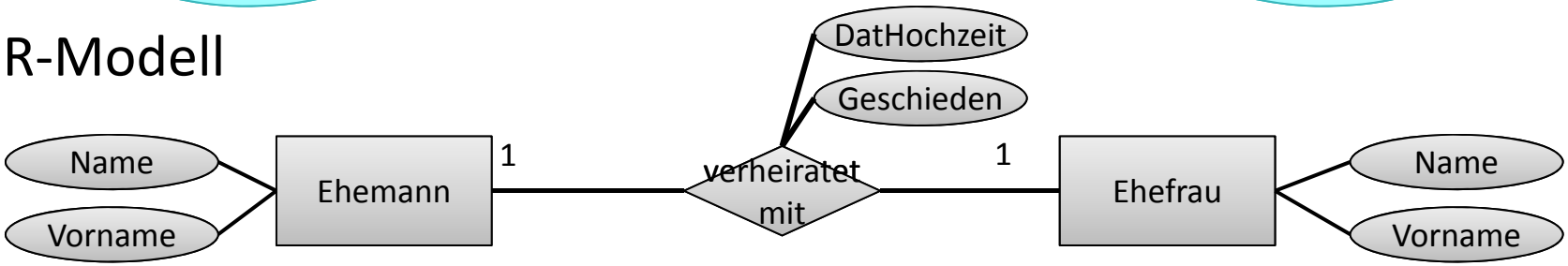
Ehefr.	ID	Name	VName
	9	Schmidt	Gabi
	8	Müller	Susanne
	7	Müller-Berg	Brigitte

Relationales Modell (Teil 2)

Attribute von Beziehungen am Bsp "Ehemann&Ehefrau"

– Entitätsmengen

– ER-Modell



– Relationen

Ehema.	ID	Name	VName
	1	Schmidt	Martin
	2	Müller	Frank
	3	Berg	Dirk

Ehe	Ema	Efr
	1	9
	2	8
	3	7

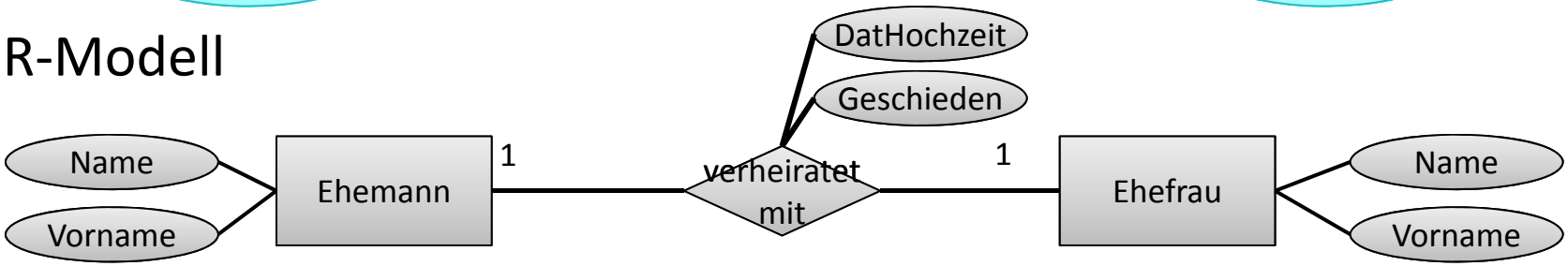
Ehefr.	ID	Name	VName
	9	Schmidt	Gabi
	8	Müller	Susanne
	7	Müller-Berg	Brigitte

Relationales Modell (Teil 2)

Attribute von Beziehungen am Bsp "Ehemann&Ehefrau"

– Entitätsmengen

– ER-Modell



– Relationen

Beide Spalten ohne Doppelte!

Ehema.	ID	Name	VName
	1	Schmidt	Martin
	2	Müller	Frank
	3	Berg	Dirk

Ehe	Ema	Efr	HDat	G
	1	9	12.1.80	J
	2	8	24.5.90	N
	3	7	23.4.92	N

Ehefr.	ID	Name	VName
	9	Schmidt	Gabi
	8	Müller	Susanne
	7	Müller-Berg	Brigitte

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Beziehungen auf Relationen

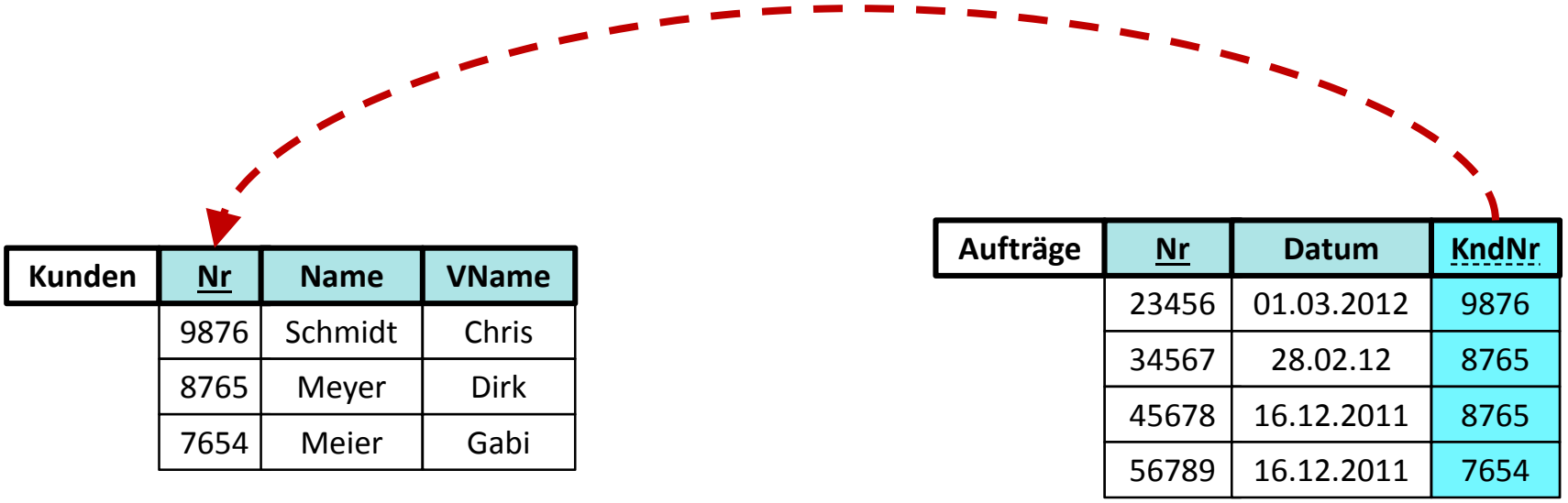
- 1:n Beziehung im relationalen Datenmodell
- n:m Beziehung im relationalen Datenmodell
- 1:1 Beziehung im relationalen Datenmodell
- Attribute von Beziehungen

und das Konzept des Fremdschlüssel.

Übergang vom ER-Modell zum Relationalen Modell

Fremdschlüssel

- Attribut bzw. Attribute, die auf den Primärschlüssel einer anderen Relation verweisen
- dient zur Umsetzung von Beziehungen zwischen Relationen
- Beispiel: Relation "Aufträge"
  - Fremdschlüssel KndNr verweist auf Primärschlüssel der Relation Kunden



Kunden	Nr	Name	VName
	9876	Schmidt	Chris
	8765	Meyer	Dirk
	7654	Meier	Gabi

Aufträge	Nr	Datum	KndNr
	23456	01.03.2012	9876
	34567	28.02.12	8765
	45678	16.12.2011	8765
	56789	16.12.2011	7654

Übergang vom ER-Modell zum Relationalen Modell

Fremdschlüssel

- Attribut bzw. Attribute, die auf den Primärschlüssel einer anderen Relation verweisen
- dient zur Umsetzung von Beziehungen zwischen Relationen
- Beispiel: Relation "Verkäufe"

Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

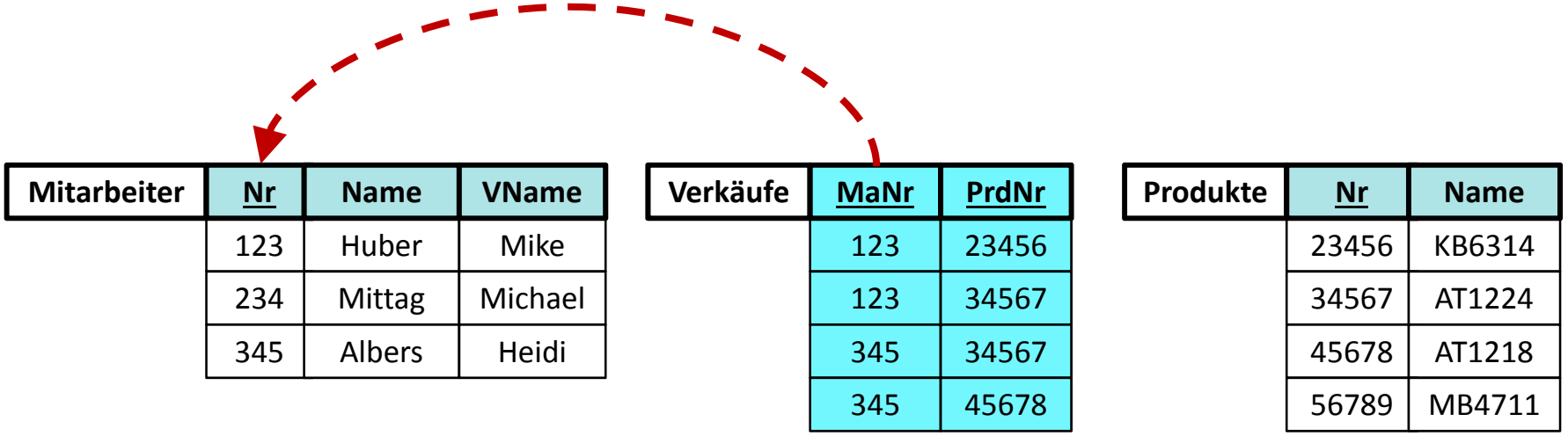
Verkäufe	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

Fremdschlüssel

- Attribut bzw. Attribute, die auf den Primärschlüssel einer anderen Relation verweisen
- dient zur Umsetzung von Beziehungen zwischen Relationen
- Beispiel: Relation "Verkäufe"
  - Fremdschlüssel MaNr verweist auf Primärschlüssel der Relation Mitarbeiter



Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

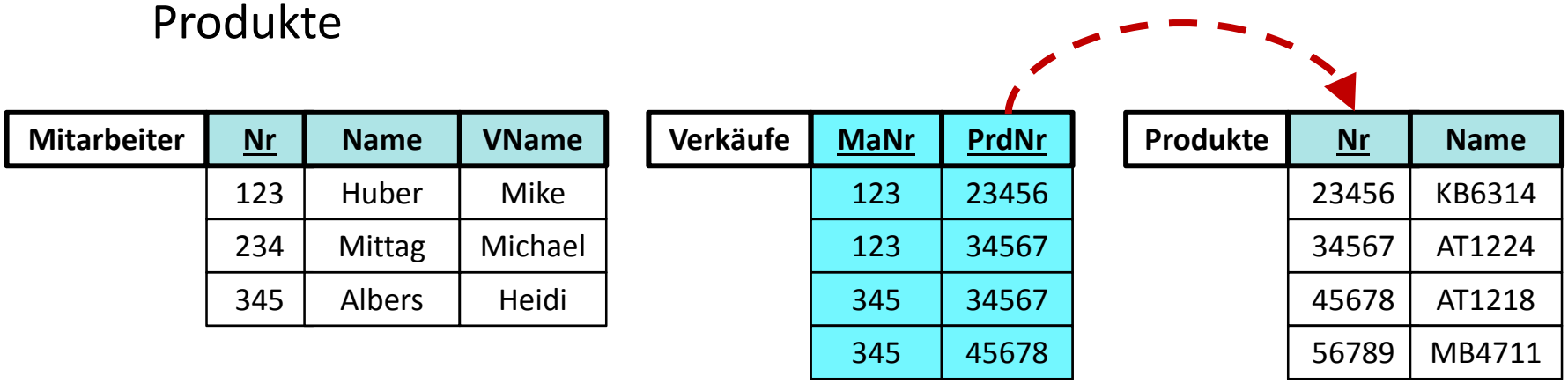
Verkäufe	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

Fremdschlüssel

- Attribut bzw. Attribute, die auf den Primärschlüssel einer anderen Relation verweisen
- dient zur Umsetzung von Beziehungen zwischen Relationen
- Beispiel: Relation "Verkäufe"
  - Fremdschlüssel MaNr verweist auf Primärschlüssel der Relation Mitarbeiter
  - Fremdschlüssel PrdNr verweist auf Primärschlüssel der Relation Produkte



Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

Verkäufe	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Übergang vom ER-Modell zum Relationalen Modell

Fremdschlüssel

- Attribut bzw. Attribute, die auf den Primärschlüssel einer anderen Relation verweisen
- dient zur Umsetzung von Beziehungen zwischen Relationen
- Beispiel: Relation "Verkäufe"
  - Fremdschlüssel MaNr verweist auf Primärschlüssel der Relation Mitarbeiter
  - Fremdschlüssel PrdNr verweist auf Primärschlüssel der Relation Produkte

Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi

Verkäufe	MaNr	PrdNr
	123	23456
	123	34567
	345	34567
	345	45678

Produkte	Nr	Name
	23456	KB6314
	34567	AT1224
	45678	AT1218
	56789	MB4711

Zusammengesetzter Primärschlüssel der Relation "Verkäufe"

Übergang vom ER-Modell zum Relationalen Modell

Abbildung von Beziehungen auf Relationen

- 1:n Beziehung im relationalen Datenmodell
- n:m Beziehung im relationalen Datenmodell
- 1:1 Beziehung im relationalen Datenmodell
- Attribute von Beziehungen

und das Konzept des Fremdschlüssel.

Integritätsregeln

1. Integritätsregel

- Kein Bestanteil eines Primärschlüssels darf leer sein.
- Der Primärschlüssel identifiziert jeden Datensatz eindeutig.

Integritätsregeln

1. Integritätsregel

- Kein Bestanteil eines Primärschlüssels darf leer sein.
- Der Primärschlüssel identifiziert jeden Datensatz eindeutig.

2. Integritätsregel (Referenzielle Integrität)

- Zu jedem Fremdschlüssel (außer dem leeren) existiert immer ein Wert im zugehörigen Primärschlüssel.

Integritätsregeln

1. Integritätsregel

- Kein Bestanteil eines Primärschlüssels darf leer sein.
- Der Primärschlüssel identifiziert jeden Datensatz eindeutig.

2. Integritätsregel (Referenzielle Integrität)

- Zu jedem Fremdschlüssel (außer dem leeren) existiert immer ein Wert im zugehörigen Primärschlüssel.

Auswirkungen

- Leerer Fremdschlüssel ist zulässig

Integritätsregeln

1. Integritätsregel

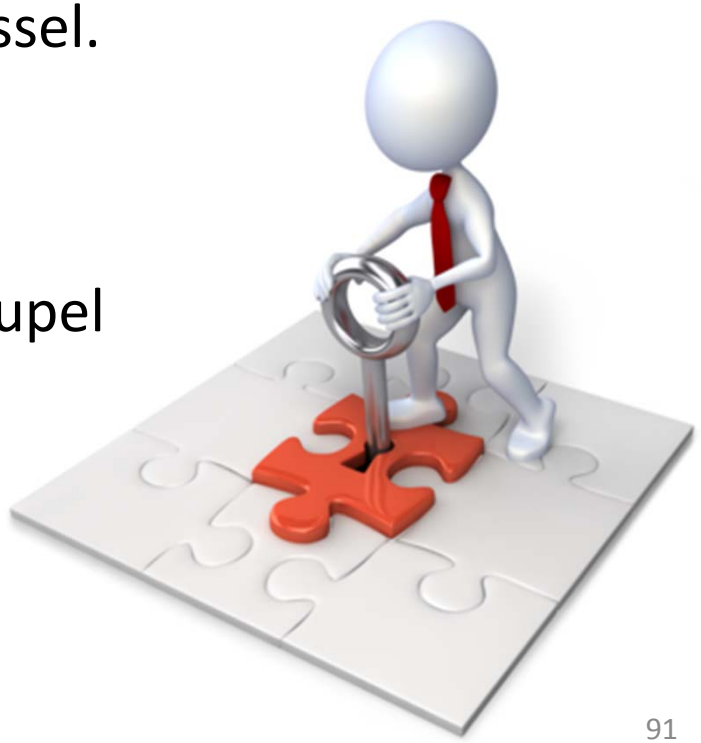
- Kein Bestanteil eines Primärschlüssels darf leer sein.
- Der Primärschlüssel identifiziert jeden Datensatz eindeutig.

2. Integritätsregel (Referenzielle Integrität)

- Zu jedem Fremdschlüssel (außer dem leeren) existiert immer ein Wert im zugehörigen Primärschlüssel.

Auswirkungen

- Leerer Fremdschlüssel ist zulässig
- Löschen/Ändern von referenzierten Tupel müssen berücksichtigt werden



Integritätsregeln

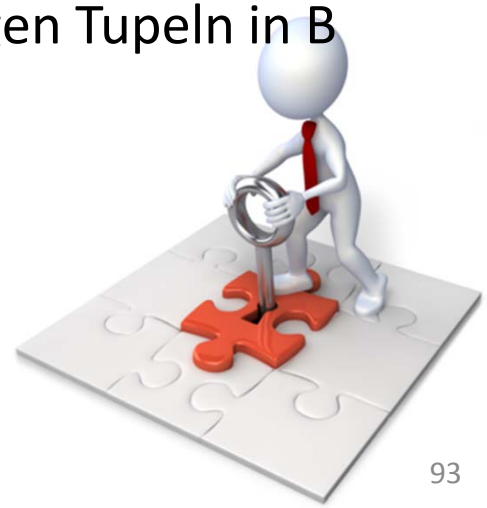
Auswirkung 1 – Leerer Fremdschlüssel

- Der Fremdschlüssel darf "leer" sein, was aber evtl. nicht immer fachlich gewünscht ist
- Beispiele
  - Benutzeraccount ohne Beziehung zum Kunden macht Sinn (z.B. für Admin)
  - Auftrag ohne Kunden macht keinen Sinn
- Es kann beim Entwurf der Relation (auch) für den Fremdschlüssel angegeben werden, dass er nicht leer bleiben darf

Integritätsregeln

Auswirkung 2 – Löschen/Ändern von referenzierten Tupeln

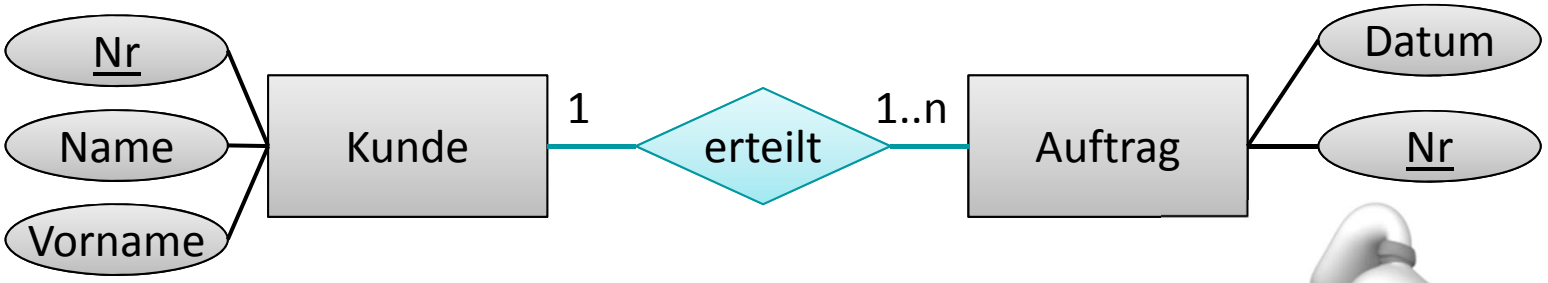
- Regel darf niemals (auch nicht kurzzeitig) verletzt werden
- Was tun, z.B. bei
  - Anlegen eines Auftrags, der immer einen Kunden erfordert
    ➔ Kunde muss bereits zuvor angelegt worden sein.
  - Löschens eines Kunden, für den es Aufträge gibt
    ➔ Auswirkungen auf die Aufträge müssen beachtet werden
  - Ändern des Primärschlüssels eines Kunden, für den es Aufträge gibt
    ➔ Auswirkungen auf die Aufträge müssen beachtet werden
- Beim Entwurf der Relationen A und B muss definiert werden, was bei Löschen/Ändern der Tupel in A, mit den zugehörigen Tupeln in B passieren soll
  - Zurückweisen des Lösch-/Änderungsversuch
  - Löschen/Ändern aller zugehörigen Aufträge
  - Leeren des Fremdschlüssels der Aufträge
  - Kaskadierendes Vorgehen als Spezialfall



Prüfungsvorbereitung

Beispielhafte Aufgaben

- Wie lautet die erste und wie die zweite Integritätsregel?
- Überführen Sie das dargestellte ER-Modell in ein Datenmodell. Füllen Sie diese mit geeigneten Beispieldaten, die die Kardinalität der Beziehung illustrieren.



- Welche Konsequenz der 2. Integritätsregel ist beim Löschen von Aufträgen (im Datenmodell aus der vorherigen Aufgabe) zu beachten? Welche Alternativen gäbe es und welche favorisieren Sie?

Inhalt

Ziel und Einordnung

Wiederholung

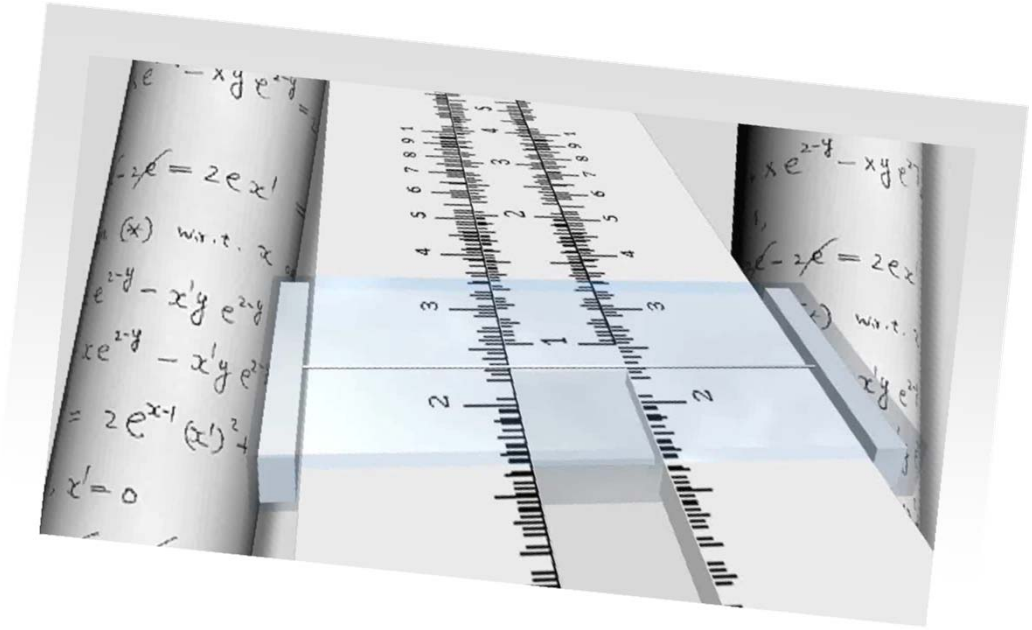
- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Arbeiten mit dem Relationalen Modell

Basis für Arbeit im relationalen Modell ist Relationenalgebra, u.a.

- Restriktion (Selektion)
- Projektion
- Join
- weitere, z.B.
  - Mengenoperation (Vereinigung, Schnitt, Differenz)



Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

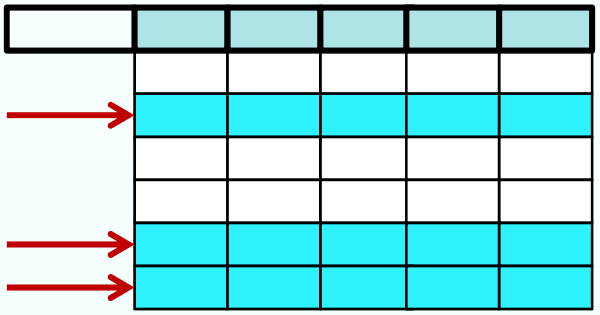
Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen



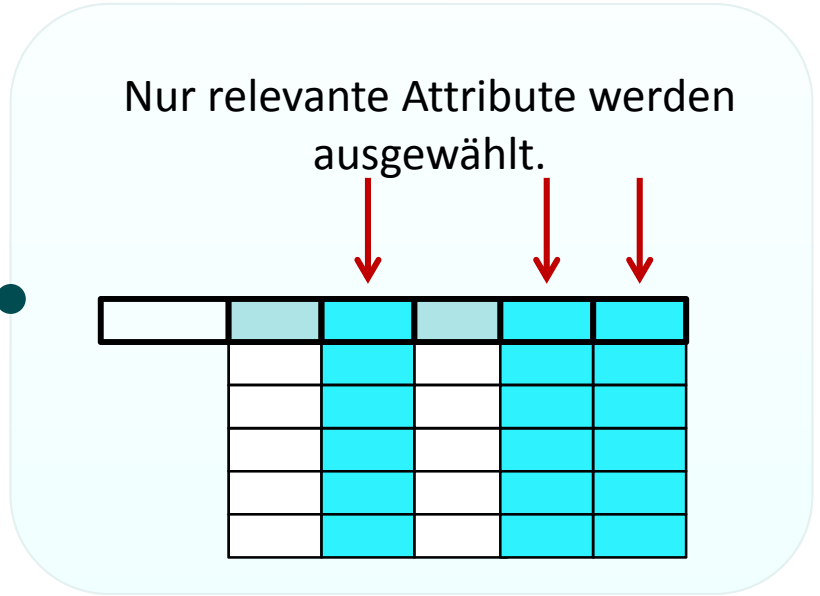
Nur relevante Tupel werden ausgewählt.

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen



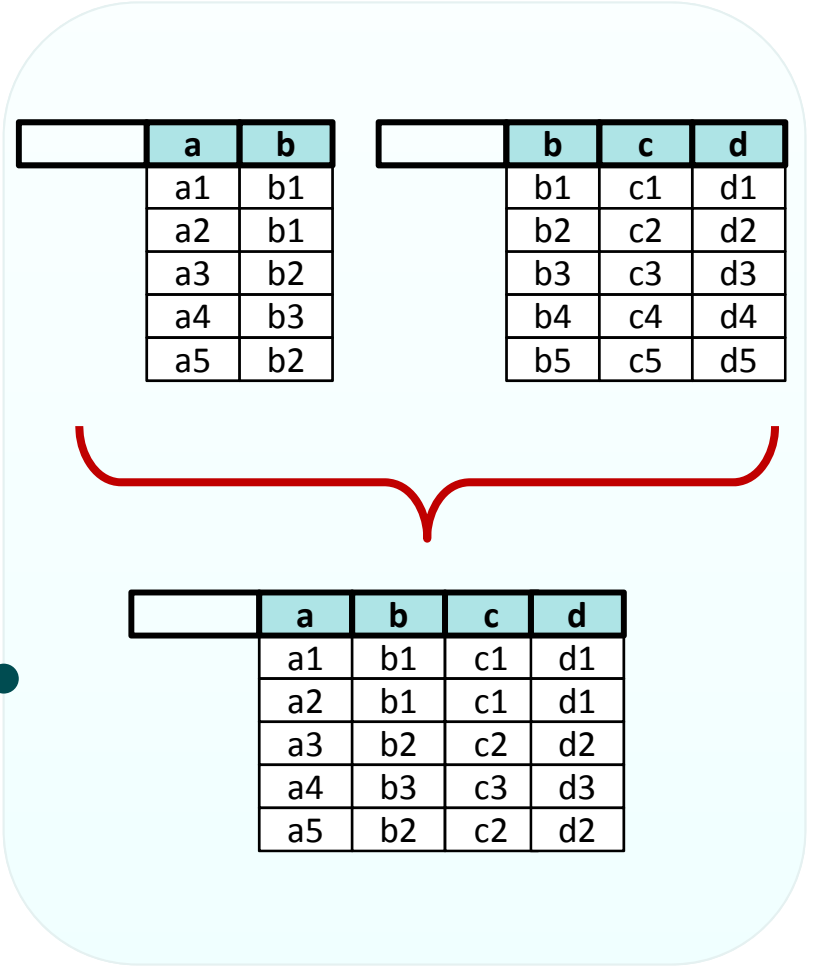
Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

...



	a	b
	a1	b1
	a2	b1
	a3	b2
	a4	b3
	a5	b2

	b	c	d
	b1	c1	d1
	b2	c2	d2
	b3	c3	d3
	b4	c4	d4
	b5	c5	d5

	a	b	c	d
	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

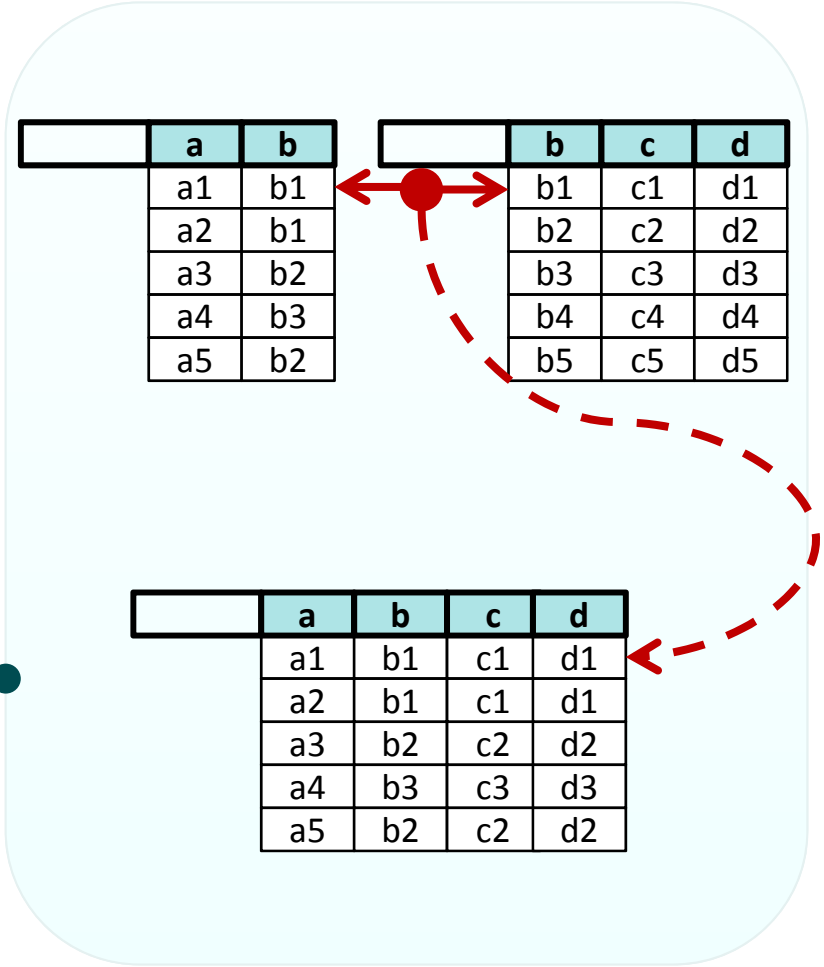
Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

...



	a	b
	a1	b1
	a2	b1
	a3	b2
	a4	b3
	a5	b2

	b	c	d
	b1	c1	d1
	b2	c2	d2
	b3	c3	d3
	b4	c4	d4
	b5	c5	d5

	a	b	c	d
	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

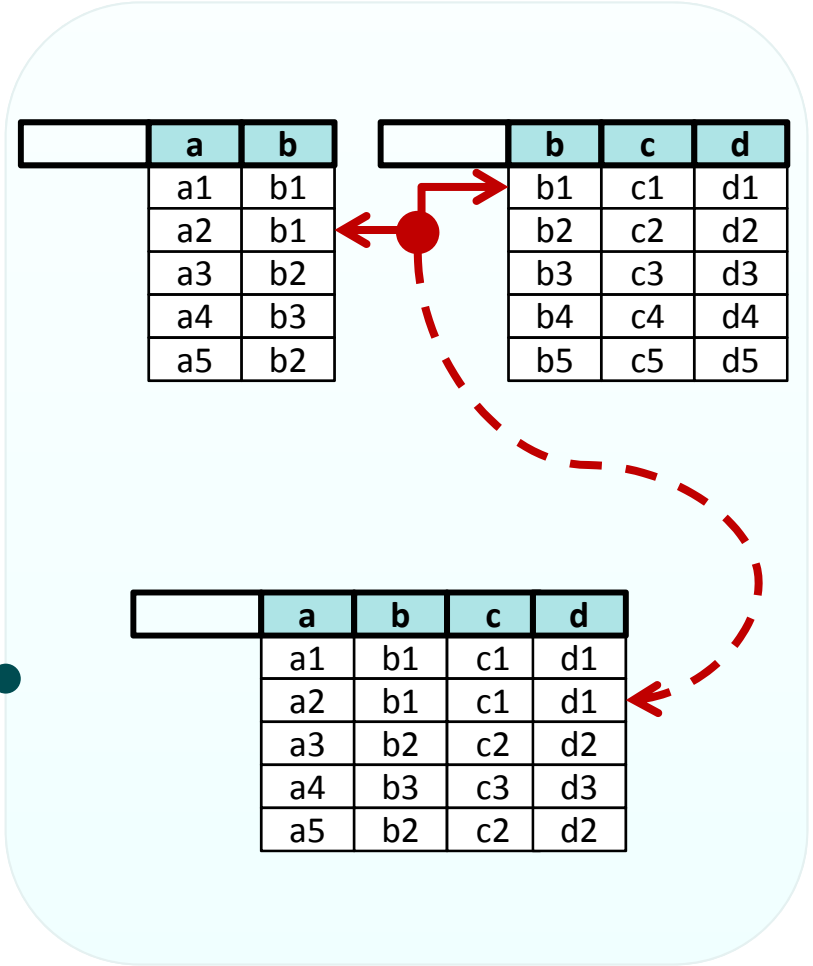
Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

...



	a	b
	a1	b1
	a2	b1
	a3	b2
	a4	b3
	a5	b2

	b	c	d
	b1	c1	d1
	b2	c2	d2
	b3	c3	d3
	b4	c4	d4
	b5	c5	d5

	a	b	c	d
	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

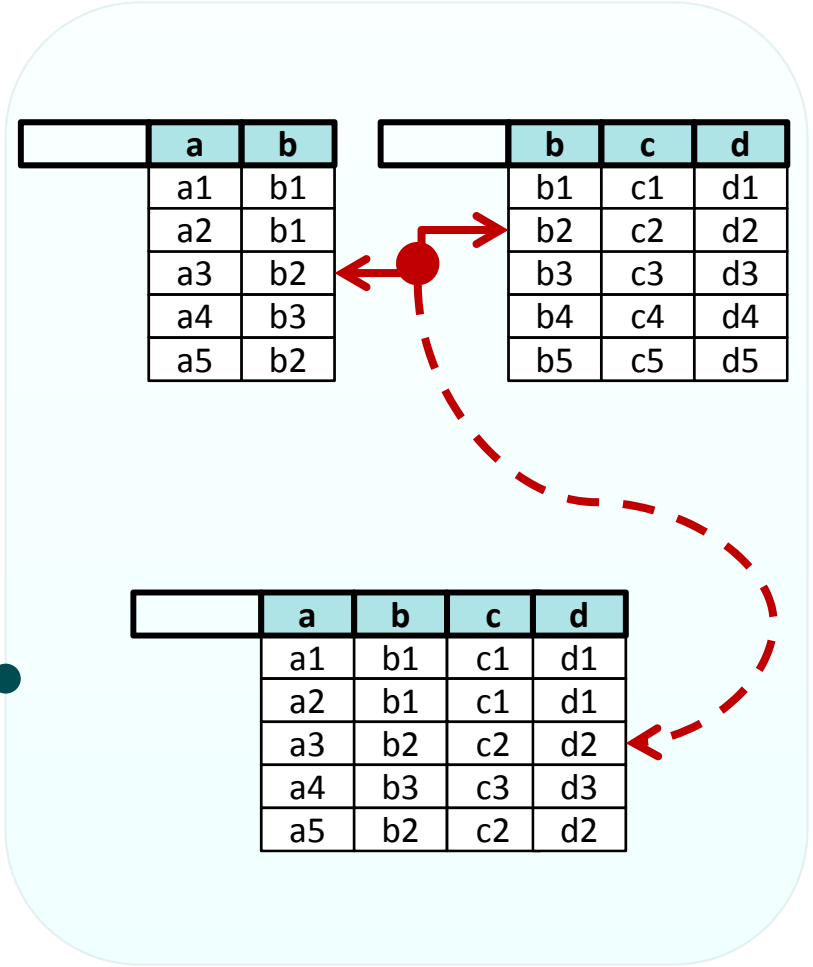
Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

...



	a	b
	a1	b1
	a2	b1
	a3	b2
	a4	b3
	a5	b2

	b	c	d
	b1	c1	d1
	b2	c2	d2
	b3	c3	d3
	b4	c4	d4
	b5	c5	d5

	a	b	c	d
	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

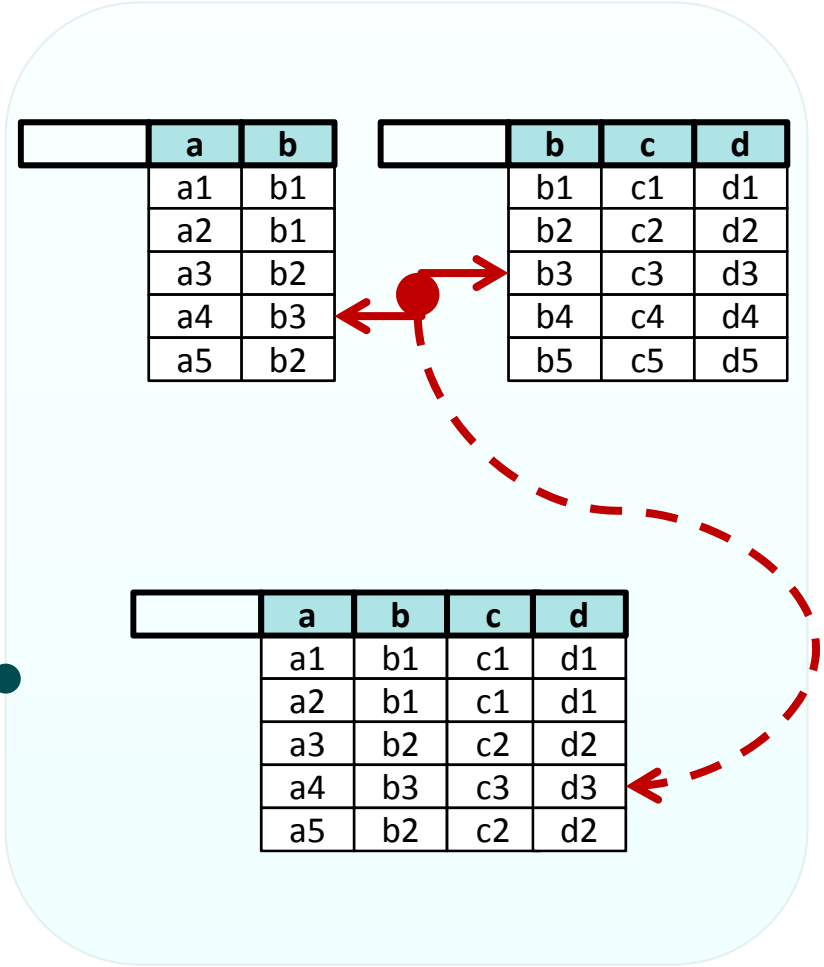
Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

...



	a	b
	a1	b1
	a2	b1
	a3	b2
	a4	b3
	a5	b2

	b	c	d
	b1	c1	d1
	b2	c2	d2
	b3	c3	d3
	b4	c4	d4
	b5	c5	d5

	a	b	c	d
	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

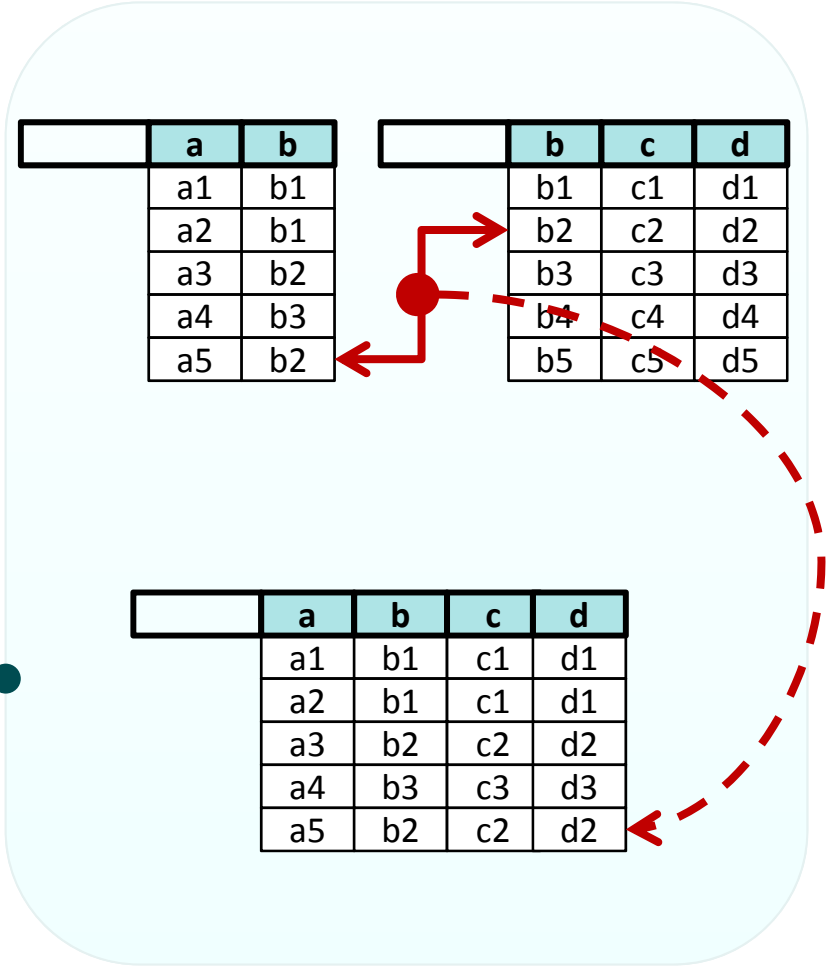
Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

...



	a	b
	a1	b1
	a2	b1
	a3	b2
	a4	b3
	a5	b2

	b	c	d
	b1	c1	d1
	b2	c2	d2
	b3	c3	d3
	b4	c4	d4
	b5	c5	d5

	a	b	c	d
	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

Theoretisches Fundament für Abfragesprachen relationaler Datenbanken

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

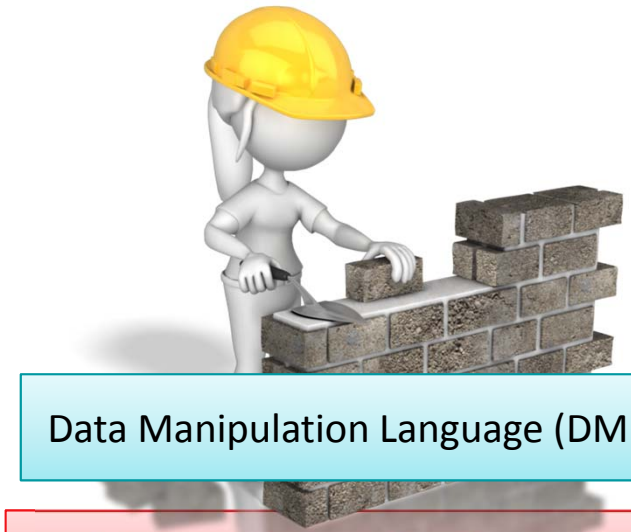
Theoretisches Fundament für Abfragesprachen relationaler Datenbanken

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen



Data Manipulation Language (DML)

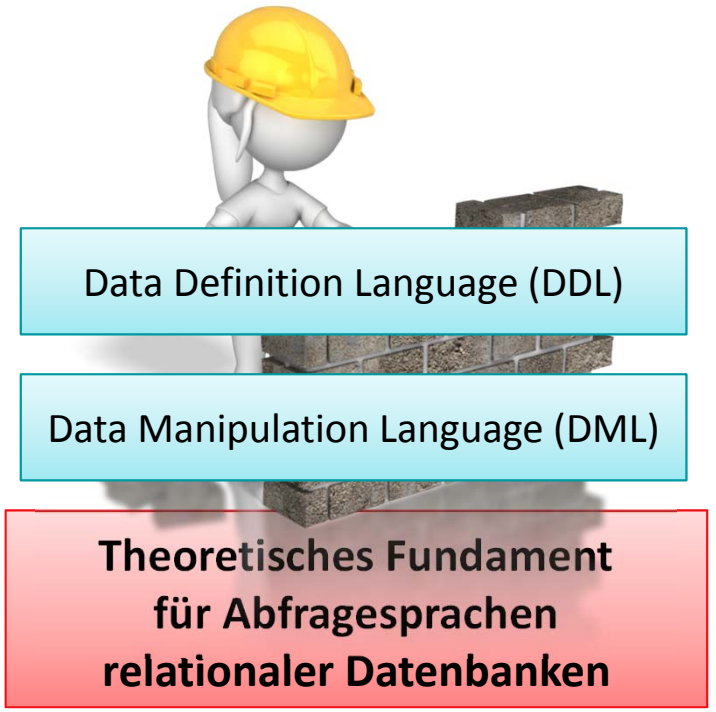
Theoretisches Fundament für Abfragesprachen relationaler Datenbanken

Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

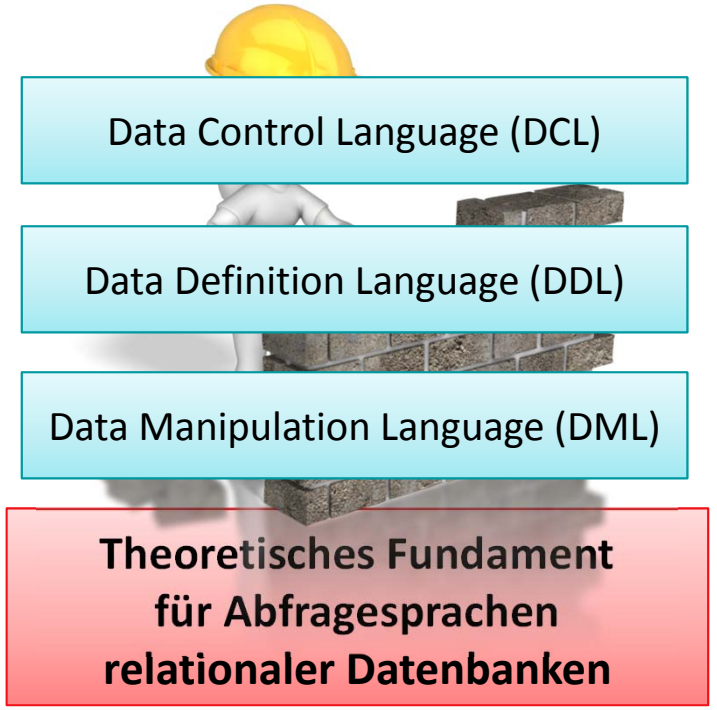


Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen

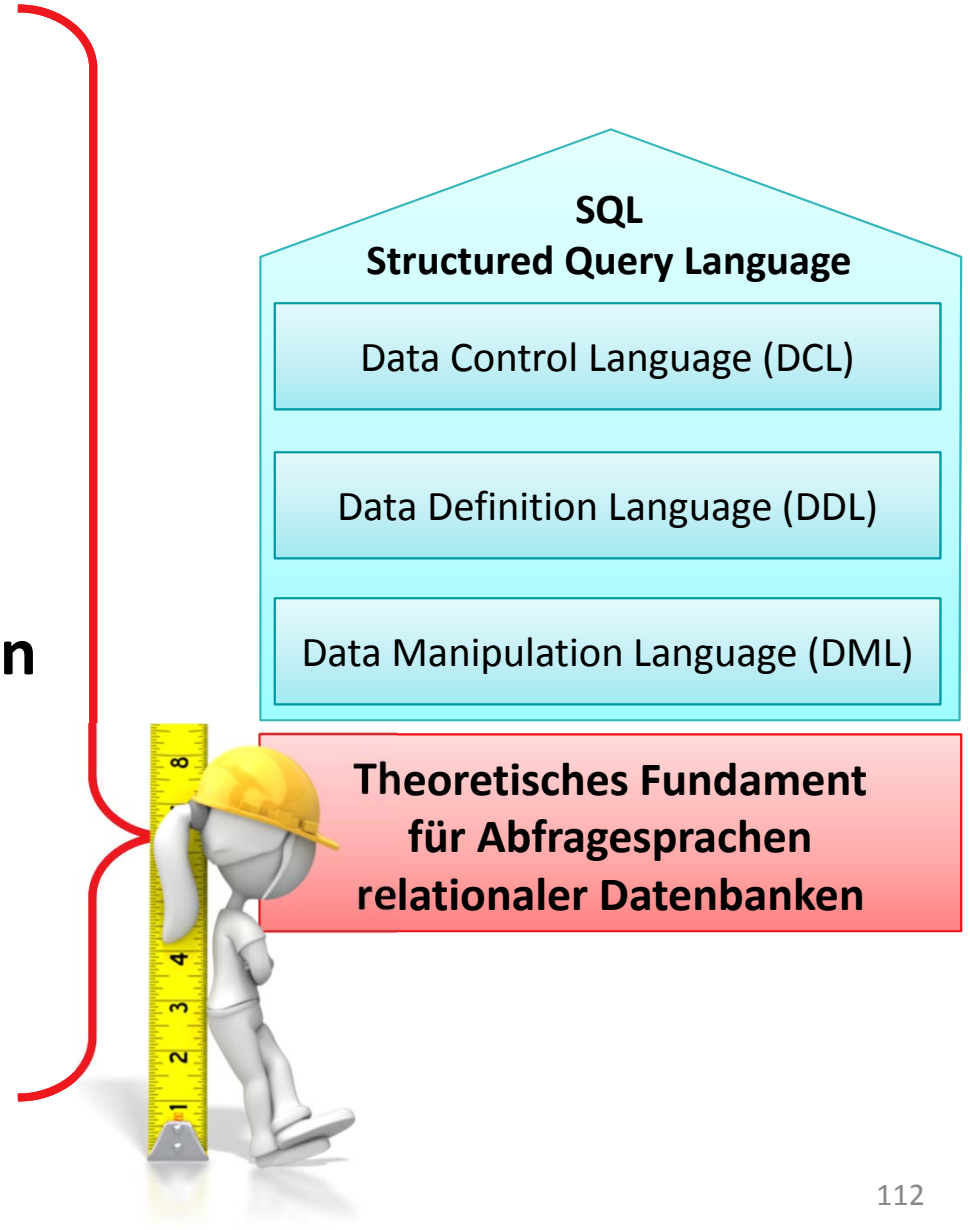


Operationen der Relationenalgebra

Restriktion: relevante Datensätze einer Relation auszuwählen und dabei ggf. zu sortieren

Projektion: relevante Attribute einer Relation auszuwählen

Join: verknüpfte Datensätzen mehrerer Relationen auszuwählen



Prüfungsvorbereitung

Beispielhafte Aufgaben

– Welche Operation der Relationenalgebra ist durch die dargestellten Tabellen illustriert?

a)

	a1	b1	c1	d1
	a3	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

b)

	a1	b1	c1	d1
	a2	b1	c1	d1
	a3	b2	c2	d2
	a4	b3	c3	d3
	a5	b2	c2	d2

Prüfungsvorbereitung

Beispielhafte Aufgaben

– Stellen Sie das Ergebnis einer Join-Operation der folgenden Tabellen auf Mitarbeiter.Nr = Verkauf.MNr dar, wobei das Attribut anhand dessen der Join ausgeführt wird, nur einmal in der Ergebnisrelation enthalten sein soll

Verkäufe	VNr	MNr	Produkt	Anzahl
	987	123	TV	1
	876	234	CD	23
	765	345	TV	2
	654	345	CD	12
	543	234	TV	4

Mitarbeiter	Nr	Name	VName
	123	Huber	Mike
	234	Mittag	Michael
	345	Albers	Heidi
	456	Huber	Harald

Inhalt

Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

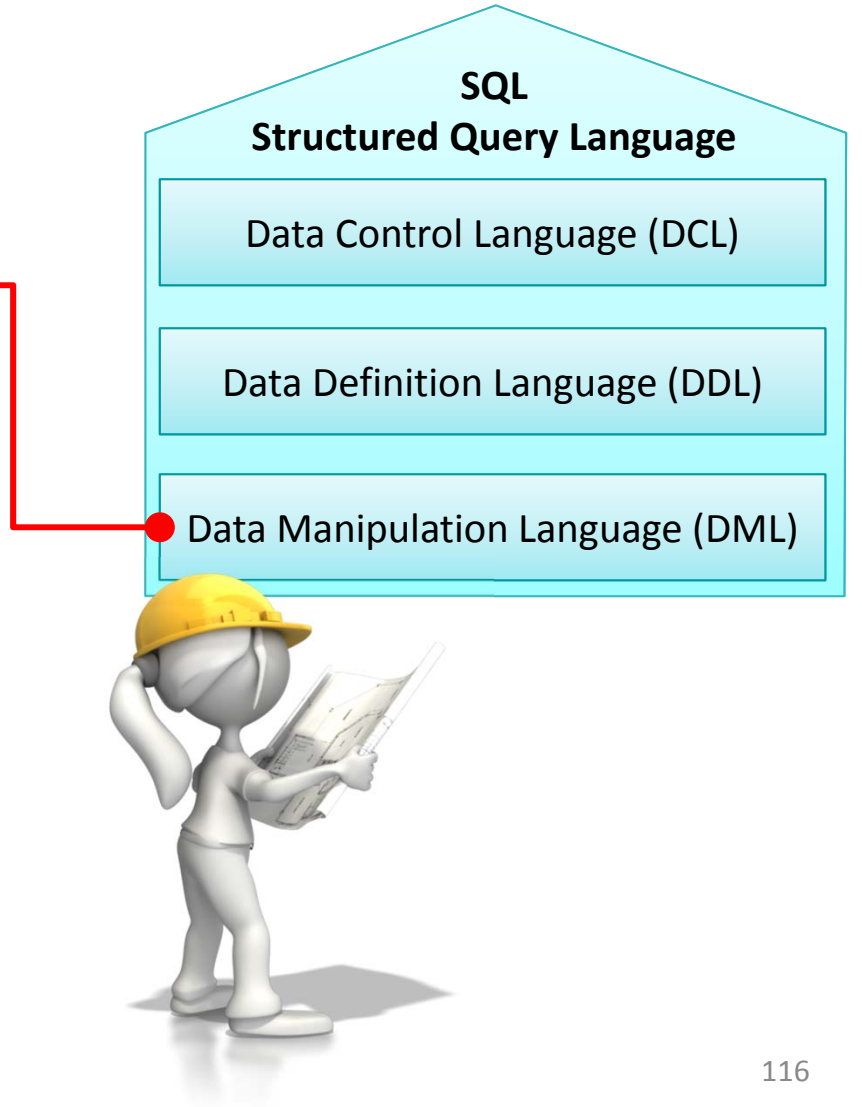
Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML): dient zur Abfrage, zum Hinzufügen, zur Veränderung und zum Löschen von Daten

- SELECT
- UPDATE
- INSERT
- DELETE

Data Definition Language (DDL)

Data Control Language (DCL)



SQL DML

Datensätze auswählen mit SQL

```
SELECT name, vorname FROM kunden WHERE
 ort='Berlin';
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname	Ort
	123	Albers	Willi	Aachen
	234	Boehrs	Vera	Berlin
	345	Dinkel	Ulrike	Berlin
	456	Dinkels	Thomas	Berlin
	567	Esser	Thomas	Dessau

SQL DML

Datensätze auswählen mit SQL

```
SELECT name, vorname FROM kunden WHERE
 ort='Berlin';
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname	Ort
	123	Albers	Willi	Aachen
	234	Boehrs	Vera	Berlin
	345	Dinkel	Ulrike	Berlin
	456	Dinkels	Thomas	Berlin
	567	Esser	Thomas	Dessau

Selektion/ Restriktion

Projektion

SQL DML

Datensätze auswählen mit SQL

```
SELECT name, vorname FROM kunden WHERE
 ort='Berlin';
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname	Ort
	123	Albers	Willi	Aachen
	234	Boehrs	Vera	Berlin
	345	Dinkel	Ulrike	Berlin
	456	Dinkels	Thomas	Berlin
	567	Esser	Thomas	Dessau

```
SELECT Name, Vorname FROM Kunden
WHERE ort='Berlin';
```

Ergebnis

SQL DML

Datensätze auswählen mit SQL

```
SELECT name, vorname FROM kunden WHERE
 ort='Berlin';
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname	Ort
	123	Albers	Willi	Aachen
	234	Boehrs	Vera	Berlin
	345	Dinkel	Ulrike	Berlin
	456	Dinkels	Thomas	Berlin
	567	Esser	Thomas	Dessau

```
SELECT Name, Vorname FROM Kunden
WHERE ort='Berlin';
```

Ergebnis

Name	Vorname
Boehrs	Vera
Dinkel	Ulrike
Dinkels	Thomas

SQL DML

Datensätze einfügen mit SQL

```
INSERT INTO kunden(Vorname, Name, KndNr)
 VALUES ('Simon', 'Jakob', 345);
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera

```
INSERT INTO kunden(Vorname, Name, KndNr)
VALUES ('Simon', 'Jakob', 345);
```

Ergebnisrelation

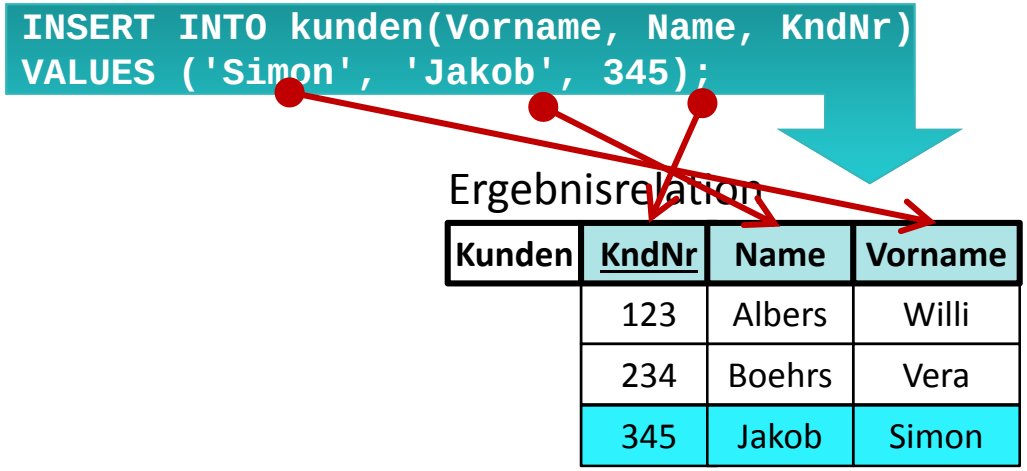
SQL DML

Datensätze einfügen mit SQL

```
INSERT INTO kunden(Vorname, Name, KndNr)
 VALUES ('Simon', 'Jakob', 345);
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera



SQL DML

Ändern von Datensätzen mit SQL

```
UPDATE kunden SET name='Albers'
 WHERE kndnr=234;
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera
	345	Muster	Michael

```
UPDATE kunden SET name='Albers'
WHERE kndnr=234;
```

Ergebnisrelation

SQL DML

Ändern von Datensätzen mit SQL

```
UPDATE kunden SET name='Albers'
 WHERE kndnr=234;
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera
	345	Muster	Michael

```
UPDATE kunden SET name='Albers'
WHERE kndnr=234;
```

Ergebnisrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Albers	Vera
	345	Muster	Michael

SQL DML

Löschen von Datensätzen mit SQL

```
DELETE FROM kunden WHERE name='Albers';
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Albers	Vera
	345	Muster	Michael

```
DELETE FROM kunden
WHERE name='Albers';
```

Ergebnisrelation

SQL DML

Löschen von Datensätzen mit SQL

```
DELETE FROM kunden WHERE name='Albers';
```

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Albers	Vera
	345	Muster	Michael

```
DELETE FROM kunden
WHERE name='Albers';
```

Ergebnisrelation

Kunden	KndNr	Name	Vorname
	345	Muster	Michael

SQL DML

Datensätze aus Tabellen mit INNER JOIN verbinden

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte INNER JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

Ausgangsrelationen

Produkte	PID	Name	LID
	123	Multi AB	987
	234	Flexi 123	987
	345	Mega+	876
	456	Super XL	

Lieferanten	ID	Firma	Ort
	987	Müller AG	Berlin
	876	Meier GmbH	Potsdam
	765	Bach&Sohn	Cottbus


```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte INNER JOIN lieferanten
ON produkte.lid = lieferanten.id;
```


Ergebnis

PID	Name	ID	Firma
123	Multi AB	987	Müller AG
234	Flexi 123	987	Müller AG
345	Mega+	876	Meier GmbH

SQL DML

Datensätze aus Tabellen mit LEFT OUTER JOIN verbinden

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte LEFT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

Ausgangsrelationen

Produkte	PID	Name	LID
	123	Multi AB	987
	234	Flexi 123	987
	345	Mega+	876
	456	Super XL	

Lieferanten	ID	Firma	Ort
	987	Müller AG	Berlin
	876	Meier GmbH	Potsdam
	765	Bach&Sohn	Cottbus

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte LEFT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

SQL DML

Datensätze aus Tabellen mit LEFT OUTER JOIN verbinden

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte LEFT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

Ausgangsrelationen

Produkte	PID	Name	LID
	123	Multi AB	987
	234	Flexi 123	987
	345	Mega+	876
	456	Super XL	

Lieferanten	ID	Firma	Ort
	987	Müller AG	Berlin
	876	Meier GmbH	Potsdam
	765	Bach&Sohn	Cottbus

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte LEFT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

Ergebnis

PID	Name	ID	Firma
123	Multi AB	987	Müller AG
234	Flexi 123	987	Müller AG
345	Mega+	876	Meier GmbH
456	Super XL		

SQL DML

Datensätze aus Tabellen mit RIGHT OUTER JOIN verbinden

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte RIGHT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

Ausgangsrelationen

Produkte	PID	Name	LID
	123	Multi AB	987
	234	Flexi 123	987
	345	Mega+	876
	456	Super XL	

Lieferanten	ID	Firma	Ort
	987	Müller AG	Berlin
	876	Meier GmbH	Potsdam
	765	Bach&Sohn	Cottbus


```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte RIGHT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

SQL DML

Datensätze aus Tabellen mit RIGHT OUTER JOIN verbinden

```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte RIGHT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

Ausgangsrelationen

Produkte	PID	Name	LID
	123	Multi AB	987
	234	Flexi 123	987
	345	Mega+	876
	456	Super XL	

Lieferanten	ID	Firma	Ort
	987	Müller AG	Berlin
	876	Meier GmbH	Potsdam
	765	Bach&Sohn	Cottbus


```
SELECT produkte.pid, produkte.name,
       lieferanten.id, lieferanten.firma
FROM produkte RIGHT JOIN lieferanten
ON produkte.lid = lieferanten.id;
```

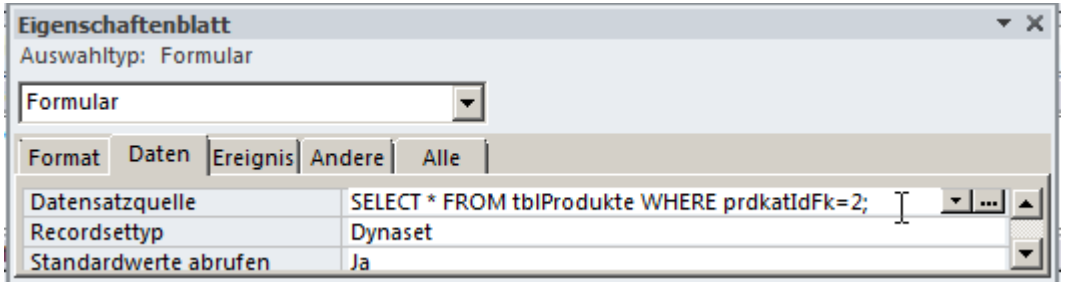

Ergebnis

PID	Name	ID	Firma
123	Multi AB	987	Müller AG
234	Flexi 123	987	Müller AG
345	Mega+	876	Meier GmbH
		765	Bach&Sohn

SQL DML

SQL in MS Access

– SQL-Abfragen können als Datenquelle für Formulare verwendet werden



– Einfügen, Ändern und Löschen mit SQL mit vordefinierter VBA-Funktionen

```
CurrentDB.Execute("<SQL-Anweisung>")
```

– Grafische Abfragen und SQL
  - werden von MS Access in SQL übersetzt
  - SQL-Ansicht zeigt das generierte SQL
  - Ausführung von SQL direkt über SQL-Ansicht eines leeren Abfrageentwurfs oder Abfrage eines entsprechenden Abfragetyps möglich

Prüfungsvorbereitung

Beispielhafte Aufgaben

– Schreiben Sie den einen zugehörigen SELECT-Befehl in SQL:

Ausgangsrelation:

Kunden	KndNr	Name	Ort	Alter
	123	Albers	Köln	18
	234	Berger	Köln	17
	345	Yilmaz	Berlin	17
	456	Schmidt	Berlin	32
	567	Steffen	München	17

Ergebnisrelation:

KndNr	Name	Alter	Ort
123	Albers	18	Köln
345	Yilmaz	17	Berlin
456	Schmidt	32	Berlin

– Schreiben Sie die zugehörigen SQL-Befehle:

Ausgangsrelation:

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Albers	Vera
	345	Muster	Michael

Ergebnisrelation:

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Berger	Vera
	567	Schuster	Dirk

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bisher in folgendem Format benutzt

```
SELECT <Spalte1>,
       <Spalte2>
FROM <Tabelle>
  WHERE <Spalte> = <Bedingung>;
```

Beispiele

```
SELECT kndName, kndVorname, kndOrt FROM tblKunden
  WHERE kndOrt = 'Berlin';
```

```
SELECT * FROM tblProdukte
  WHERE prdPreis > 200;
```

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Sortierung

```
SELECT <Spalte1>,
       <Spalte2>
FROM <Tabelle>
 ORDER BY <Spalte> [ASC|DESC];
```

Beispiele

```
SELECT kndName, kndVorname, kndOrt
FROM tblKunden
  ORDER BY kndOrt DESC;
```

```
SELECT prdBezeichnung, prdPreis
FROM tblProdukte
  ORDER BY prdPreis DESC;
```

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Statistikfunktionen

```
SELECT <Funktion> AS <Bezeichnung>
FROM <Tabelle>
  WHERE <Spalte> = <Bedigung>;
```

Funktion	Bezeichnung	Erläuterung
AVG()	Durchschnitt (Average)	Durchschnittswert, ermittelt über alle Zeilen des SELECT-Ergebnisses
COUNT()	Anzahl (Count)	Anzahl aller Zeilen des SELECT-Ergebnisses
MAX()	Maximum	Maximalwert aller Zeilen des SELECT-Ergebnisses
MIN()	Minimum	Minimalwert aller Zeilen des SELECT-Ergebnisses
SUM()	Summe	Summe, ermittelt über alle Zeilen des SELECT-Ergebnisses

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Statistikfunktionen

```
SELECT <Funktion> AS <Bezeichnung>
FROM <Tabelle>
  WHERE <Spalte> = <Bedigung>;
```

Beispiele

```
SELECT COUNT(*) AS Anzahl
FROM tblKunden
  WHERE kndOrt = 'Berlin';
```

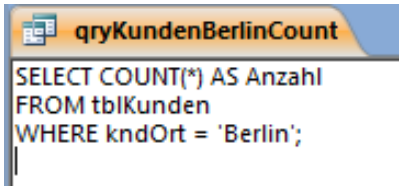
Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Statistikfunktionen

```
SELECT <Funktion> AS <Bezeichnung>
FROM <Tabelle>
  WHERE <Spalte> = <Bedigung>;
```

Beispiele

```
SELECT COUNT(*) AS Anzahl
FROM tblKunden
  WHERE kndOrt = 'Berlin';
```



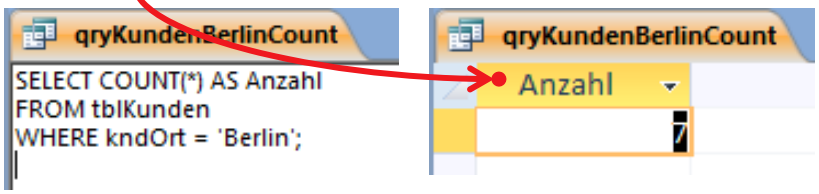
Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Statistikfunktionen

```
SELECT <Funktion> AS <Bezeichnung>
FROM <Tabelle>
  WHERE <Spalte> = <Bedigung>;
```

Beispiele

```
SELECT COUNT(*) AS Anzahl
FROM tblKunden
  WHERE kndOrt = 'Berlin';
```



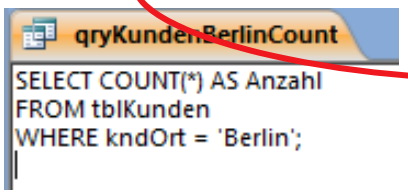
Sortieren, Gruppieren und Summen mit SQL

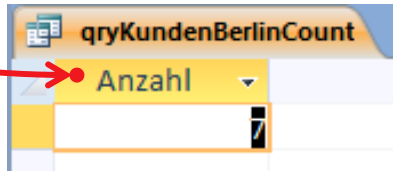
SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Statistikfunktionen

```
SELECT <Funktion> AS <Bezeichnung>
FROM <Tabelle>
  WHERE <Spalte> = <Bedigung>;
```

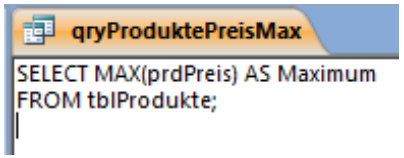
Beispiele

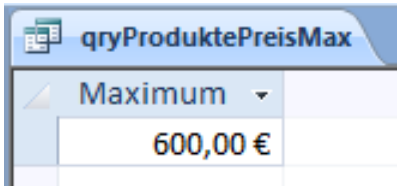
```
SELECT COUNT(*) AS Anzahl
FROM tblKunden
  WHERE kndOrt = 'Berlin';
```






```
SELECT MAX(prdPreis) AS Maxi
FROM tblProdukte;
```





Sortieren, Gruppieren und Summen mit SQL

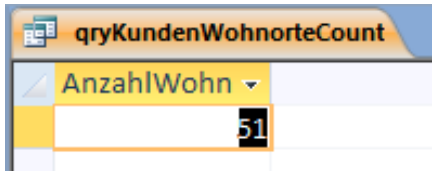
SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Statistikfunktionen

```
SELECT <Funktion> AS <Bezeichnung>
FROM <Tabelle>
  WHERE <Spalte> = <Bedigung>;
```

Beispiele

Nullwerte (leer) in Spalte kndOrt werden nicht gezählt

```
SELECT COUNT(kndOrt) AS AnzahlWohnorte
FROM tblKunden;
```




```
SELECT COUNT(DISTINCT kndOrt) AS AnzWOrte
FROM tblKunden;
```

Ohne doppelte, d.h. jeder Ort wird nur einmal gezählt

In Access nicht möglich

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Unterabfragen

```
SELECT <Funktion> AS <Bezeichnung>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

```
SELECT <Spalte 1> , <Spalte 2>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

Beispiele

```
SELECT COUNT(kndOrt) AS AnzahlWohnorte
FROM

  (SELECT DISTINCT kndOrt
   FROM tblKunden) AS Unterabfrage;
```

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Unterabfragen

```
SELECT <Funktion> AS <Bezeichnung>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

```
SELECT <Spalte 1> , <Spalte 2>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

Beispiele

```
SELECT COUNT(kndOrt) AS AnzahlWohnorte
FROM
  (SELECT DISTINCT kndOrt
   FROM tblKunden) AS Unterabfrage;
```

Ohne doppelte, d.h. jeder Ort nur einmal

Sortieren, Gruppieren und Summen mit SQL

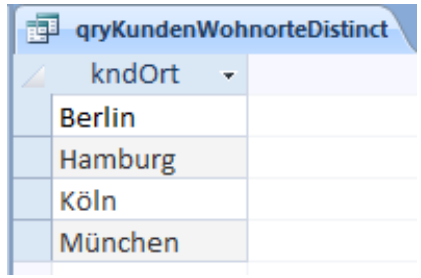
SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Unterabfragen

```
SELECT <Funktion> AS <Bezeichnung>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

```
SELECT <Spalte 1> , <Spalte 2>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

Beispiele

```
SELECT COUNT(kndOrt) AS AnzahlWohnorte
FROM

  (SELECT DISTINCT kndOrt
   FROM tblKunden) AS Unterabfrage;
```



Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Unterabfragen

```
SELECT <Funktion> AS <Bezeichnung>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

```
SELECT <Spalte 1> , <Spalte 2>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

Beispiele

```
SELECT COUNT(kndOrt) AS AnzahlWohnorte
FROM

  (SELECT DISTINCT kndOrt
   FROM tblKunden) AS Unterabfrage;
```

qryKundenWohnorteDistinct

kndOrt
Berlin
Hamburg
Köln
München

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Unterabfragen

```
SELECT <Funktion> AS <Bezeichnung>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

```
SELECT <Spalte 1> , <Spalte 2>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

Beispiele

```
SELECT COUNT(kndOrt) AS AnzahlWohnorte
FROM

  (SELECT DISTINCT kndOrt
   FROM tblKunden) AS Unterabfrage;
```

qryKundenWohnorteCou

AnzahlWohnorte
4

qryKundenWohnorteDistinct

kndOrt
Berlin
Hamburg
Köln
München

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Unterabfragen

```
SELECT <Funktion> AS <Bezeichnung>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

```
SELECT <Spalte 1> , <Spalte 2>
FROM
      (<Unterabfrage>) AS <Bezeichnung>;
```

Beispiele

```
SELECT TOP 5 *
FROM

 (SELECT *
  FROM tblProdukte
   ORDER BY prdPreis DESC)
  AS Unterabfrage;
```

prdIdPk	prdBezeichr	prdPreis	prdpktIdFk
91	Rasenmäher "I	600,00 €	2
19	Pavilion "Bern	600,00 €	1
21	Pavilion "Bern	600,00 €	1
23	Pavilion "Tren	600,00 €	1
24	Pavilion "Tren	600,00 €	1

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Gruppierung und Mehrfachgruppierung

```
SELECT <Spalte1> AS <Bezeichnung>,
       <Spalte2> AS <Bezeichnung>,
      <Funktion> AS <Bezeichnung>
FROM  <Tabelle>|(<Unterabfrage>) AS <Bezeichnung>
 GROUP BY <Spalte1>, <Spalte2>;
 HAVING <Bedingung>
```

Beispiele

```
SELECT kndOrt, COUNT(*) AS KundenImOrt
FROM tblKunden
 GROUP BY kndOrt;
```

kndOrt	KundenImO
Berlin	7
Hamburg	14
Köln	12
München	18

```
SELECT kndOrt, COUNT(*) AS KundenImOrt
FROM tblKunden
 GROUP BY kndOrt
 HAVING COUNT(*) > 7;
```

kndOrt	KundenImO
Hamburg	14
Köln	12
München	18

Sortieren, Gruppieren und Summen mit SQL

SQL-Select-Anweisung bietet weitere Möglichkeiten, hier bspw. Gruppierung und Mehrfachgruppierung

```
SELECT <Spalte1> AS <Bezeichnung>,
       <Spalte2> AS <Bezeichnung>,
      <Funktion> AS <Bezeichnung>
FROM  <Tabelle>|(<Unterabfrage>) AS <Bezeichnung>
 GROUP BY <Spalte1>, <Spalte2>;
 HAVING <Bedingung>
```

Beispiele

```
SELECT kndOrt,
       kndStrasse,
       COUNT(*) AS KundenInOrtStr
FROM tblKunden
  GROUP BY kndOrt,
           kndStrasse;
```

qryKundenGruppiertOrtStrasse

kndOrt	kndStrasse	KundenInOr
Berlin	Bahnhofstraße	3
Berlin	Dorfstraße	1
Berlin	Gartenstraße	1
Berlin	Hauptstraße	1
Berlin	Schulstraße	1
Hamburg	Bahnhofstraße	3
Hamburg	Dorfstraße	4
Hamburg	Gartenstraße	1
Hamburg	Hauptstraße	2
Hamburg	Schulstraße	4
Köln	Bahnhofstraße	2
Köln	Dorfstraße	1
Köln	Gartenstraße	2

Prüfungsvorbereitung

Beispielhafte Aufgaben

– Wie lautet die SQL-Anweisung, mit der die Telefonnummer von Silke Müller in der Spalte kndTelefon einer gegebenen Tabelle tblKunden in 030 9876543 geändert werden kann?

Prüfungsvorbereitung

Beispielhafte Aufgaben

- Mit welcher SQL-Abfrage kann der durchschnittliche Umsatz in Berlin ermittelt werden?
- Wie lautet die SQL-Abfrage, um den Gesamtumsatz (Summe) des 16.06.12 zu bestimmen?

Verkauf	Vnr	Produkt	Filiale	Datum	Umsatz
	123	TV	Köln	12.06.2012	2500
	234	SAT	Köln	15.06.2012	1200
	345	DVD	Berlin	15.06.2012	1000
	456	SAT	Berlin	16.06.2012	1800
	567	DVD	München	16.06.2012	2100
	678	DVD	Berlin	16.06.2012	900
	789	SAT	Berlin	17.06.2012	1800
	890	DVD	München	17.06.2012	2000

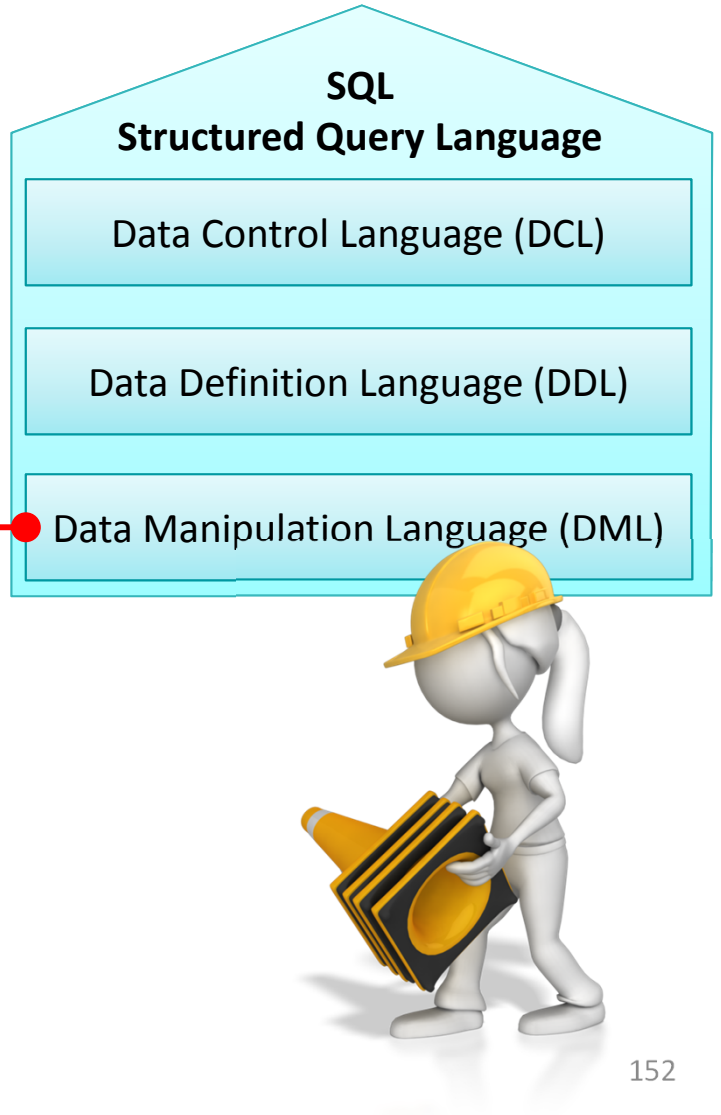
Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML): dient zur Abfrage, zum Hinzufügen, zur Veränderung und zum Löschen von Daten

- SELECT
- UPDATE
- INSERT
- DELETE

Data Definition Language (DDL)

Data Control Language (DCL)



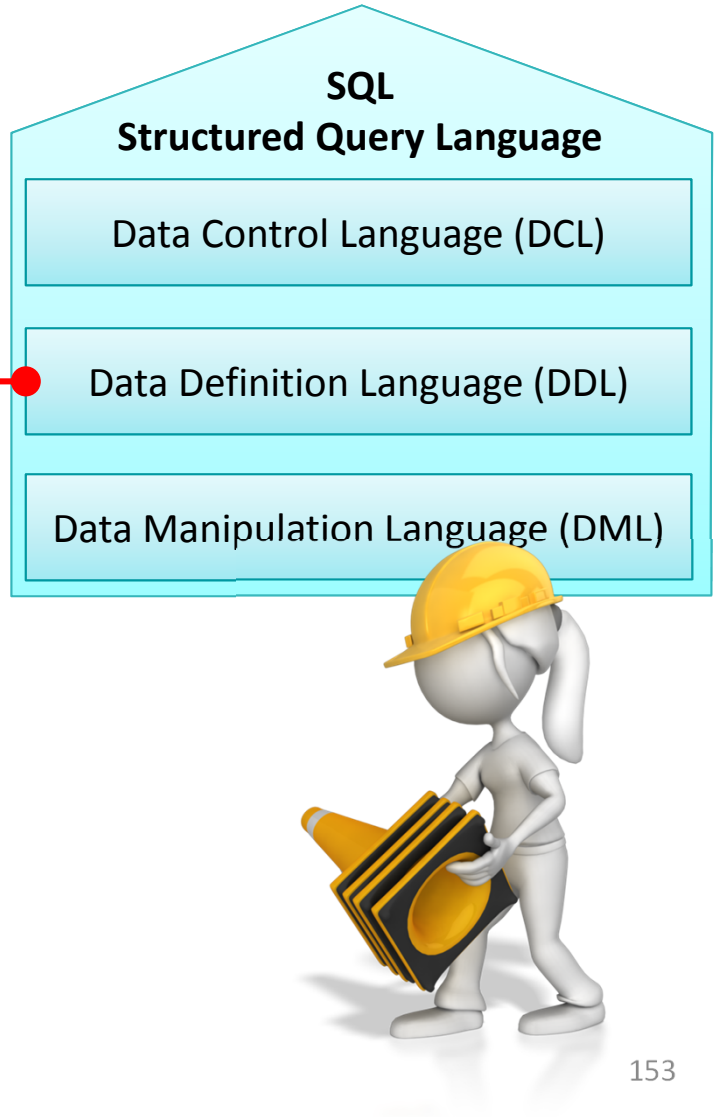
Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML)

Data Definition Language (DDL): dient zum Erzeugen, Verändern und Löschen der Strukturen, die für die Speicherung der Daten benutzt werden (z.B. Tabellen, Spalten)

- CREATE
- ALTER
- DROP

Data Control Language (DCL)



SQL Data Definition Language

Erzeugen einer Tabelle in SQL (für MS Access)

```
CREATE TABLE Bestellung
(
  IdPk INTEGER CONSTRAINT PirmSchluessel PRIMARY KEY,
  BestellDatum DATE,
  Anzahl SMALLINT,
  prdIdFk INTEGER CONSTRAINT FremdSchluesselPrd
            REFERENCES Produkte(IdPk)
);
```

Als Relation

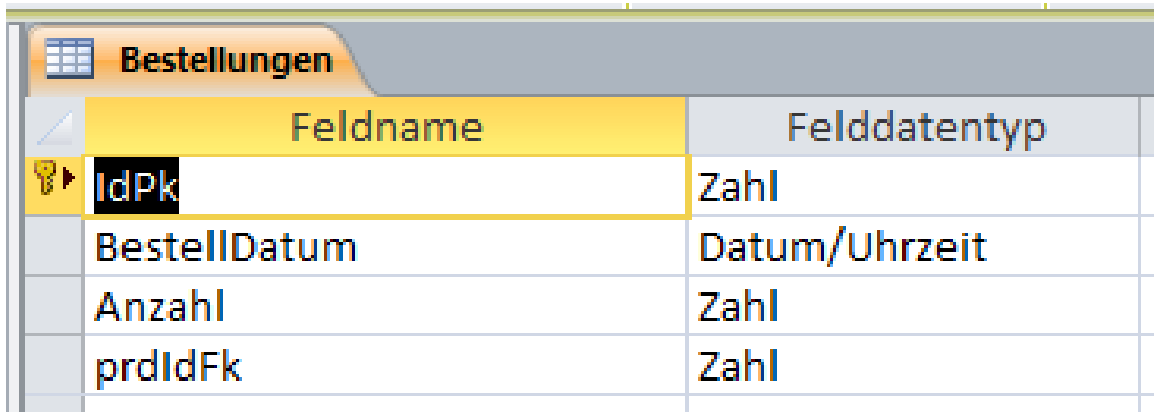
Bestellung	IdPk	BestellDatum	Anzahl	prdIdFk
	…	…	…	…

SQL Data Definition Language

SQL (für MS Access)

```
CREATE TABLE Bestellung
(
  IdPk INTEGER CONSTRAINT PirmSchluessel PRIMARY KEY,
  BestellDatum DATE,
  Anzahl SMALLINT,
  prdIdFk INTEGER CONSTRAINT FremdSchluesselPrd
            REFERENCES Produkte(IdPk)
);
```

In MS Access Entwurfsansicht

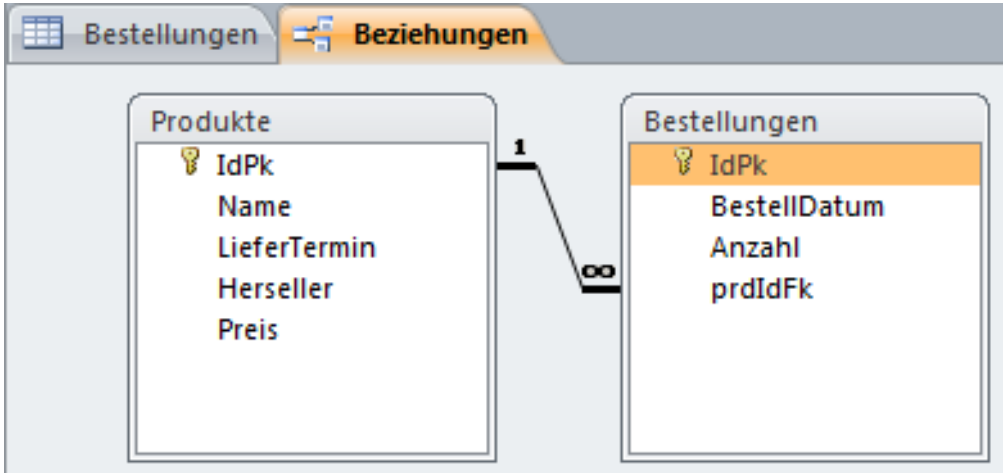


SQL Data Definition Language

SQL (für MS Access)

```
CREATE TABLE Bestellung
(
  IdPk INTEGER CONSTRAINT PirmSchluessel PRIMARY KEY,
  BestellDatum DATE,
  Anzahl SMALLINT,
  prdIdFk INTEGER CONSTRAINT FremdSchluesselPrd
            REFERENCES Produkte(IdPk)
);
```

In MS Access Beziehungsansicht



SQL Data Definition Language

Vorher (Entwurfsansicht)

Verkaeufer

Feldname	Felddatentyp
VerkIdPk	Zahl
AbtIdPk	Zahl
Name	Text
Gehalt	Währung
AbtLeister	Ja/Nein

SQL (für MS Access)

```
ALTER TABLE Verkaeufer
 ADD COLUMN Vorname VARCHAR(25);
```

Nachher (Entwurfsansicht)

Verkaeufer

Feldname	Felddatentyp
VerkIdPk	Zahl
AbtIdPk	Zahl
Name	Text
Gehalt	Währung
AbtLeister	Ja/Nein
Vorname	Text

SQL Data Definition Language

Vorher (Entwurfsansicht)

Verkaeufer

Feldname	Felddatentyp
VerkIdPk	Zahl
AbtIdPk	Zahl
Name	Text
Gehalt	Währung
AbtLeister	Ja/Nein
Vorname	Text

SQL (für MS Access)

```
ALTER TABLE Verkaeufer
 DROP CONSTRAINT PrimSchluesselVerk;
```

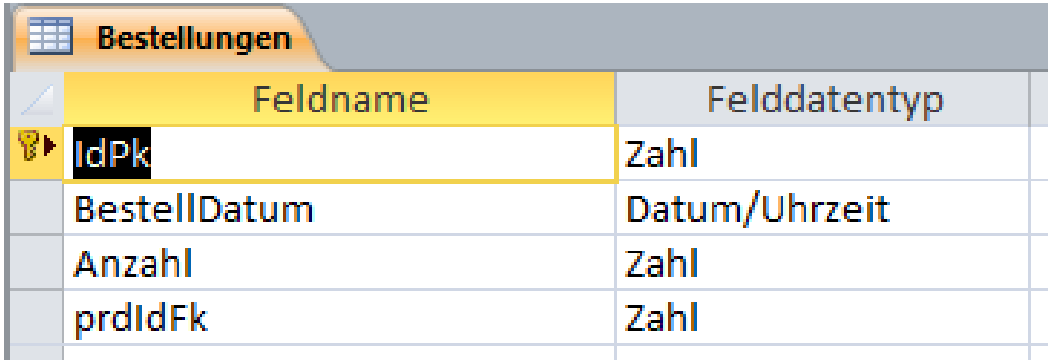
Nachher (Entwurfsansicht)

Verkaeufer

Feldname	Felddatentyp
VerkIdPk	Zahl
AbtIdPk	Zahl
Name	Text
Gehalt	Währung
AbtLeiter	Ja/Nein
Vorname	Text

SQL Data Definition Language

Vorher



SQL (für MS Access)

```
DROP TABLE Bestellungen;
```

Nachher

Prüfungsvorbereitung

Beispielhafte Aufgaben

– Sie sehen folgende Ausgangs- und Ergebnis-relationen. Schreiben Sie die zugehörigen SQL-Befehle.

a) Ausgangsrelation:

Kunden	KndNr	Name	Ort	Alter
	123	Albers	Köln	18
	234	Berger	Köln	17
	345	Yilmaz	Berlin	17

Ergebnisrelation:

Kunden	KndNr	Name	Ort	GebDat
	123	Albers	Köln	
	345	Yilmaz	Berlin	
	456	Schmidt	Berlin	

b) Ausgangsrelation:

Ergebnisrelation:

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Berger	Vera
	567	Schuster	Dirk

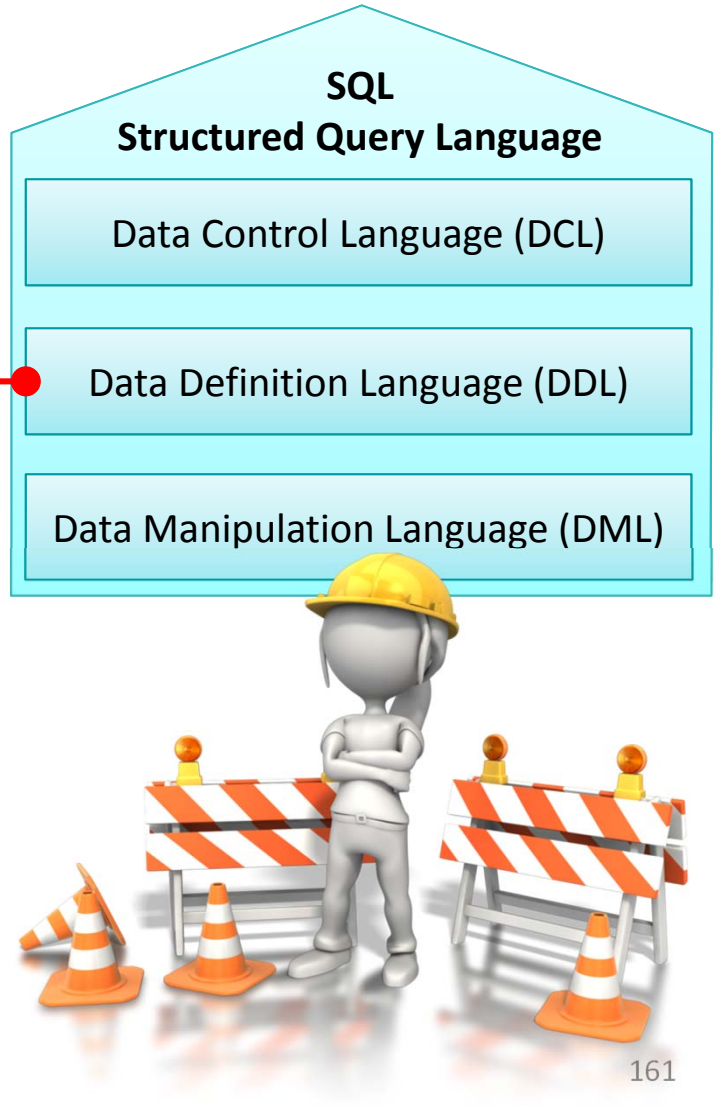
Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML)

Data Definition Language (DDL): dient zum Erzeugen, Verändern und Löschen der Strukturen, die für die Speicherung der Daten benutzt werden (z.B. Tabellen, Spalten)

- CREATE
- ALTER
- DROP

Data Control Language (DCL)



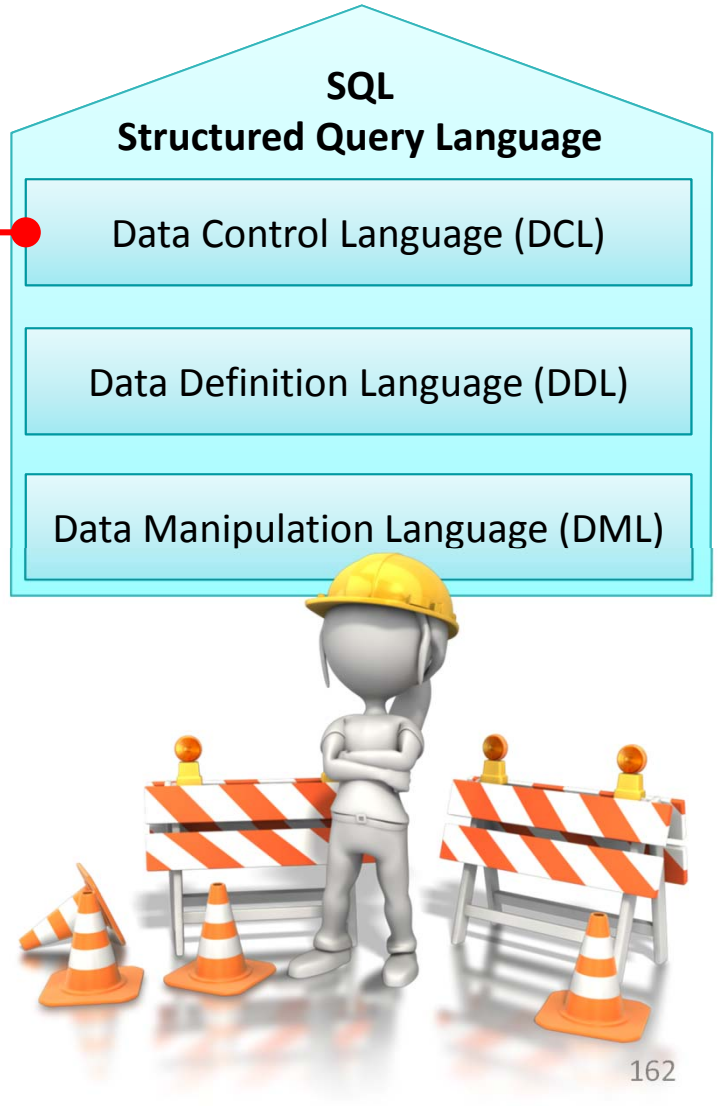
Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML)

Data Definition Language (DDL)

Data Control Language (DCL): dient zum Einrichten, Festlegen und Entziehen von Zugriffsrechten für Benutzer und Gruppen auf den Strukturen und Aktionsmöglichkeiten der Datenbank

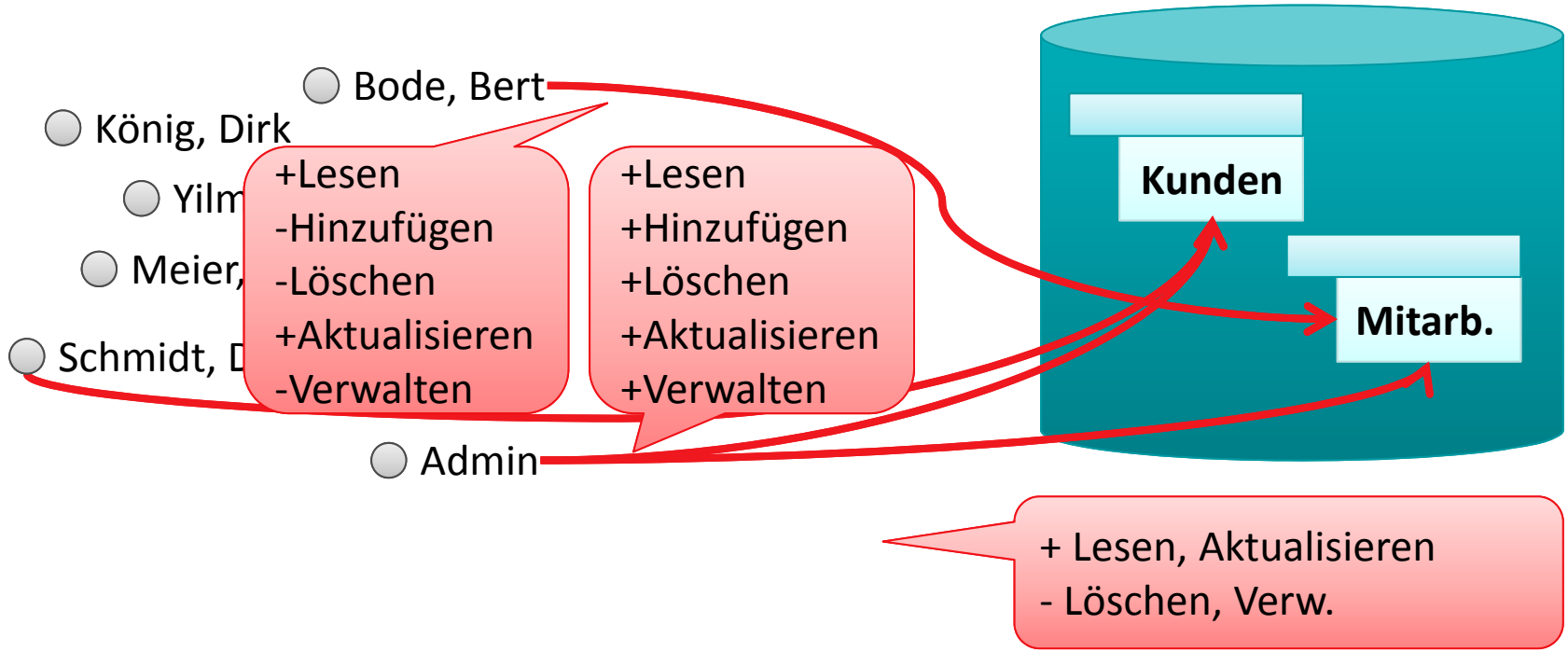
- GRANT
- REVOKE



Benutzer und Gruppen

Benutzer

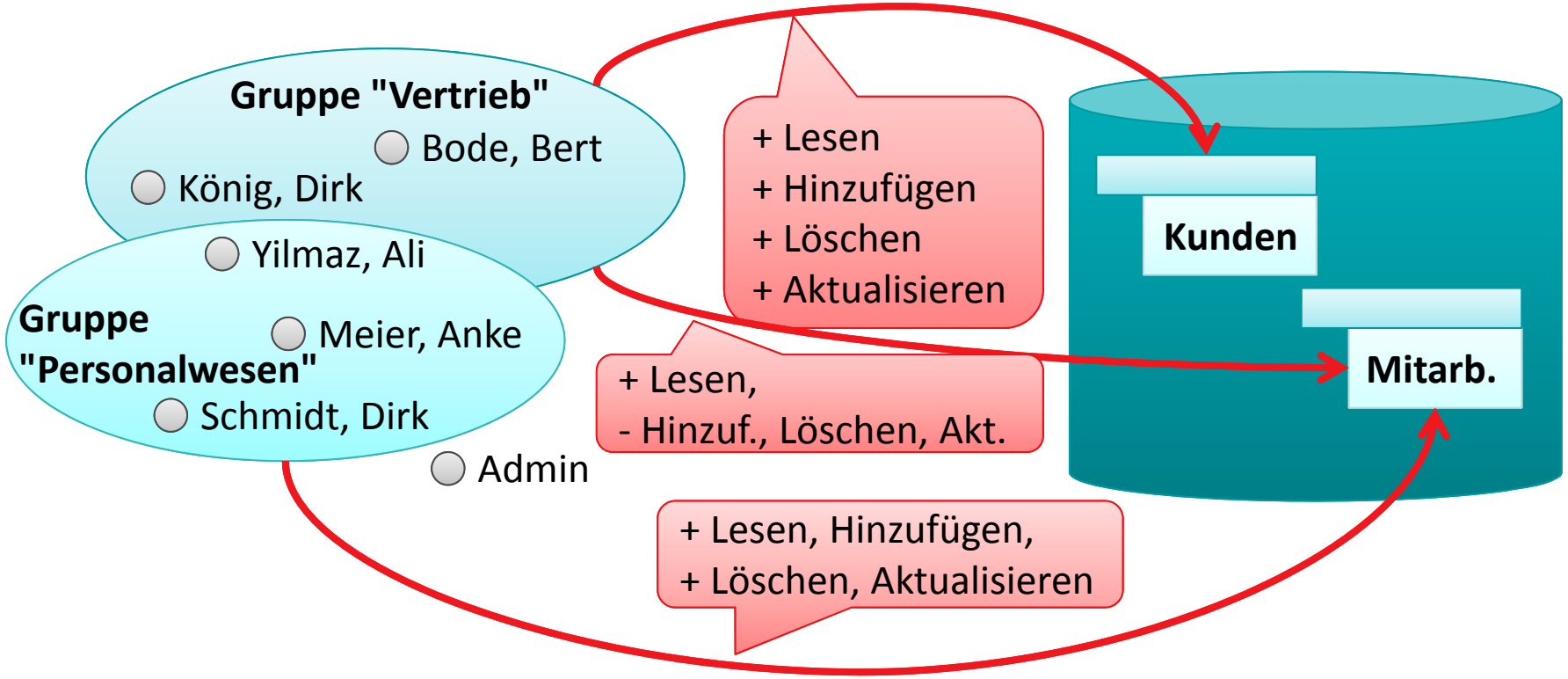
- Identifizierbar und authentifizierbar
- für Zugang zur Datenbank
- bestimmte Aktionsmöglichkeiten eingerichtet oder entzogen
- kann einer oder mehreren Gruppen zugeordnet werden



Benutzer und Gruppen

Gruppen

- umfasst einen oder mehrere Benutzer
- für Zugang zur Datenbank
- bestimmte Aktionsmöglichkeiten eingerichtet oder entzogen



Benutzer und Gruppen in SQL

Anlegen von Benutzern/Gruppen

```
CREATE USER <Benutername> <Passwort>
CREATE GROUP <Gruppenname>
```

Zuordnen von Benutzern zu Gruppen

```
ADD USER <Benutername> TO <Gruppenname>
ADD USER <Benutername1>, <Benutername2>, ... TO <Gruppe>
```

Entfernen von Benutzer aus Gruppen

```
DROP USER <Benutername> FROM <Gruppenname>
```

Entfernen von Benutzern/Gruppen

```
DROP USER <Benutername>
DROP GROUP <Gruppenname>
```

Beispiel

Anlegen von Dirk König und Andrea Meier als Benutzer

```
CREATE USER dkoenig geheim123
CREATE USER ameier geheim234
```

Anlegen der Gruppen Vertrieb und Personalwesen

```
CREATE GROUP vertrieb
CREATE GROUP pesonalwesen
```

Zuordnen von Dirk König und Andrea Meier zu Gruppen

```
ADD dkoenig TO vertrieb
ADD ameier TO pesonalwesen
```

Aktionsmöglichkeiten

Für Gruppen und/oder Benutzer können Aktionsmöglichkeiten eingerichtet oder entzogen werden

– beispielsweise (MS Access)

Aktion	Beschreibung
SELECT	Lesender Zugriff auf die Datensätze einer Tabelle
INSERT	Hinzufügen neuer Datensätze zu einer Tabelle
DELETE	Löschen vorhandener Datensätze aus einer Tabelle
UPDATE	Ändern vorhandener Datensätze in einer Tabelle
DROP	Löschen von Tabellen (und deren Daten)
…	…

Aktionsmöglichkeiten

Festlegen von Aktionsmöglichkeiten auf Tabellen

```
GRANT <Aktionsmöglichkeit>
 ON TABLE <Tabellenname>
 TO <BenutzerOderGruppe>
```

Entziehen von Aktionsmöglichkeiten auf Tabellen

```
REVOKE <Aktionsmöglichkeit>
 ON TABLE <Tabellenname>
 FROM <BenutzerOderGruppe>
```

Beispiel

Festlegen von Aktionsmöglichkeiten auf Tabellen

– Vertrieb darf Kunden lesen und bearbeiten

```
GRANT
 SELECT, INSERT, UPDATE, DELETE
 ON TABLE tblKunden
 TO Vertrieb
```

– D. König darf (zusätzlich) Mitarbeiter lesen

```
GRANT SELECT
 ON TABLE tblMitarbeiter
 TO dkoenig
```

– Personalwesen darf Mitarbeiter lesen, hinzufügen, bearbeiten, aber nicht löschen

```
GRANT SELECT, INSERT, UPDATE
 ON TABLE tblMitarbeiter
 TO Personalwesen
```

Zusammenfassung

Schutz durch Nutzung von SQL Data Control Language

- Einrichten von Gruppen und Benutzern

  `CREATE USER/GROUP <BenutzerGruppe>`

- Hinzufügen von Benutzern zu Gruppen

  `ADD USER <Benutzer> TO <Gruppe>`

- Einrichten von Zugriffsrechten

```
GRANT <Aktionmgl>
 ON TABLE <Name>
 TO <BenutzerOderGruppe>
```

- Entfernen der Gruppenmitgliedschaft und eines Benutzerns: **`DROP`**
- Entziehen von Zugriffsrechten mit **`REVOKE`** (ähnlich **`GRANT`**)

Prüfungsvorbereitung

Beispielhafte Aufgaben

- Legen Sie mit SQL zwei beliebige Gruppen und zwei Benutzer an. Ordnen Sie einen Benutzer der ersten Gruppe und den zweiten Benutzer beiden Gruppen zu. Wie lautet der SQL-Befehl?
- Gegeben sind die Gruppen Vertrieb, Personal und Produktion sowie die Tabellen Kunden, Mitarbeiter und Produkte.
  - Gewähren Sie der Gruppe Vertrieb lesenden Zugriff auf die Tabelle Produkte. Wie lautet der SQL-Befehl?
  - Entziehen Sie der Gruppe Vertrieb den schreibenden Zugriff (Einfügen, Ändern und Löschen) auf der Mitarbeiter. Wie lautet der SQL-Befehl?

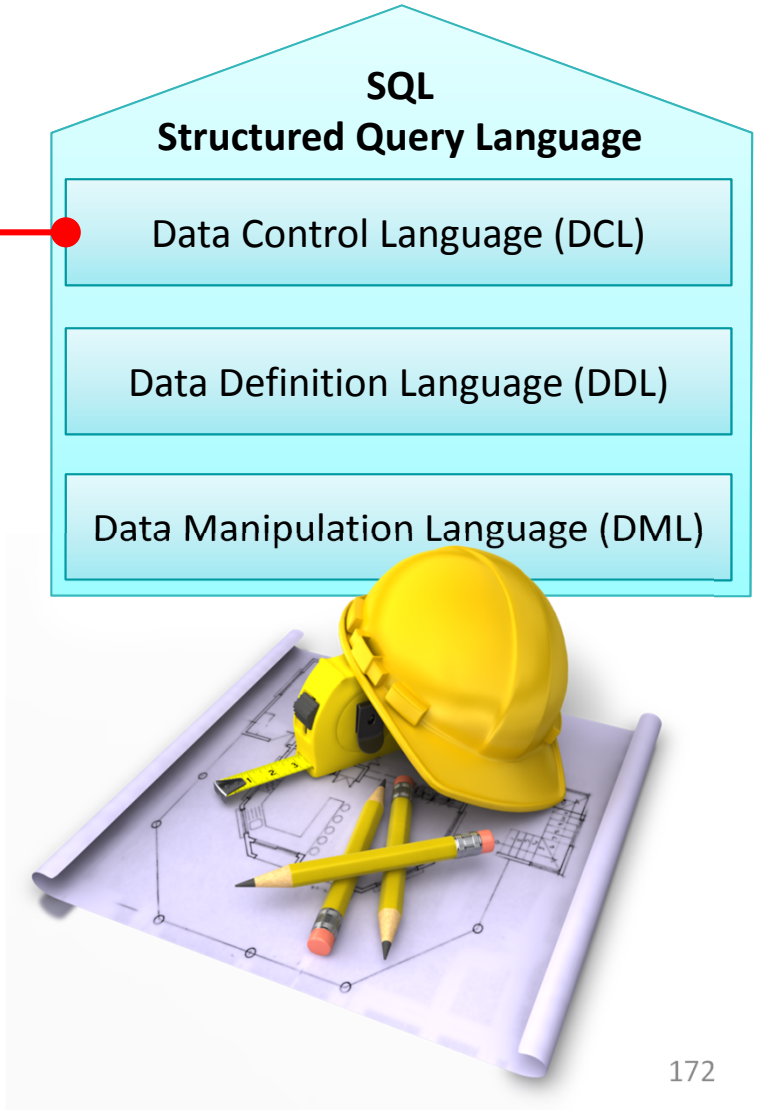
Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML)

Data Definition Language (DDL)

Data Control Language (DCL): dient zum Einrichten, Festlegen und Entziehen von Zugriffsrechten für Benutzer und Gruppen auf den Strukturen und Aktionsmöglichkeiten der Datenbank

- GRANT
- REVOKE

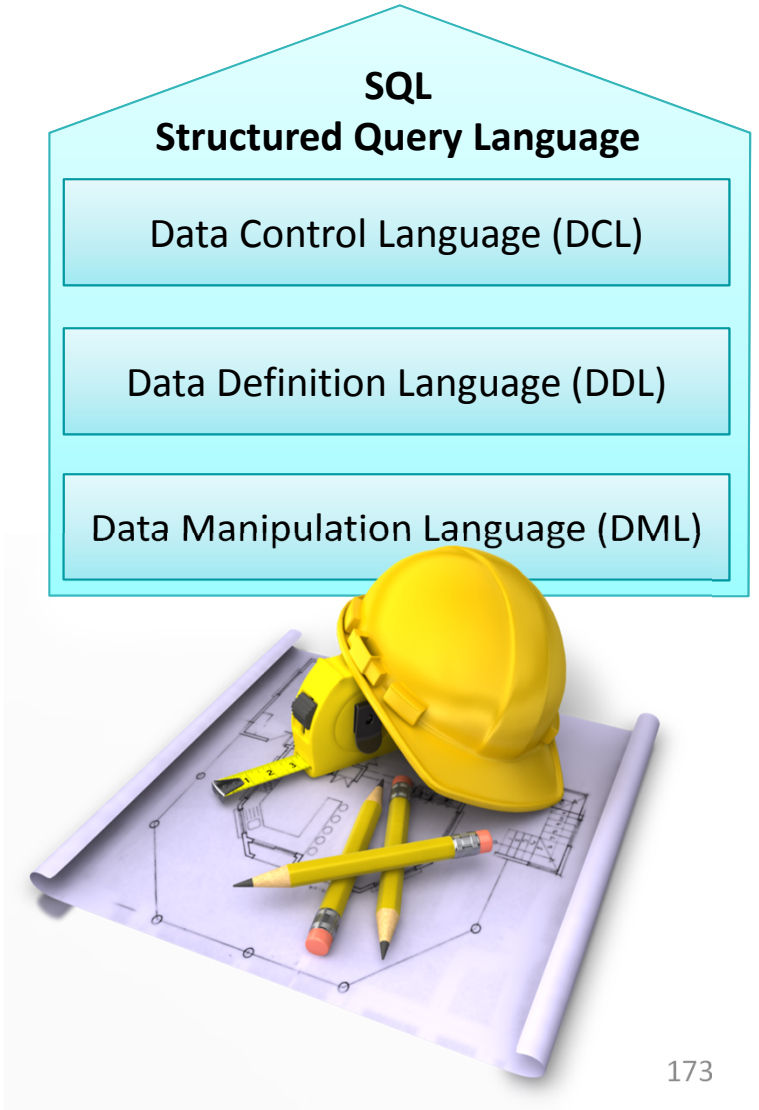


Bestandteile des SQL-Sprachumfangs

Data Manipulation Language (DML)

Data Definition Language (DDL)

Data Control Language (DCL)



Prüfungsvorbereitung

Beispielhafte Aufgaben

- In welche Teilsprachen lässt sich SQL unterteilen? Wozu dienen diese Teilsprachen?
- Nennen Sie pro SQL-Teilsprache zwei Befehle und erläutern Sie, wozu diese verwendet werden!

Inhalt

Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Programmierschnittstelle zur Datenbank

Programmiersprache

– dient der Programmentwicklung

Datenbanksprache

– dient u.a. der Abfrage, Manipulation von Daten

Programmiersprache

```
Dim strName _
     As String
Let strName =".."

'...
```

Datenbanksprache

```
SELECT * FROM
Kunden
WHERE Stadt='Köln'
```

Programmierschnittstelle zur Datenbank

Um innerhalb einer Programmiersprache auf die Datenbank zugreifen zu können, ...

Programmiersprache

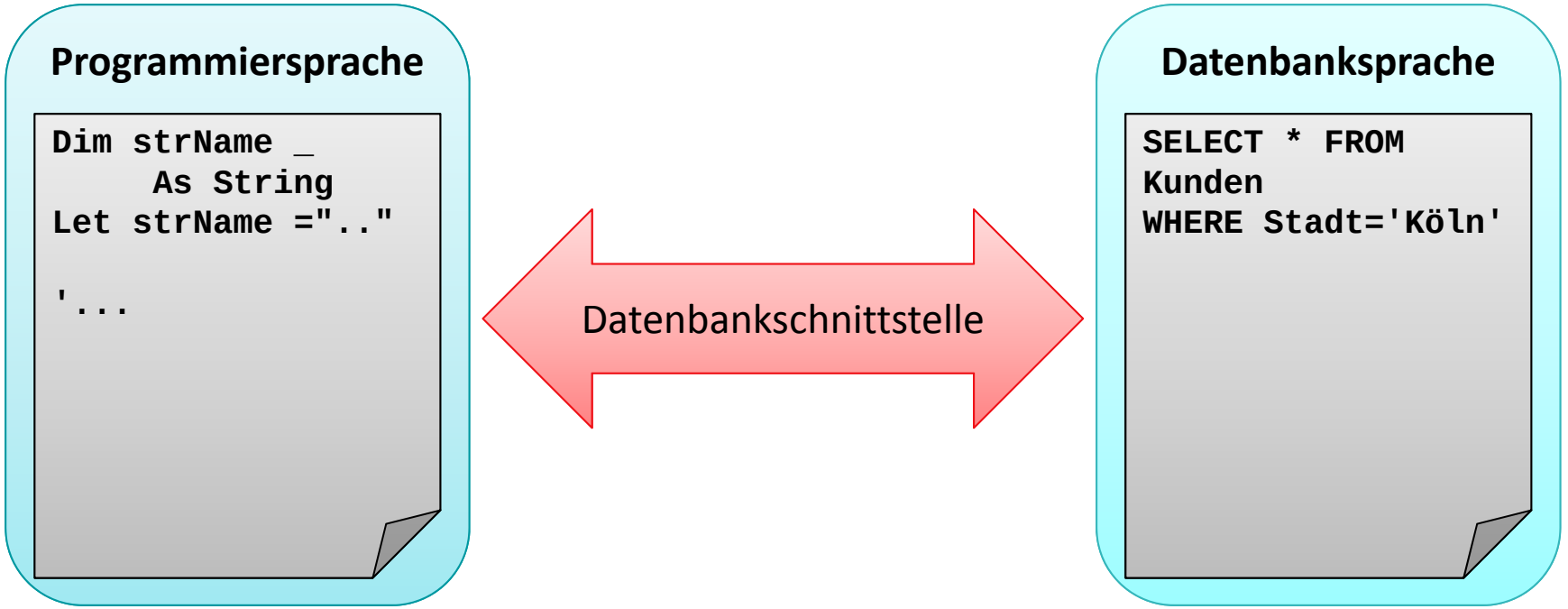
```
Dim strName _
     As String
Let strName =".."

'...
```

Datenbanksprache

```
SELECT * FROM
Kunden
WHERE Stadt='Köln'
```

Programmierschnittstelle zur Datenbank

Um innerhalb einer Programmiersprache auf die Datenbank zugreifen zu können, muss eine Verbindung zwischen Sprachelementen der Programmiersprache und Elementen der Datenbanksprache existieren:



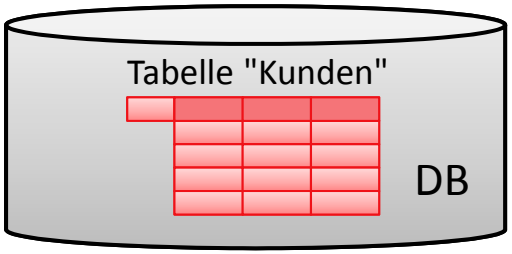
Funktionen einer Datenbankschnittstelle

Funktionen einer Datenbankschnittstelle am Beispiel des Recordsets

- Abfragen von Datensätzen
  - Auswählen nach bestimmten Kriterien
  - Navigation über die gefundenen Datensätze
- Einfügen von neuen Datensätzen
- Ändern vorhandener Datensätze
- Löschen vorhandener Datensätze

Abfragen von Datensätzen per Recordset

Ziel: Aus Kunden den Vornamen und Nachnamen des ersten Kunden lesen

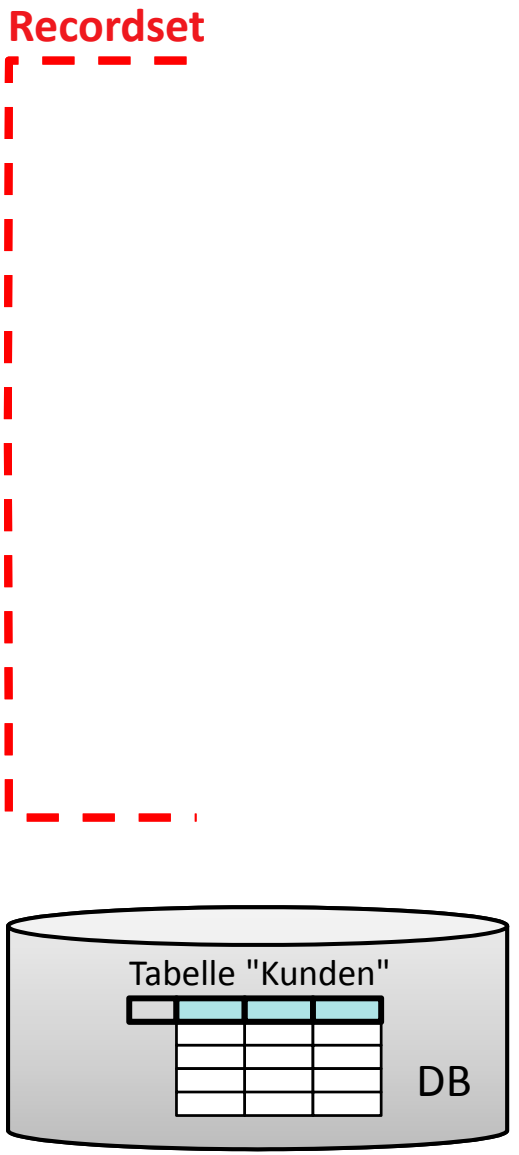


Abfragen von Datensätzen per Recordset

Ziel: Aus Kunden den Vornamen und Nachnamen des ersten Kunden lesen

Ansatz

– Recordset deklarieren

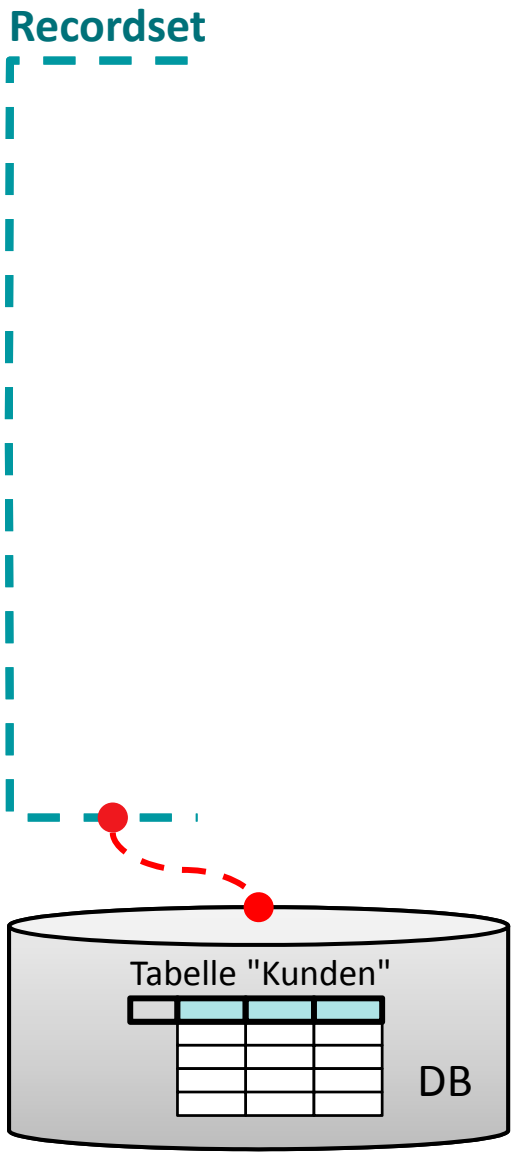


Abfragen von Datensätzen per Recordset

Ziel: Aus Kunden den Vornamen und Nachnamen des ersten Kunden lesen

Ansatz

- Recordset deklarieren
- Verbindung zur Datenbank vorbereiten

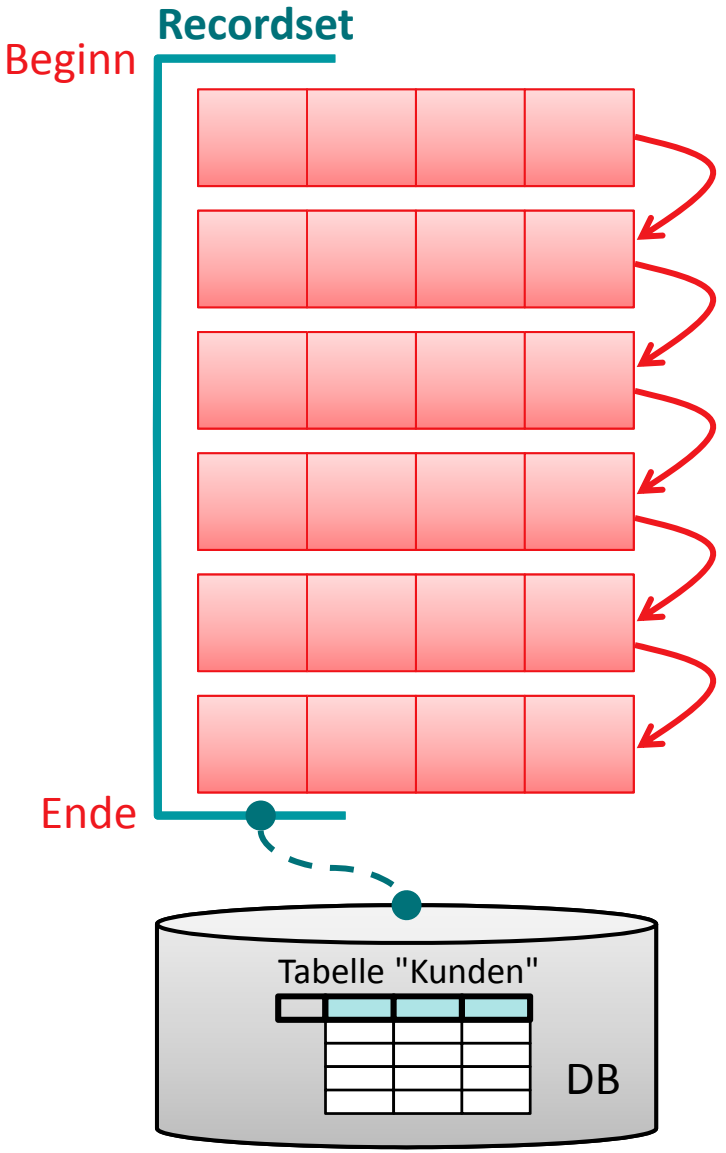


Abfragen von Datensätzen per Recordset

Ziel: Aus Kunden den Vornamen und Nachnamen des ersten Kunden lesen

Ansatz

- Recordset deklarieren
- Verbindung zur Datenbank vorbereiten
- Recordset mit einer Abfrage (z.B. SQL) oder Tabelle initialisieren und dadurch füllen

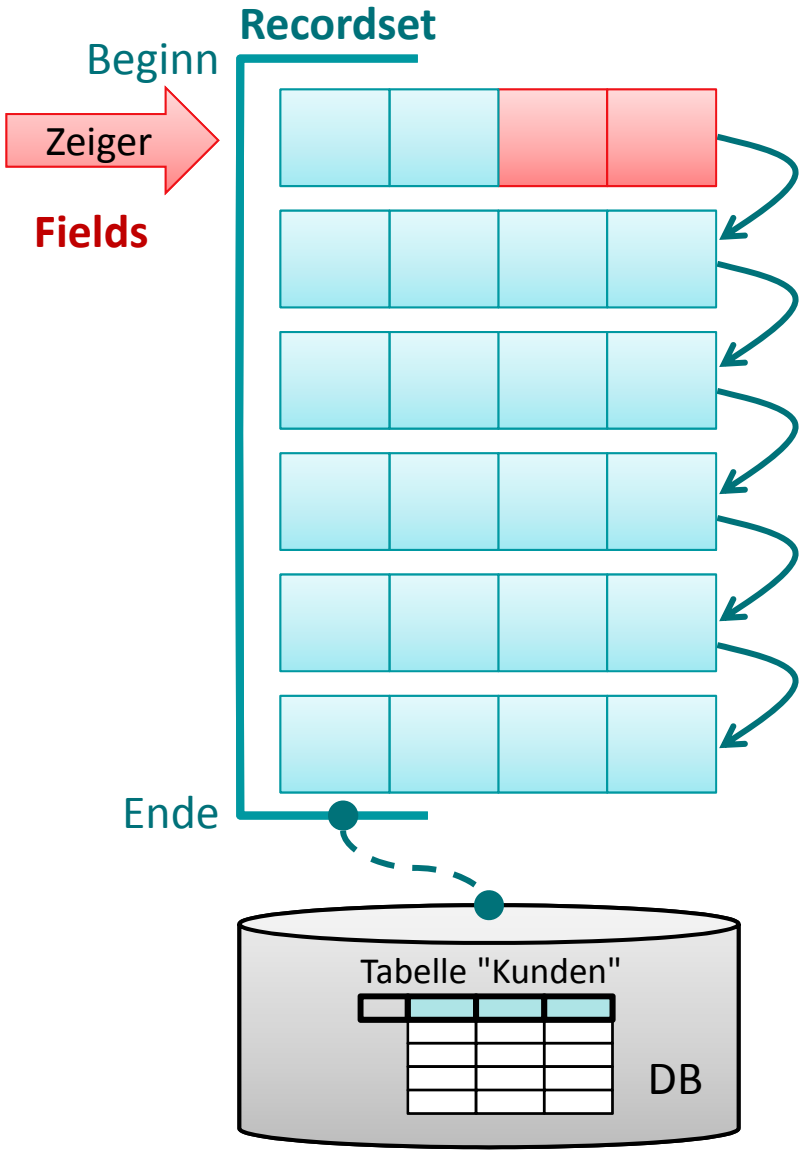


Abfragen von Datensätzen per Recordset

Ziel: Aus Kunden den Vornamen und Nachnamen des ersten Kunden lesen

Ansatz

- Recordset deklarieren
- Verbindung zur Datenbank vorbereiten
- Recordset mit einer Abfrage (z.B. SQL) oder Tabelle initialisieren und dadurch füllen
- per Zeiger (steht auf erstem Element) auf Felder zugreifen

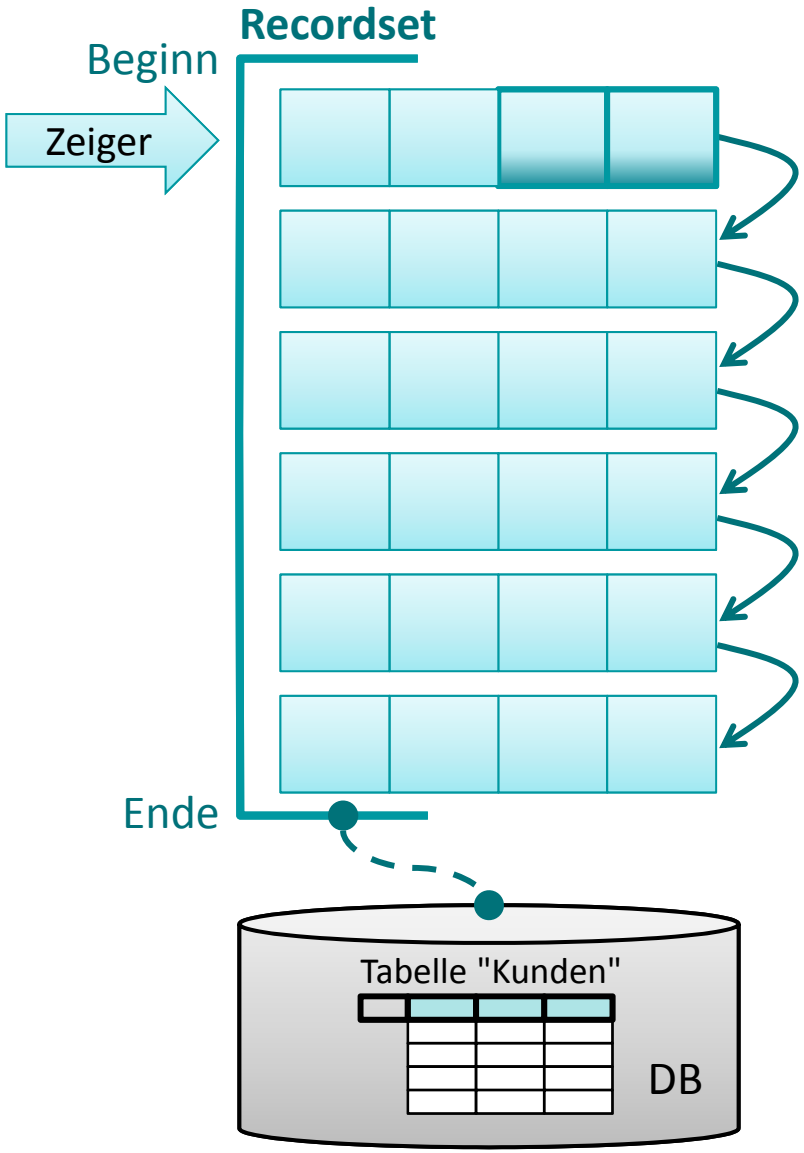


Abfragen von Datensätzen per Recordset

Ziel: Aus Kunden den Vornamen und Nachnamen des ersten Kunden lesen

Ansatz

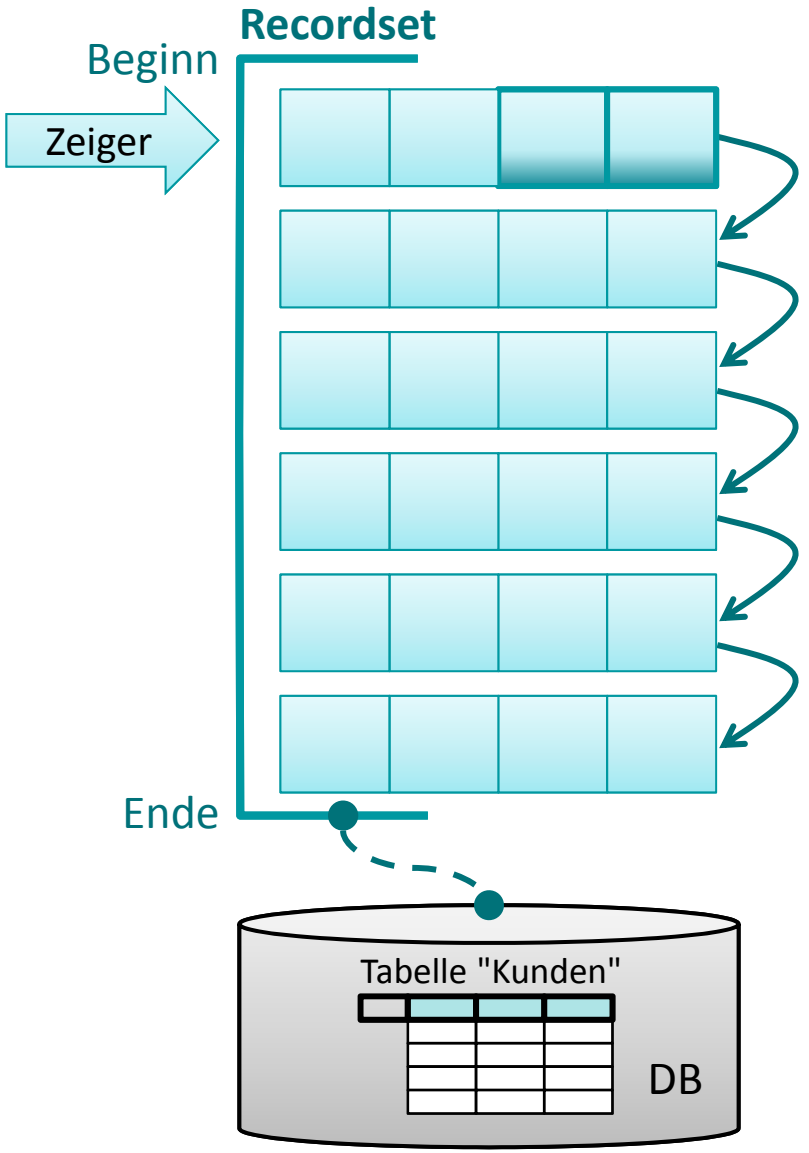
- Recordset deklarieren
- Verbindung zur Datenbank vorbereiten
- Recordset mit einer Abfrage (z.B. SQL) oder Tabelle initialisieren und dadurch füllen
- per Zeiger (steht auf erstem Element) auf Felder zugreifen



Abfragen von Datensätzen per Recordset

Modul

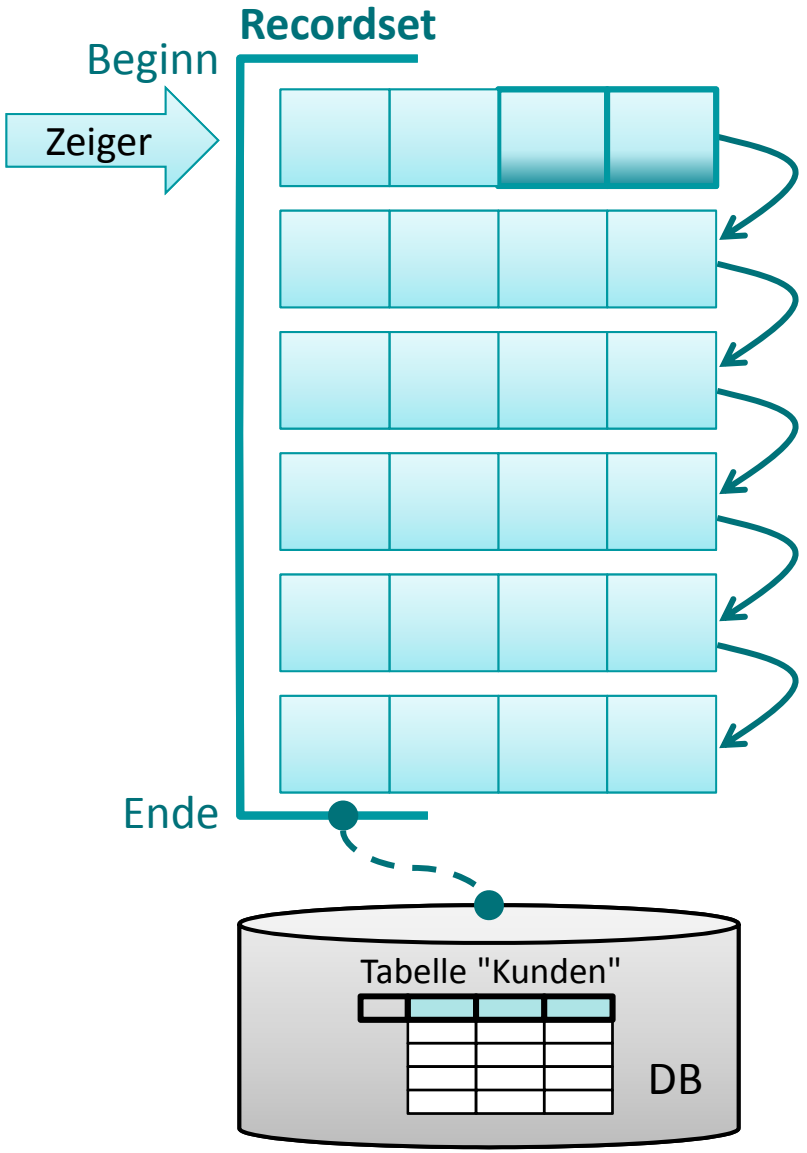
```
Option Compare Database
Option Explicit

Sub ausgebenKundeNameVorname()

' Recordset deklarieren
Dim rcsKnd As Recordset

' Verbindung zur DB vorbereiten
Dim db As Database
Set db = CurrentDb

' Recordset inititalisieren
Set rcsKnd=db.OpenRecordset("tblKunden")

' auf Felder zugreifen/ausgeben
Debug.Print rcsKnd.Fields("kndName")
Debug.Print rcsKnd.Fields("kndVorname")

End Sub
```



Abfragen von Datensätzen per Recordset

Modul

```
Option Compare Database
Option Explicit

Sub ausgebenKundeNameVorname()

' Recordset deklarieren
Dim rcsKnd As Recordset

' Alternative: Kurze Version
Set rcsKnd= _
  CurrentDb.OpenRecordset("tblKunden")

' auf Felder zugreifen/ausgeben
Debug.Print rcsKnd.Fields("kndName")
Debug.Print rcsKnd.Fields("kndVorname")

End Sub
```



Funktionen einer Datenbankschnittstelle

Funktionen einer Datenbankschnittstelle am Beispiel des Recordsets

- Abfragen von Datensätzen
  - Auswählen nach bestimmten Kriterien
  - Navigation über die gefundenen Datensätze
- Einfügen von neuen Datensätze
- Ändern vorhandener Datensätze
- Löschen vorhandener Datensätze

Navigation im Recordset per Zeiger

Navigationsmöglichkeiten

- zum nächsten Datensatz verschieben

  `<einRecordset>.MoveNext`

- zum vorherigen Datensatz verschieben

  `<einRcrdst>.MovePrevious`

- zum letzten Datensatz verschieben

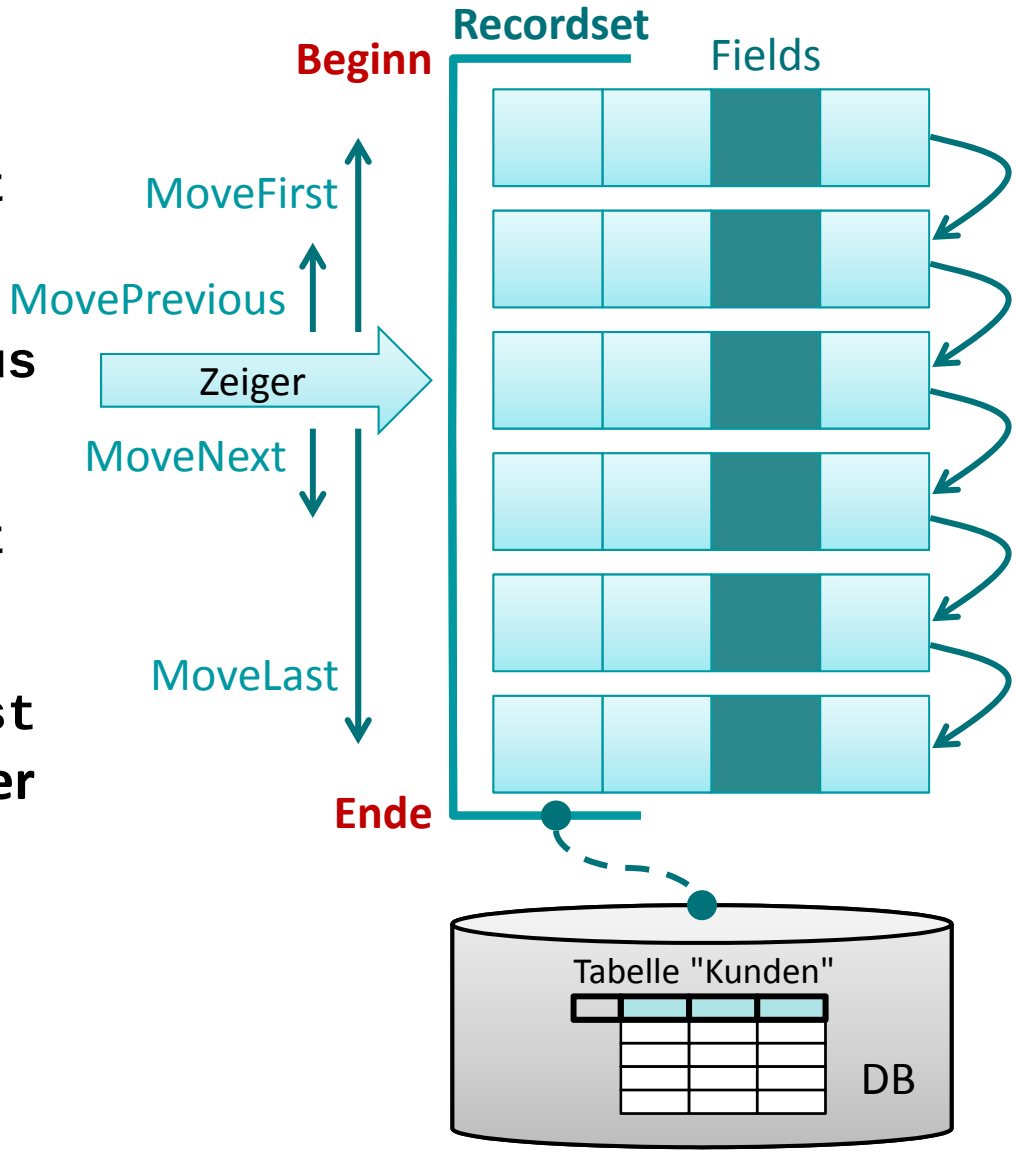
  `<einRecordset>.MoveLast`

- zum ersten Datensatz verschieben

  `<einRecordset>.MoveFirst`

Positionsbestimmung, ob Zeiger am Beginn oder am Ende steht (End of File)

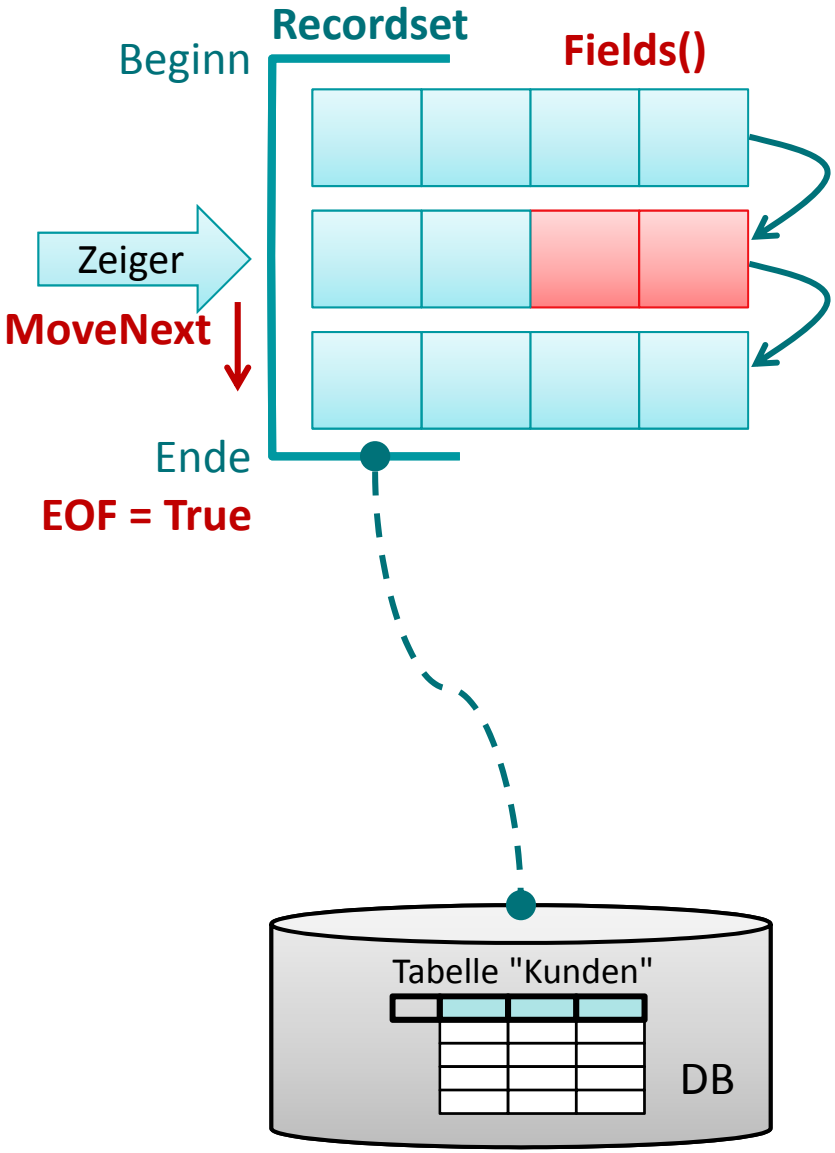
`<einRecordset>.BOF`
`<einRecordset>.EOF`



Navigation im Recordset per Zeiger

Modul

```
Option Compare Database
Option Explicit

Sub ausgebenAlleKunden()
' Recordset deklarieren
Dim rcsKnd As Recordset
' Verbindung zur DB vorbereiten
Dim db As Database
Set db = CurrentDb
' Recordset inititalisieren
Set rcsKnd=db.OpenRecordset("tblKunden")

'Schleife bis zum Ende des Recordset
Do Until rcsKnd.EOF
' Name, Vorname des aktuellen
' Elements ausgeben (Zeiger zeigt drauf)
Debug.Print rcsKnd.Fields("kndName") & _
 ", " & rcsKnd.Fields("kndVorname")
' Zeiger weiterrücken
rcsKnd.MoveNext
' Solange fortfahren, bis Ende erreicht
Loop

End Sub
```



Suchen und Finden

Navigations-möglichkeiten

- ersten Datensatz finden, der ein bestimmtes Kriterium erfüllt

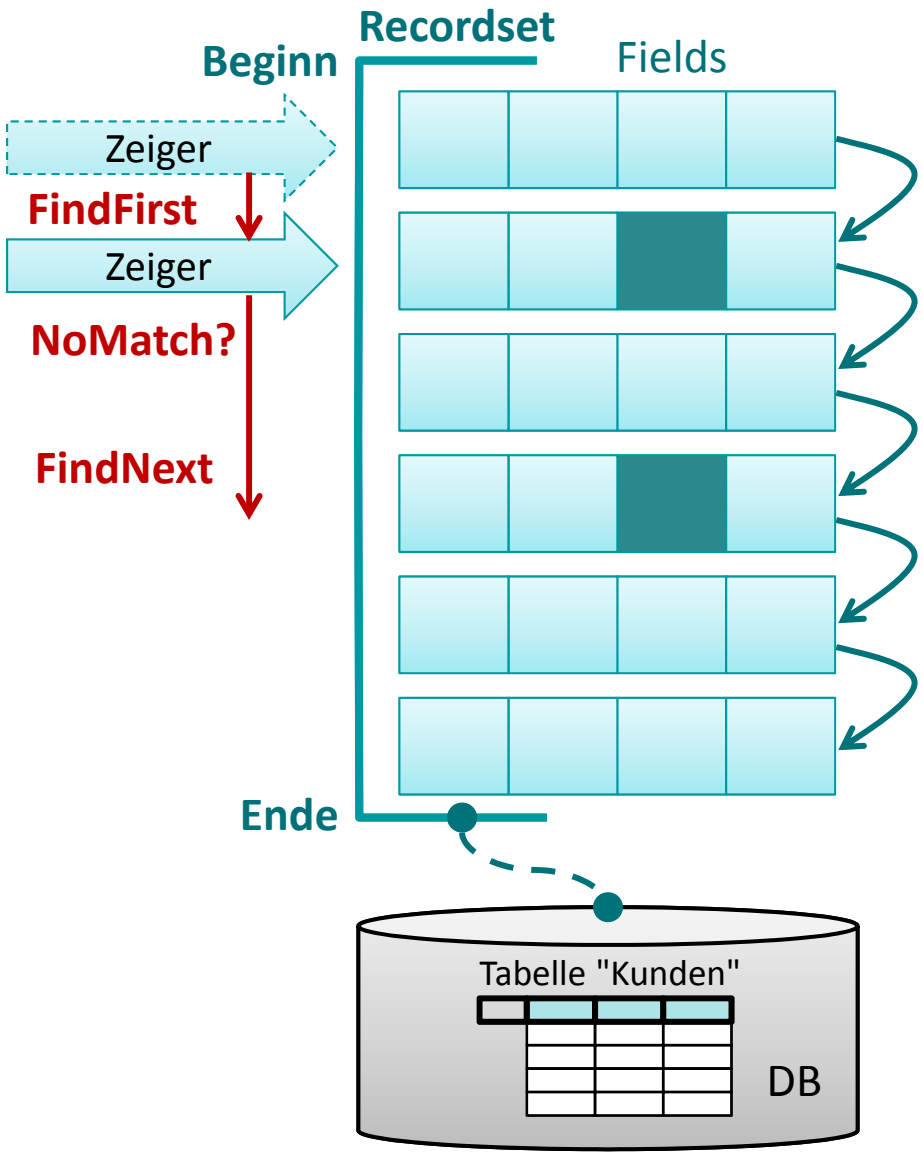
  `<einRcrdst>.FindFirst`

- nächsten Datensatz finden, der ein bestimmtes Kriterium erfüllt

  `<einRcrdst>.FindNext`

Positionsbestimmung, ob Datensätze gefunden werden konnten

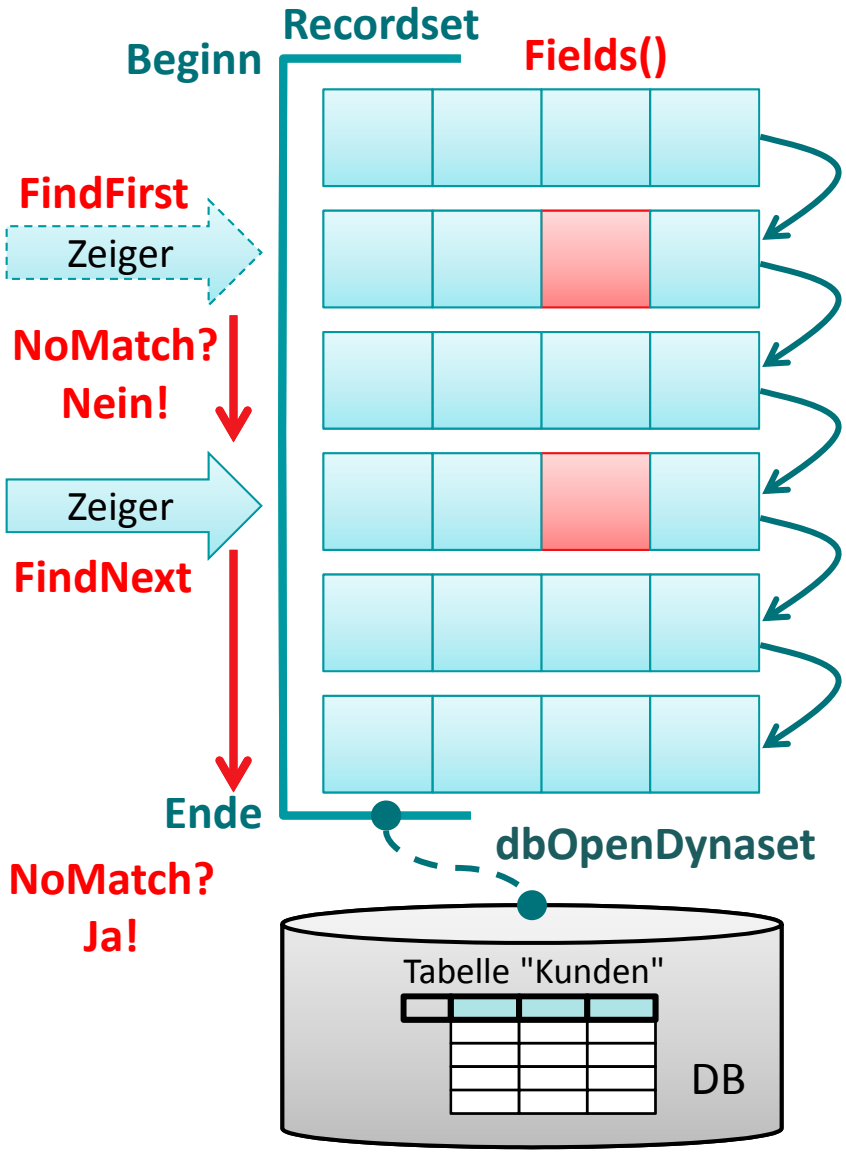
`<einRecordset>.NoMatch`



Suchen und Finden im Recordset

Modul

```
Option Compare Database
Option Explicit

Sub sucheKundenMeier()
' Verbindung zur DB vorbereiten
Dim db As Database
Set db = CurrentDb
' Recordset deklarieren, initialisieren
Dim rcsKnd As Recordset
Set rcsKnd=db.OpenRecordset("tblKunden",
                            dbOpenDynaset)
' Suchen nach Kunden "Meier"
rcsKnd.FindFirst ("kndName = 'Meier'")

' Schleife bis zum Ende des Recordset
Do Until rcsKnd.NoMatch
' Name und Vorname ausgeben
Debug.Print rcsKnd.Fields("kndName") & _
  ", " & rcsKnd.Fields("kndVorname")
' Nächsten finden
rcsKnd.FindNext ("kndName = 'Meier'")
' Weiterer Schleifendurchlauf
Loop
End Sub
```



Funktionen einer Datenbankschnittstelle

Funktionen einer Datenbankschnittstelle am Beispiel des Recordsets

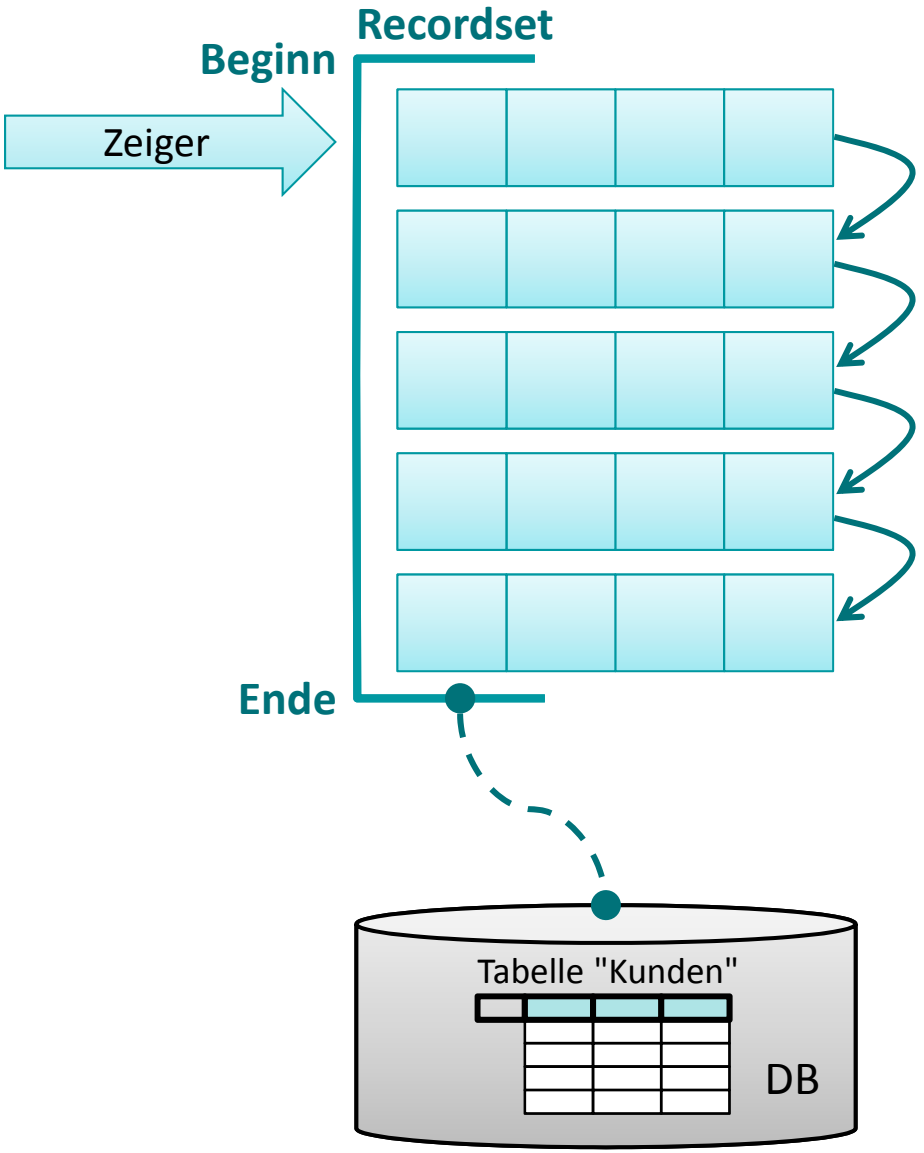
- Abfragen von Datensätzen
  - Auswählen nach bestimmten Kriterien
  - Navigation über die gefundenen Datensätze
- Einfügen von neuen Datensätzen
- Ändern vorhandener Datensätze
- Löschen vorhandener Datensätze

Einfügen von neuen Datensätzen

Ziel: Ein Datensatz soll Hinzugefügt werden

Ansatz

– Recordset als Dnyaset befüllen

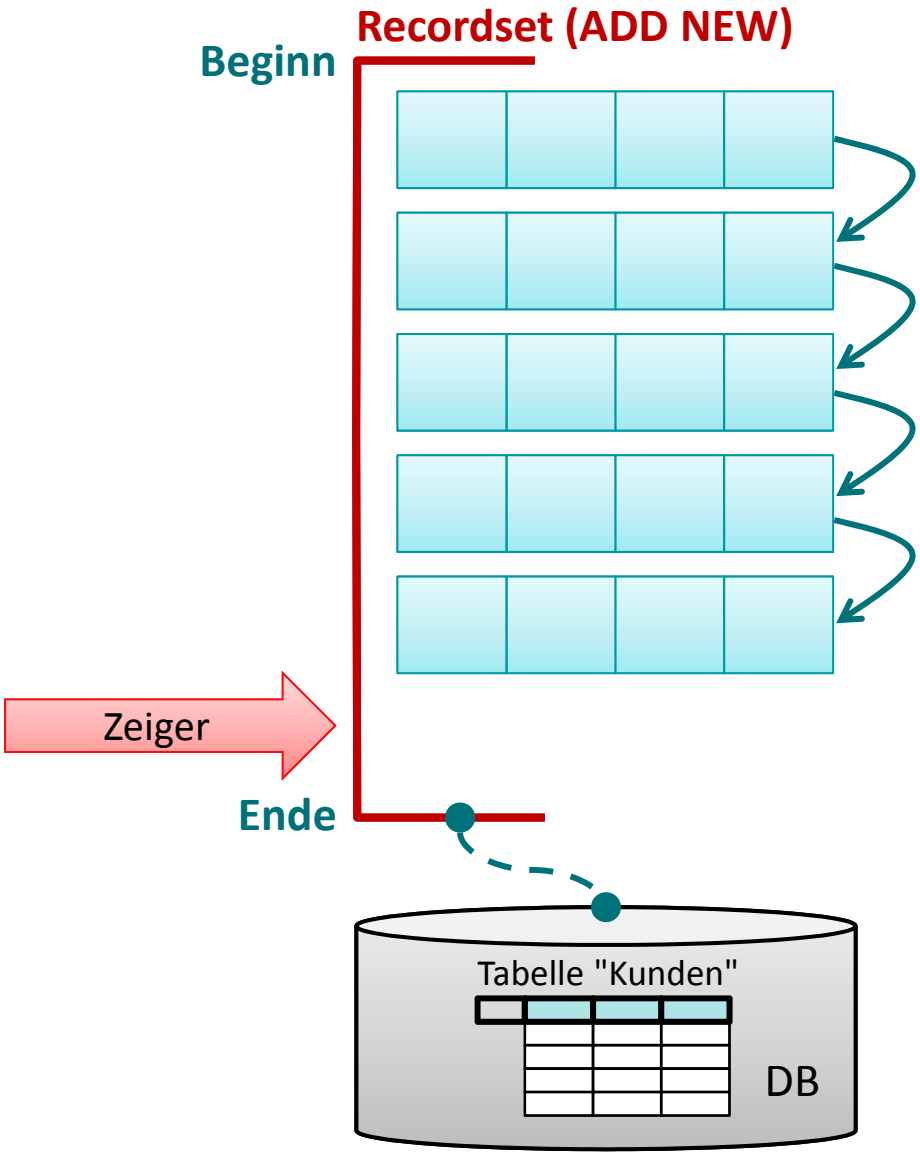


Einfügen von neuen Datensätzen

Ziel: Ein Datensatz soll Hinzugefügt werden

Ansatz

- Recordset als Dnyaset befüllen
- Anfügemodus des Recordset aktivieren

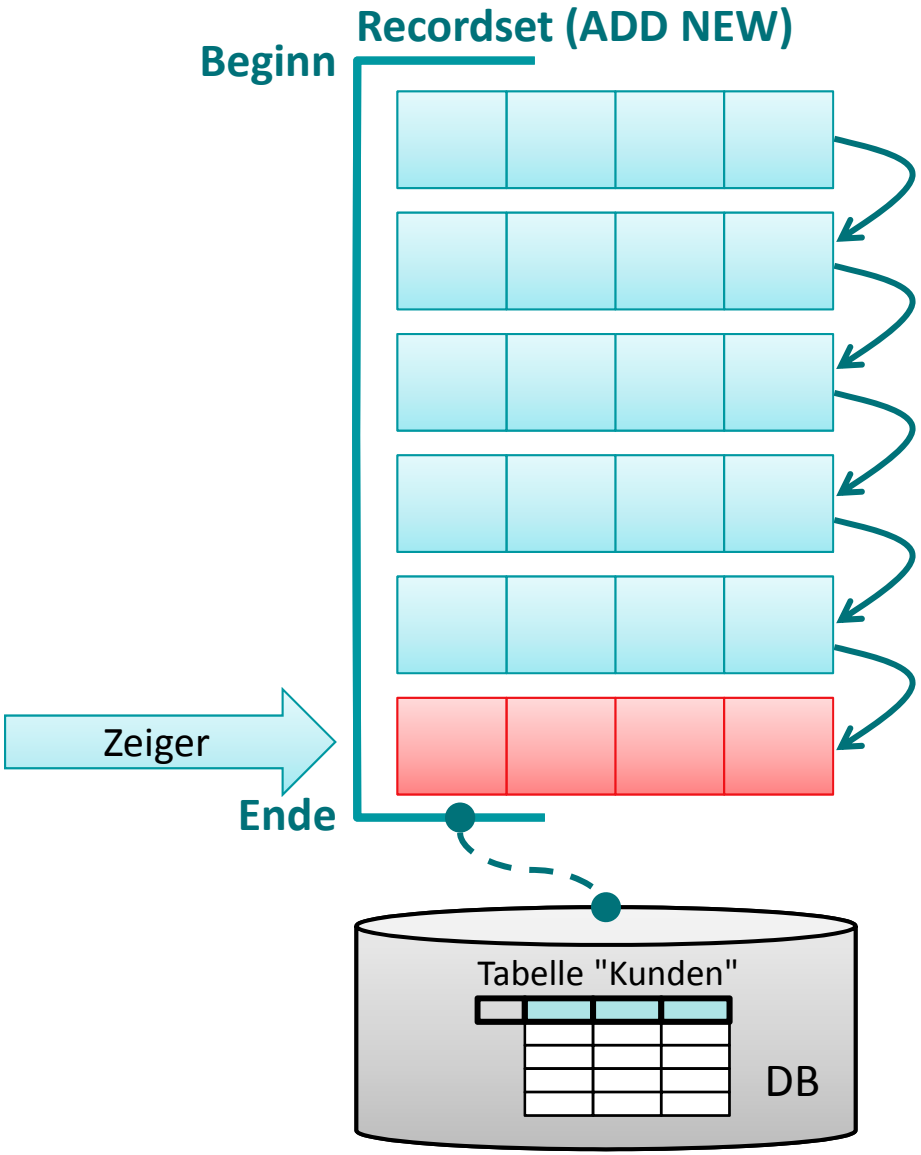


Einfügen von neuen Datensätzen

Ziel: Ein Datensatz soll Hinzugefügt werden

Ansatz

- Recordset als Dnyaset befüllen
- Anfügemodus des Recordset aktivieren
- Neuen Eintrag feldweise belegen (Reihenfolge egal)

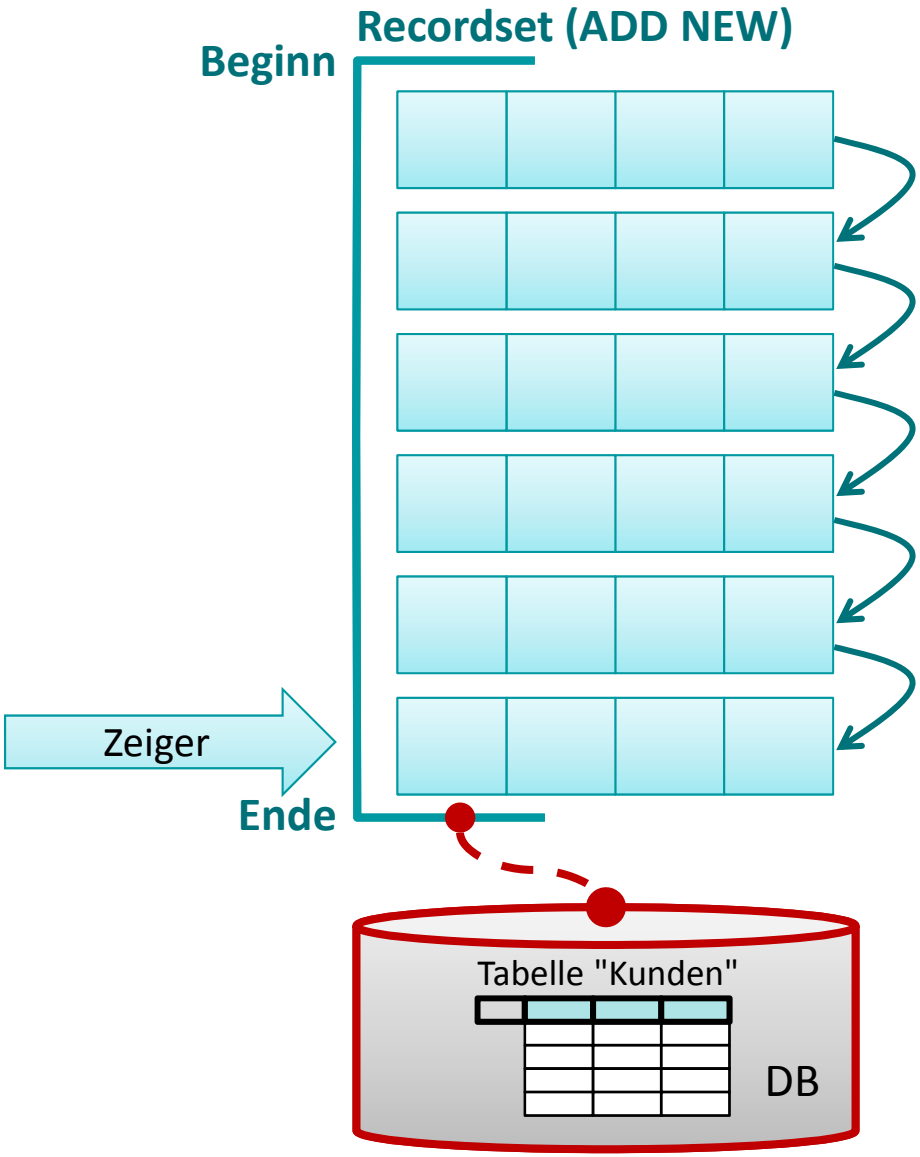


Einfügen von neuen Datensätzen

Ziel: Ein Datensatz soll Hinzugefügt werden

Ansatz

- Recordset als Dnyaset befüllen
- Anfügemodus des Recordset aktivieren
- Neuen Eintrag feldweise belegen (Reihenfolge egal)
- Aktualisierung abschließen

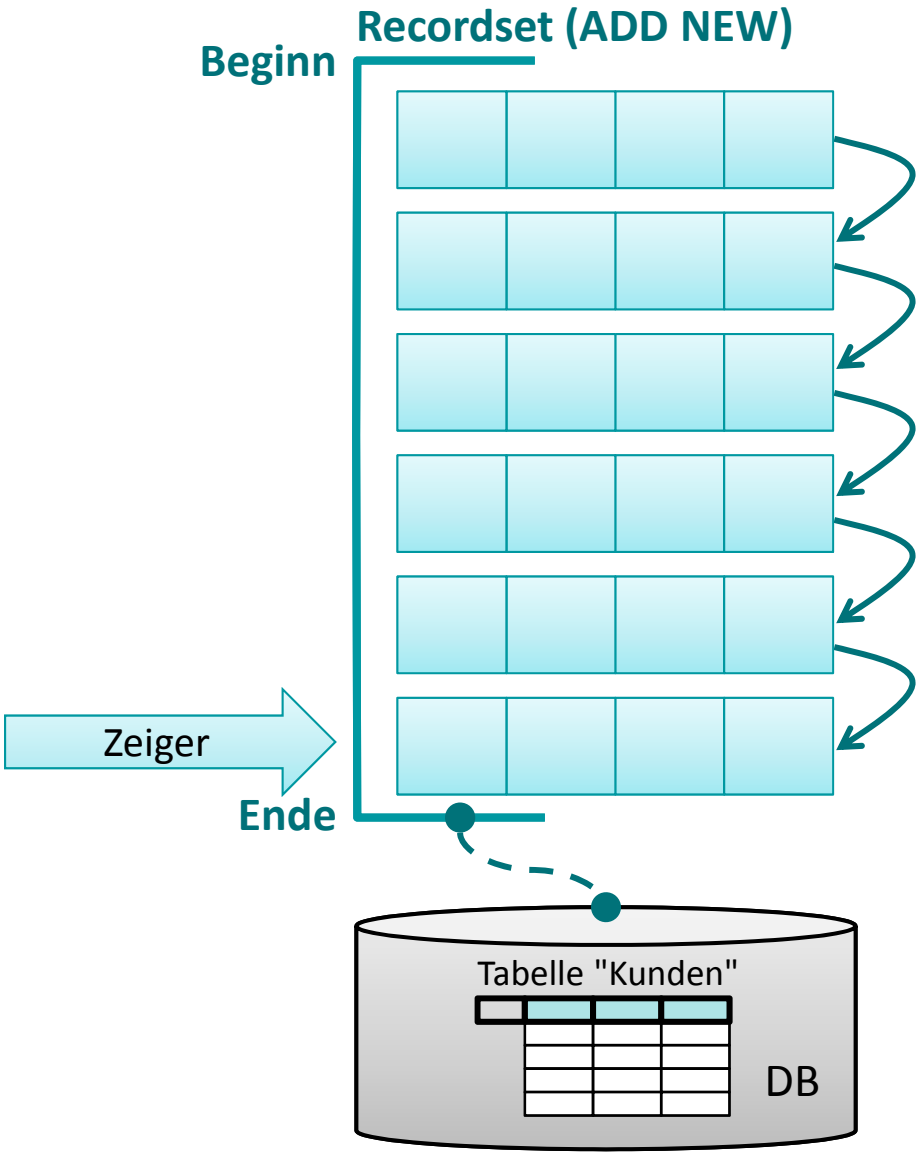


Einfügen von neuen Datensätzen

Ziel: Ein Datensatz soll Hinzugefügt werden

Ansatz

- Recordset als Dnyaset befüllen
- Anfügemodus des Recordset aktivieren
- Neuen Eintrag feldweise belegen (Reihenfolge egal)
- Aktualisierung abschließen



Ändern und Hinzufügen im Recordset

Ziel: Ein Datensatz soll Hinzugefügt werden

Ansatz

- Recordset als Dnyaset befüllen
- Anfügemodus des Recordset aktivieren
- Neuen Eintrag feldweise belegen (Reihenfolge egal)
- Aktualisierung abschließen

Modul

```
Option Compare Database
Option Explicit

Sub hinzufuegenKunde()
' Verbindung zur DB vorbereiten
Dim db As Database
Set db = CurrentDb
' Recordset deklarieren, initialisieren
Dim rcsKnd As Recordset
Set rcsKnd=db.OpenRecordset("tblKunden",
                            dbOpenDynaset)

' Anfügemodus aktivieren
rcsKnd.AddNew

 ' Felder des neuen Eintrags belegen
 rcsKnd.Fields("kndName") = "Schmidt"
 rcsKnd.Fields("kndVorname") = "Udo"

 ' Aktualisierung durchführen
 rcsKnd.Update


End Sub
```

Funktionen einer Datenbankschnittstelle

Funktionen einer Datenbankschnittstelle am Beispiel des Recordsets

- Abfragen von Datensätzen
  - Auswählen nach bestimmten Kriterien
  - Navigation über die gefundenen Datensätze
- Einfügen von neuen Datensätzen
- Ändern vorhandener Datensätze
- Löschen vorhandener Datensätze

Ändern und Hinzufügen im Recordset

Ziel: Ein Datensatz des Recordset soll geändert werden

Ansatz

- Recordset als Dynaset füllen
- nach zu änderndem Datensatz suchen
- Prüfen, ob Datensatz gefunden
- Wenn ja, dann Änderungsmodus des Recordset aktivieren
- Änderung des Feldes vornehmen
- Aktualisierung abschließen

Modul

```
Option Compare Database
Option Explicit

Sub aendernKunde()
' Verbindung zur DB vorbereiten
Dim db As Database
Set db = CurrentDb
' Recordset deklarieren, initialisieren
Dim rcsKnd As Recordset
Set rcsKnd=db.OpenRecordset("tblKunden",
                            dbOpenDynaset)
' Suchen nach Kunde mit ID4
rcsKnd.FindFirst ("kndIdPk = 4")
If rcsKnd.NoMatch Then
    Debug.Print "Kein Datensatz!"
 Else
    ' Änderungsmodus aktivieren
    rcsKnd.Edit
    ' Änderung vornehmen
    rcsKnd.Fields("kndName") = "Schmidt"
    ' Aktualisierung durchführen
    rcsKnd.Update
 End If
End Sub
```

Einfügen, Ändern und Löschen

Bearbeitungsmodus

- Änderungsmodus aktivieren

  `<einRecordset>.Edit`

- Einfügemodus aktivieren

  `<einRecordset>.AddNew`

- Bearbeitungsmodus abschließen

  `<einRecordset>.Update`

- Löschmodus aktvieren

  `<einRecordset>.Delete`

Funktionen einer Datenbankschnittstelle

Funktionen einer Datenbankschnittstelle am Beispiel des Recordsets

- Abfragen von Datensätzen
  - Auswählen nach bestimmten Kriterien
  - Navigation über die gefundenen Datensätze
- Einfügen von neuen Datensätze
- Ändern vorhandener Datensätze
- Löschen vorhandener Datensätze

Ein Recordset

- ist eine geordnete Menge von Datensätzen, die aus einer oder mehreren Tabelle einer Datenbank geladen werden.
- kann abhängig von seinem Typ
  - einen Schnappschuss des Datenbestandes repräsentieren
  - mit der Datenbank verbunden sein und Auswirkungen von Änderungen widerspiegeln
- besitzt einen Zeiger, mit dem über Datensätze navigiert werden kann (MoveNext, MovePrevious, ...)
- bietet Möglichkeit auf Werte des Elementes zuzugreifen, auf das der Zeiger zeigt (Fields)
- lässt nach Datensätzen suchen (FindFirst, FindNext, ...)
- kann in verschiedene Modus geschaltet werden
  - Änderungsmodus (Edit)
  - Hinzufüge-Modus (AddNew)
  - Löschmodus (Delete)

Prüfungsvorbereitung

Beispielhafte Aufgabe

- Welche Aufgaben hat eine Datenbankschnittstelle?
- Welche Ausgabe erzeugt diese Prozedur? Warum? Welche Daten enthält die Tabelle nach Ausführung der Prozedur? Warum?

```
Hier steht dann eine
richtige Prozedur mit
Recordset.
```

Eine Tabelle

Kunden	KndNr	Name	Ort	Alter
	123	Albers	Köln	18
	234	Berger	Köln	
	345	Yilmaz	Berlin	
	...	...	...	

Inhalt

Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Transaktionen

Definition: Folge von Datenbankoperationen,

- die hinsichtlich der Konsistenz/Integritätsbedingungen als atomare Einheit angesehen wird.[1]
- die ausgehend von einem konsistenten Zustand der Datenbank immer in einen konsistenten Zustand führt.[2]
- die mit besonderen Kommandos
  - begonnen,
  - erfolgreich abgeschlossen oder
  - nicht erfolgreich beendet wird

1) vgl. [4], S. 139
2) vgl. [2], S. 59f.

Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit

Transaktionen

ACID-Eigenschaften

- Atomarität
  - Transaktion wird entweder vollständig oder gar nicht ausgeführt
  - tritt bei einer Operation der Transkation ein Fehler auf, werden diese Operation und alle bereits (erfolgreich) ausgeführten Operationen zurückgesetzt
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit

Transaktionen

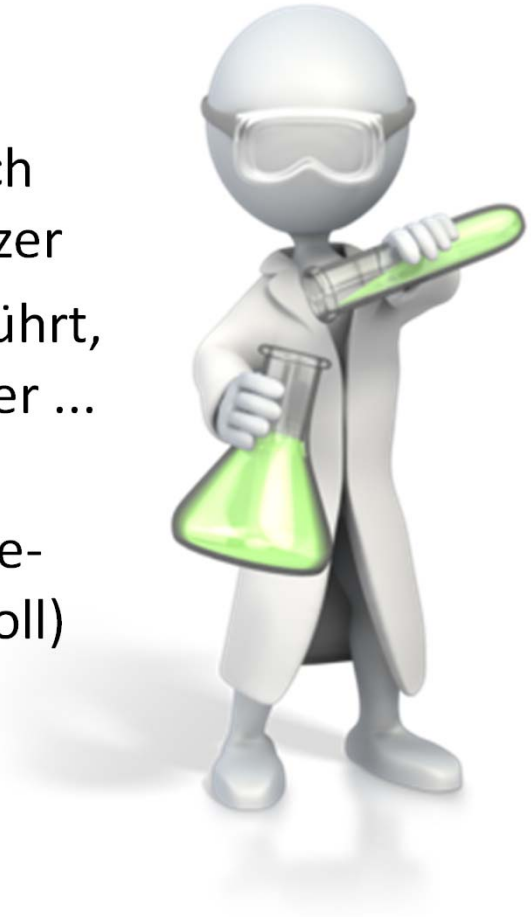
ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
  - Transaktion führt die Datenbank stets von einem konsistenten Zustand in den nächsten konsistenten Zustand
  - Vor und nach der Ausführung der Transaktion sind stets alle Integritätsbedingungen erfüllt
- Isolation
- Dauerhaftigkeit

Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
  - Transaktionen laufen isoliert voneinander ab, d.h. aus Sicht des Benutzers verhält sich Datenbank so, als sei er der einzige Benutzer
  - parallele Transaktionen werden so ausgeführt, als würden sie nacheinander ablaufen, aber ...
  - DBMS stellt Isolation durch verschiedene Mechanismen sicher (z.B. aus Performance-Gründen nicht immer nacheinander sinnvoll)
- Dauerhaftigkeit



Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit
  - abgeschlossene Transaktionen müssen auch nach einem unmittelbar anschließenden Fehlerzustand gespeichert sein
  - insbesondere, auch wenn
    - Stromausfall zum Löschen des Cache-Speichers im RAM führt
    - Festplattendefekt die Datenbank-Datei zerstört

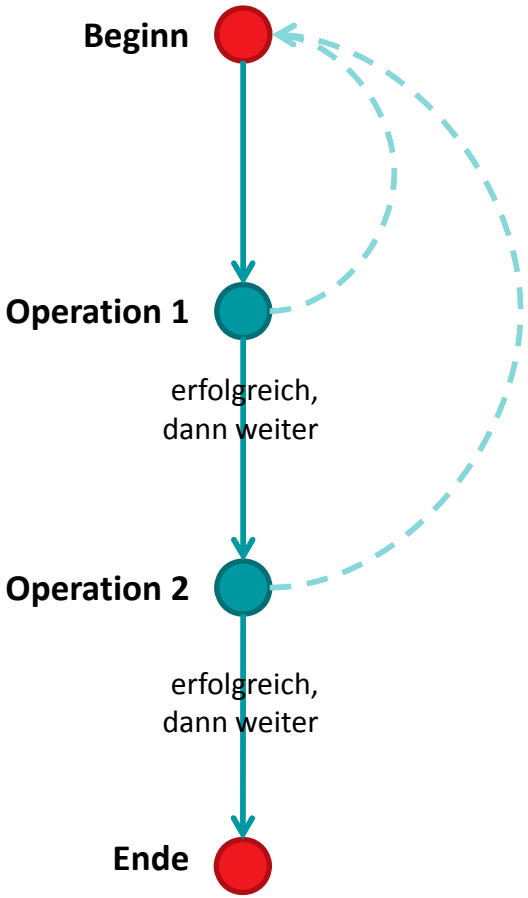
Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit

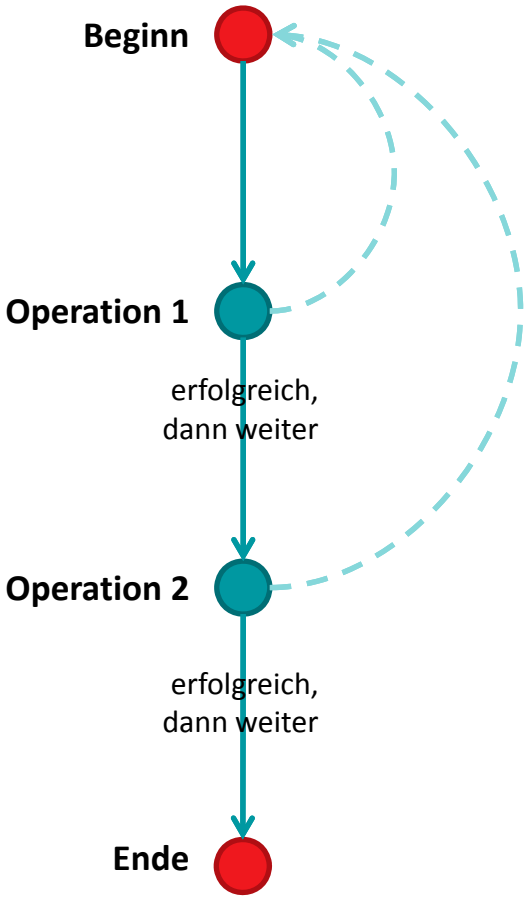
Transaktionen

Erfolgreich abgeschlossene Transaktion

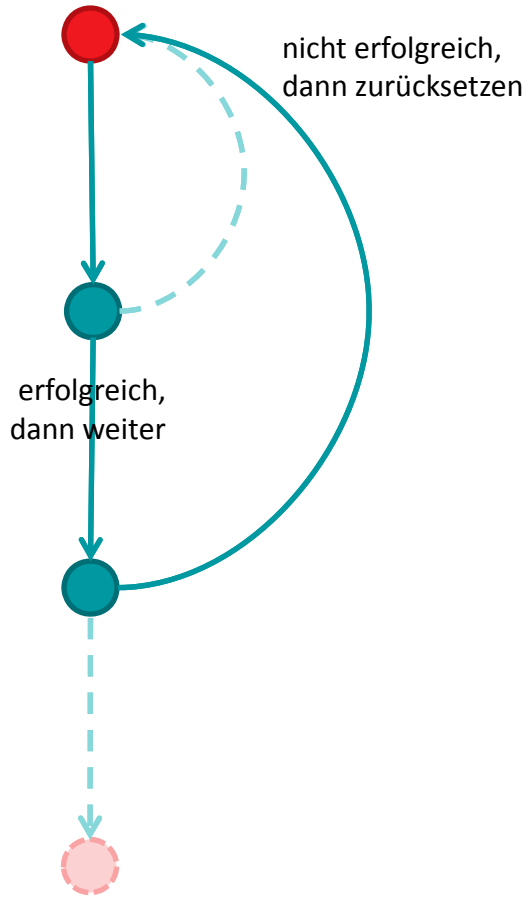


Transaktionen

Erfolgreich abgeschlossene Transaktion

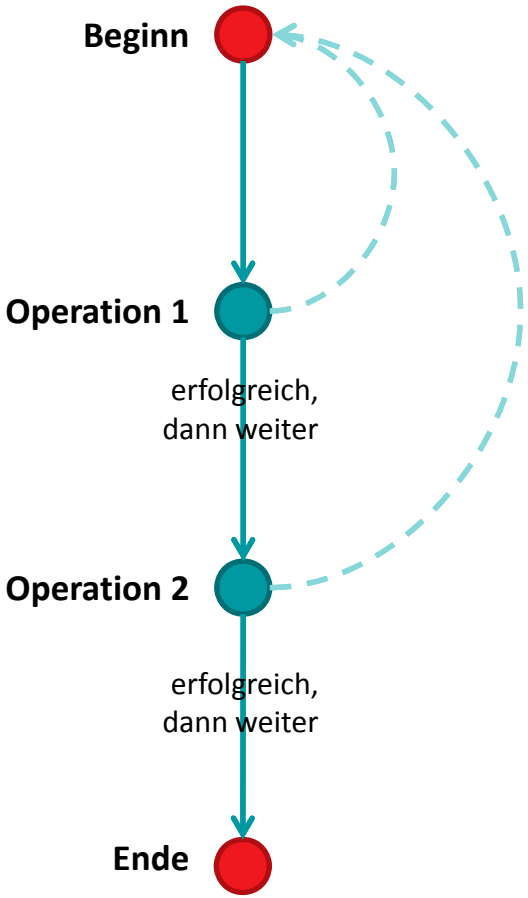


Nicht erfolgreich beendete Transaktion

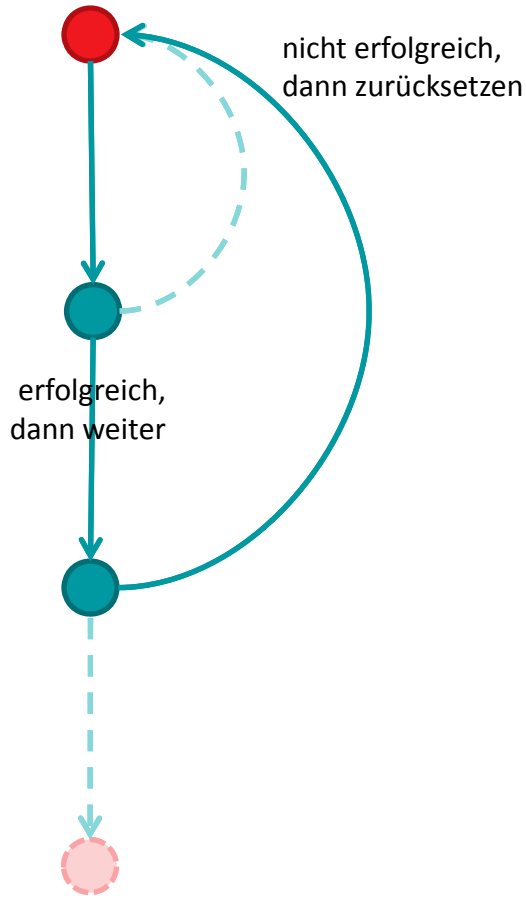


Transaktionen

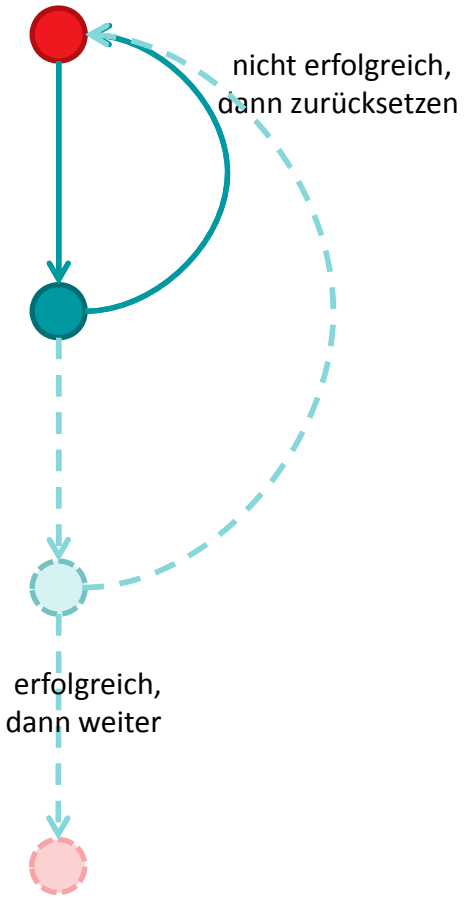
Erfolgreich abgeschlossene Transaktion



Nicht erfolgreich beendete Transaktion



Nicht erfolgreich beendete Transaktion



Transaktionen in SQL

SQL kennt besondere Befehle, mit denen Transaktionen

- begonnen
  - **`BEGIN TRANSACTION`** bzw. **`START TRANSACTION`**[1]
  - führt alle nachfolgenden SQL-Befehle innerhalb einer Transaktion aus
- erfolgreich abgeschlossen
  - **`COMMIT`** bzw. **`COMMIT TRANSACTION`**
  - schließt eine Transaktion ab, alle durchgeführten Operationen werden dauerhaft in der Datenbank wirksam
- nicht erfolgreich beendet
  - **`ROLLBACK`** bzw. **`ROLLBACK TANSACTION`**
  - schließt die Transaktion ab, verwirft alle durchgeführten Operationen

werden können.

1) BEGIN TRANSACTION/START TRANSACTION ist nicht Teil des standardisierten SQL-Sprachumfangs wird aber von den gängigen Datenbanken als Erweiterung unterstützt.

Transaktionen in MS Access

Stattdessen bietet MS Access in Zusammenhang mit dem Workspace-Objekt Funktionen zum

- beginnen einer Transaktion
- erfolgreichen Abschließen (Commit)
- erfolglosem Beenden (Rollback)

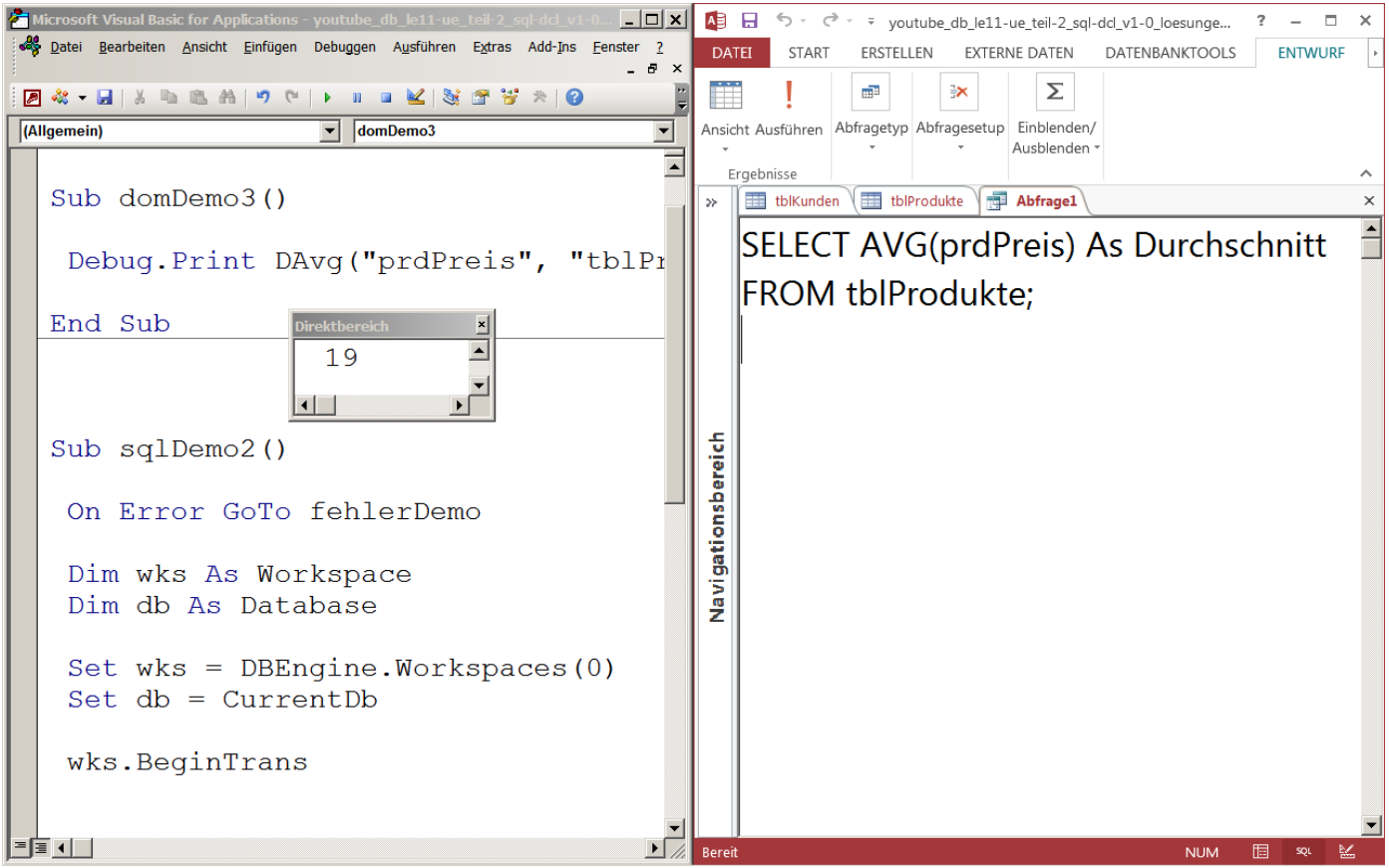
```
Sub demoTrans()

 On Error GoTo fehlerDemo

 Dim wks As Workspace
 Dim db As Database

 Set wks = DBEngine.Workspaces(0)
 Set db = CurrentDb

 wks.BeginTrans
 db.Execute "<Irgendein SQL>", _
        dbFailOnError
 '...
 wks.CommitTrans
 db.Close
 wks.Close
Exit Sub

fehlerDemo:
 wks.Rollback
 db.Close
 wks.Close

End Sub
```

Prüfungsvorbereitung

Beispielhafte Aufgaben

- Definieren Sie den Begriff (Datenbank-)Transaktion.
- Wofür steht ACID in Zusammenhang mit Transaktionen?
- Erläutern Sie die Eigenschaften Atomar, Konsistenz, Isoliertheit und Dauerhaftigkeit einer Transaktion.
- Betrachten Sie den folgenden Programmcode, der eine Transaktion enthält, und die gegebene Tabelle.
  - Welche Daten enthält die Tabelle, wenn die Transaktion erfolgreich abgeschlossen wird?
  - Welche Daten enthält die Tabelle, wenn die Verarbeitung in Zeile 7 einen Fehlerzustand erzeugt?

TODO


Microsoft Visual Basic for Applications - youtube_db_le11-ue_teil-2_sql-dcl_v1-0...
Datei Bearbeiten Ansicht Einfügen Debuggen Ausführen Extras Add-Ins Fenster ?
(Allgemein)
domDemo3
Sub domDemo3()
Debug.Print DAvg("prdPreis", "tblPr
End Sub
Direktbereich
19
Sub sqlDemo2()
On Error GoTo fehlerDemo
Dim wks As Workspace
Dim db As Database
Set wks = DBEngine.Workspaces(0)
Set db = CurrentDb
wks.BeginTrans
youtube_db_le11-ue_teil-2_sql-dcl_v1-0_loesunge...
DATEI START ERSTELLEN EXTERNE DATEN DATENBANKTOOLS ENTWURF
Ansicht Ausführen Abfragetyp Abfragesetup Einblenden/ Ausblenden
Ergebnisse
tblKunden tblProdukte Abfrage1
SELECT AVG(prdPreis) As Durchschnitt
FROM tblProdukte;
Navigationsbereich
Bereit
NUM
SQL

Inhalt

Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Anomalien

Anomalien in Datenbanken

- im Allgemeinen ein "Unregelmäßigkeit", Abweichung von üblichen Regeln
- können in Datenbanken Auslöser von Inkonsistenzen/Problemen sein

Arten

- Einfügeanomalie: Entität kann nicht oder nur schwer zu einer Tabelle hinzugefügt werden, weil für den Datensatz noch nicht bekannte Werte (z.B. als Teil des Primärschlüssels) benötigt werden.
- Änderungsanomalie: Tritt auf, wenn eine Entität mehrfach gespeichert wurde (Redundanz) und dadurch die mehrfache Änderung gleicher Werte notwendig ist, obwohl sich nur ein Sachverhalt geändert hat und damit das Risiko von Inkonsistenzen steigt.
- Löschanomalie: Beim Löschen einer Entität gehen Informationen im zugehörigen Datensatz verloren, die noch benötigt werden und nicht hätten gelöscht werden sollen.

Funktionale Abhängigkeit[1]

- Attribut *b* ist funktional abhängig vom Attribut *a* der gleichen Relation *R*, wenn zu jedem Wert von a höchstens ein Wert von b möglich ist
- Hinweis: *a* und *b* nicht nur als einzelne Attribute, sondern auch zusammengesetzt möglich
- Notation

$$a \rightarrow b$$

1) vgl. [2], S. 121 f.

Volle funktionale Abhängigkeit[1]

- Attribut *b* ist **voll funktional abhängig** von Attribut *a* der selben Relation *R*, wenn
  - es funktional abhängig ist von *a*
  - aber nicht funktional abhängig von einem Teil von *a*
- Hinweis: wenn *a* nicht zusammengesetzt ist, bedeutet funktionale Abhängigkeit immer auch volle funktionale Abhängigkeit
- Notation

$$a \Rightarrow b$$

1) vgl. [2], S. 121 f.

Determinante[1]

- Attribut ***d*** ist eine Determinante, wenn ein anderes Attribut der gleichen Relation voll funktional abhängig von d ist
- bedeutet vereinfacht, dass alle Attribute, von denen ein Doppelpfeil ausgeht, Determinanten sind
- Hinweis: ***d*** kann zusammengesetztes Attribut sein

1) vgl. [2], S. 126 f.

Normalformen

1. Normalform (1NF)

– Attribute einer Relation haben einen atomaren Wertebereich, d.h. insbesondere, dass es keine Wiederholungen von Werten innerhalb eines Attributwertes gibt.

2. Normalform (2NF)

– in 1. Normalform und

– alle Nicht-Schlüsselattribute sind vom gesamten Schlüssel voll-funktional abhängig

3. Normalform (3NF)

– in 2. Normalform und

– es gibt kein Nicht-Schlüsselattribut, das von anderen Nicht-Schlüsselattributen abhängig ist, jede Determinante also ein Schlüssel ist

Normalformen

1. Normalform

- Ziel: Vereinfachung der Aktualisierung und des Zugriffs
- Regel
  - Mehrere Datenwerte in einer Zelle sind nicht zulässig.
  - Alle Spalten dürfen in ihren Zellen jeweils nur atomare Werte enthalten.
- Ausgangssituation (Beispiel): **Ist das 1. Normalform?**

Verkäufer	VkNr	Name	Filiale	Plz	Ort	Produkte	Umsatz
	123	Albers	1	12345	Berlin	1. Kühlschrank, 2.Waschmaschine	120.000€
	234	Boehrs	1	12346	Berlin	1. TV, 2. Kühlschrank, 3. DVD	78.000€
	345	Dinkel	2	13456	Berlin	1. TV	56.000€
	456	Dinkels	2	13456	Berlin	1. Kühlschrank	12.000€

Normalformen

1. Normalform

– Ergebnis der Normalisierung (Beispiel):

- nur atomare Werte in allen Spalten
- Schlüssel Verkäufernummer (VkNr) nicht mehr eindeutig
- neue Schlüsselkandidaten als zusammengesetzte Schlüssel aus
  – VkNr + Priorität des von ihm verkauften Produktes (Prio)
  – VkNr + Bezeichnung des verkauften Produktes (Produkt)
- Wahl von VkNr + Prio als Primärschlüssel

Verkäufer	VkNr	Name	Filiale	Plz	Ort	Prio	Produkt	Umsatz
	123	Albers	1	12345	Berlin	1	Kühlschrank	61.000€
	123	Albers	1	12345	Berlin	2	Waschmaschine	59.000€
	234	Boehrs	1	12345	Berlin	1	TV	12.500€
	234	Boehrs	1	12345	Berlin	2	Kühlschrank	55.000€
	234	Boehrs	1	12345	Berlin	3	DVD	12.500€
	345	Dinkel	2	13456	Berlin	1	TV	55.000€
	456	Dinkels	2	13456	Berlin	1	Kühlschrank	12.000€

Normalformen

2. Normalform

- Ziele
  - Nur zusammengehörige Daten sind in einer Relation enthalten.
  - Jede Relation stellt nur einen Sachverhalt der Realität dar.
- Regel: in 1. Normalform und alle Nicht-Schlüsselattribute vom gesamten Schlüssel abhängig (volle funktionale Abhängigkeit, vom Primärschlüssel gehen Doppelpfeile aus)
- Ausgangssituation (Beispiel): **Ist das 2. Normalform?**

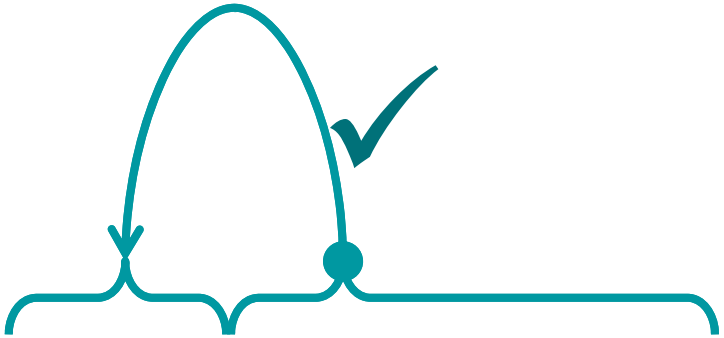
Verkäufer	VkNr	Name	Filiale	Plz	Ort	Prio	Produkt	Umsatz
	123	Albers	1	12345	Berlin	1	Kühlschrank	61.000€
	123	Albers	1	12345	Berlin	2	Waschmaschine	59.000€
	234	Boehrs	1	12345	Berlin	1	TV	12.500€
	234	Boehrs	1	12345	Berlin	2	Kühlschrank	55.000€
	234	Boehrs	1	12345	Berlin	3	DVD	12.500€
	345	Dinkel	2	13456	Berlin	1	TV	55.000€
	456	Dinkels	2	13456	Berlin	1	Kühlschrank	12.000€

Normalformen

2. Normalform

- Ergebnis der Normalisierung
  - Aufteilung auf zwei Relationen
  - Verkäufer mit Primärschlüssel VkNr, nicht zusammengesetzt
  - Produkte
    - mit zusammengesetztem Primärschlüssel VkNr, Prio
    - VkNr ist gleichzeitig Fremdschlüssel für Zuordnung zu Verkäufer

Verkäufer	VkNr	Name	Filiale	Plz	Ort
	123	Albers	1	12345	Berlin
	234	Boehrs	1	12345	Berlin
	345	Dinkel	2	13456	Berlin
	456	Dinkels	2	13456	Berlin



Produkte	VkNr	Prio	Produkt	Umsatz
	123	1	Kühlschrank	61.000€
	123	2	Waschmaschine	59.000€
	234	1	TV	12.500€
	234	2	Kühlschrank	55.000€
	234	3	DVD	12.500€
	345	1	TV	55.000€
	456	1	Kühlschrank	12.000€

Normalformen

3. Normalform

- Ziel: Nur unmittelbar zusammengehörige Daten in einer Relation enthalten, die genau einen Sachverhalt ausdrückt
- Regel
  - vereinfacht: es gibt kein Nicht-Schlüsselattribut, das von anderen Nicht-Schlüsselattributen abhängig ist (Doppelpfeile gehen nur von Schlüsselkandidaten aus)
  - formal: jede Determinante ist ein Schlüsselkandidat
- Ausgangssituation (Beispiel): **Ist das 3. NF?**

Verkäufer	VkNr	Name	Filiale	Plz	Ort
	123	Albers	1	12345	Berlin
	234	Boehrs	1	12345	Berlin
	345	Dinkel	2	13456	Berlin
	456	Dinkels	2	13456	Berlin

Produkte	VkNr	Prio	Produkt	Umsatz
	123	1	Kühlschrank	61.000€
	123	2	Waschmaschine	59.000€
	234	1	TV	12.500€
	234	2	Kühlschrank	55.000€
	234	3	DVD	12.500€
	345	1	TV	55.000€
	456	1	Kühlschrank	12.000€

Normalformen

3. Normalform

– Ergebnis der Normalisierung

- Weitere Relation Filiale
- Verkäufer mit Fremdschlüssel für Filiale

Filiale	FNr	Plz	Ort
	1	12345	Berlin
	2	13456	Berlin

Verkäufer	VkNr	Name	FNr
	123	Albers	1
	234	Boehrs	1
	345	Dinkel	2
	456	Dinkels	2

Produkte	VkNr	Prio	Produkt	Umsatz
	123	1	Kühlschrank	61.000€
	123	2	Waschmaschine	59.000€
	234	1	TV	12.500€
	234	2	Kühlschrank	55.000€
	234	3	DVD	12.500€
	345	1	TV	55.000€
	456	1	Kühlschrank	12.000€

Prüfungsvorbereitung

Beispielhafte Aufgaben

- In welcher Normalform befindet sich die folgende Relation?
- Warum befindet sich die Relation nicht in XYZ. Normalform? Bringen Sie sie in die XYZ. Normalform.

Inhalt

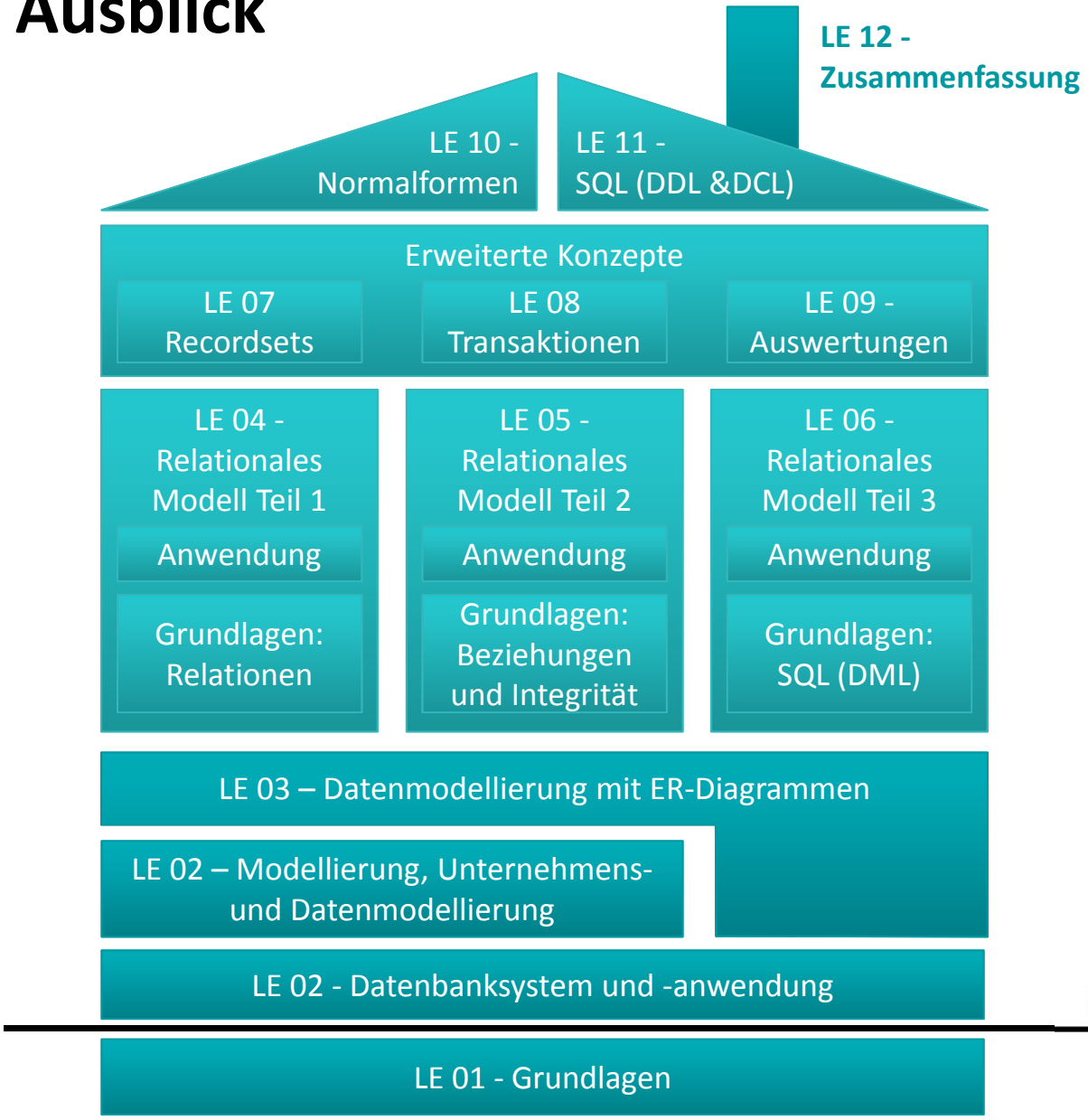
Ziel und Einordnung

Wiederholung

- Überblick und Einführung in Datenbanken
- Datenmodellierung insb. am Beispiel der ER-Modellierung
- Relationales Modell
- SQL
  - DML inkl. Statistikfunktionen
  - DDL und DCL
- Recordsets
- Transaktionen
- Normalformen

Ausblick

Ausblick


LE 12 - Zusammenfassung
LE 10 - Normalformen
LE 11 - SQL (DDL &DCL)
Erweiterte Konzepte
LE 07 Recordsets
LE 08 Transaktionen
LE 09 - Auswertungen
LE 04 - Relationales Modell Teil 1
Anwendung
Grundlagen: Relationen
LE 05 - Relationales Modell Teil 2
Anwendung
Grundlagen: Beziehungen und Integrität
LE 06 - Relationales Modell Teil 3
Anwendung
Grundlagen: SQL (DML)
LE 03 – Datenmodellierung mit ER-Diagrammen
LE 02 – Modellierung, Unternehmens- und Datenmodellierung
LE 02 - Datenbanksystem und -anwendung
LE 01 - Grundlagen

Ausblick

Viel Erfolg!

Quellen

[1] H. Krallmann: Systemanalyse im Unternehmen. 2. Aufl., Oldenbourg; 1996

[2] Edwin Schicker: Datenbanken und SQL. Eine praxisorientierte Einführung, Stuttgart, Teubner (1996).

[3] Kleuker, Stefan: Grundkurs Datenbankentwicklung. Von der Anforderungsanalyse zur komplexe Datenbankanfrage, 3. Aufl., Wiesbaden, Springer (2013).

[4] A. Fink, G. Schneiderreit, S. Voß: Grundlagen der Wirtschaftsinformatik, Physika-Verlag (Springer), 2001

BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Wirtschaftsinformatik 2
LE 12 – Zusammenfassung

Prof. Dr. Thomas Off

http://www.ThomasOff.de/lehre/beuth/wi2