BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Wirtschaftsinformatik 2
LE 06 – Relationales Modell (Teil 3)
SQL

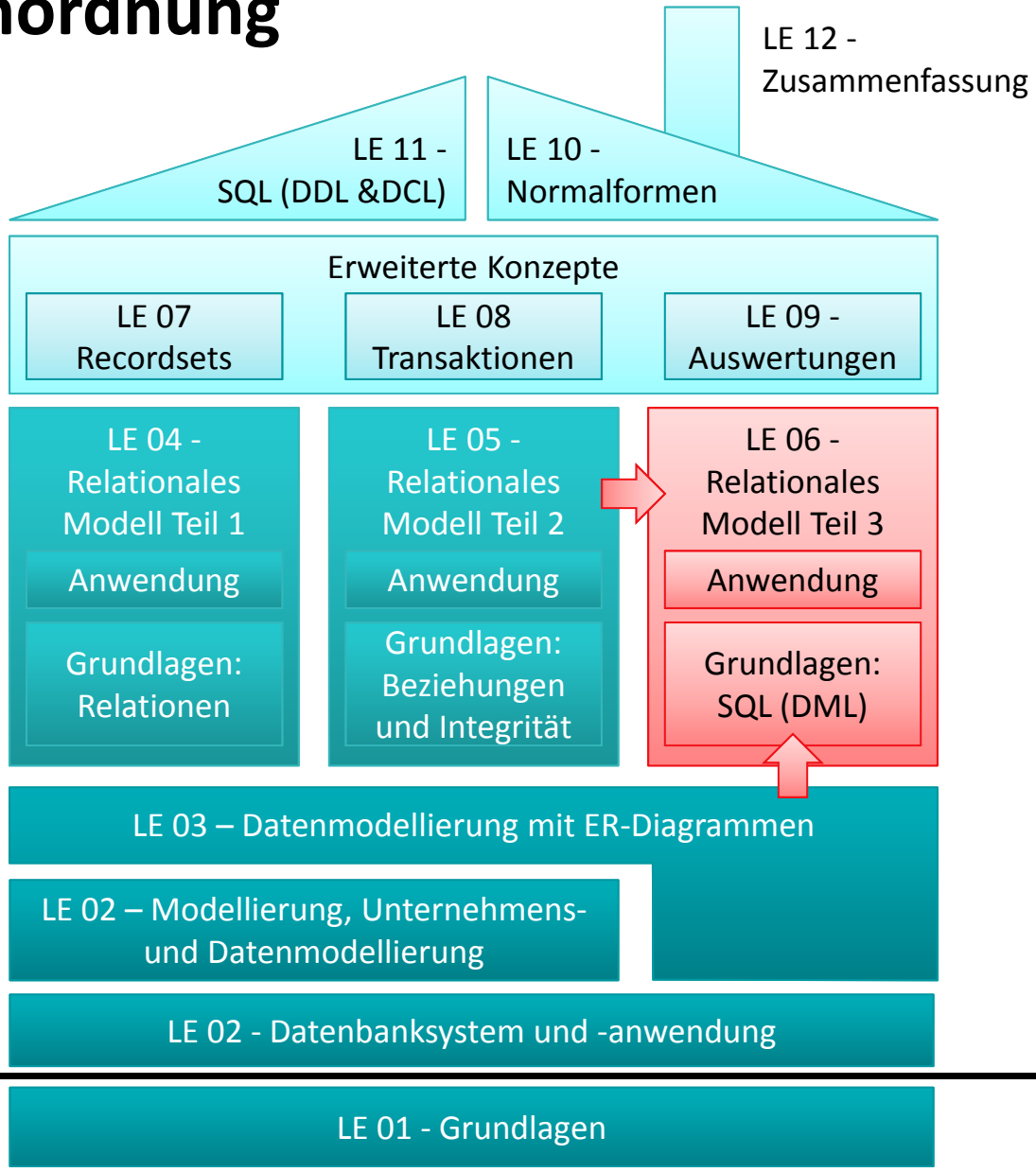
Prof. Dr. Thomas Off

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

Ziel

- Relationale Algebra und ihre wichtigsten Operationen kennenlernen
- Grundbestandteile der Datenbanksprache SQL kennenlernen
- SQL-Anweisungen der Data Manipulation Language (DML) nutzen
  - Abfrage von Daten mit SQL zur Umsetzung von Operationen der Relationalen Algebra
  - Hinzufügen, Löschen und Ändern von Daten mit SQL
- Praktische Anwendung in MS Access üben
  - Verwendung der verschiedenen SQL-Anweisungen in verschiedenen Abfragearten von MS Access
  - Integration von SQL-Anweisungen in eigene Funktionen, Prozeduren und Formulare

Einordnung


LE 12 - Zusammenfassung
LE 11 - SQL (DDL &DCL)
LE 10 - Normalformen
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

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel und Referenzielle Integrität

Relationale Algebra und ihre Operationen

SQL kennenlernen

- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

SQL in MS Access anwenden

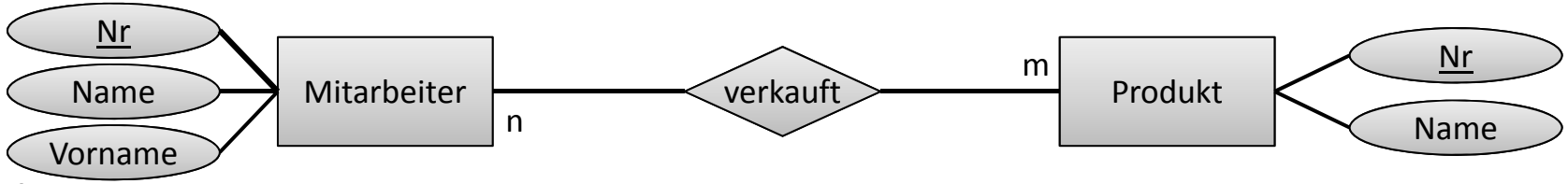
- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

Rückblick

Beziehungen werden über Fremdschlüssel hergestellt

– ER-Modell



– Relationen

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

Es gibt keinen Fremdschlüssel (ungleich "leer"), dessen Wert im zugehörigen Primärschlüssel nicht existiert (2. Integritätsregel, Referenzielle Integrität)

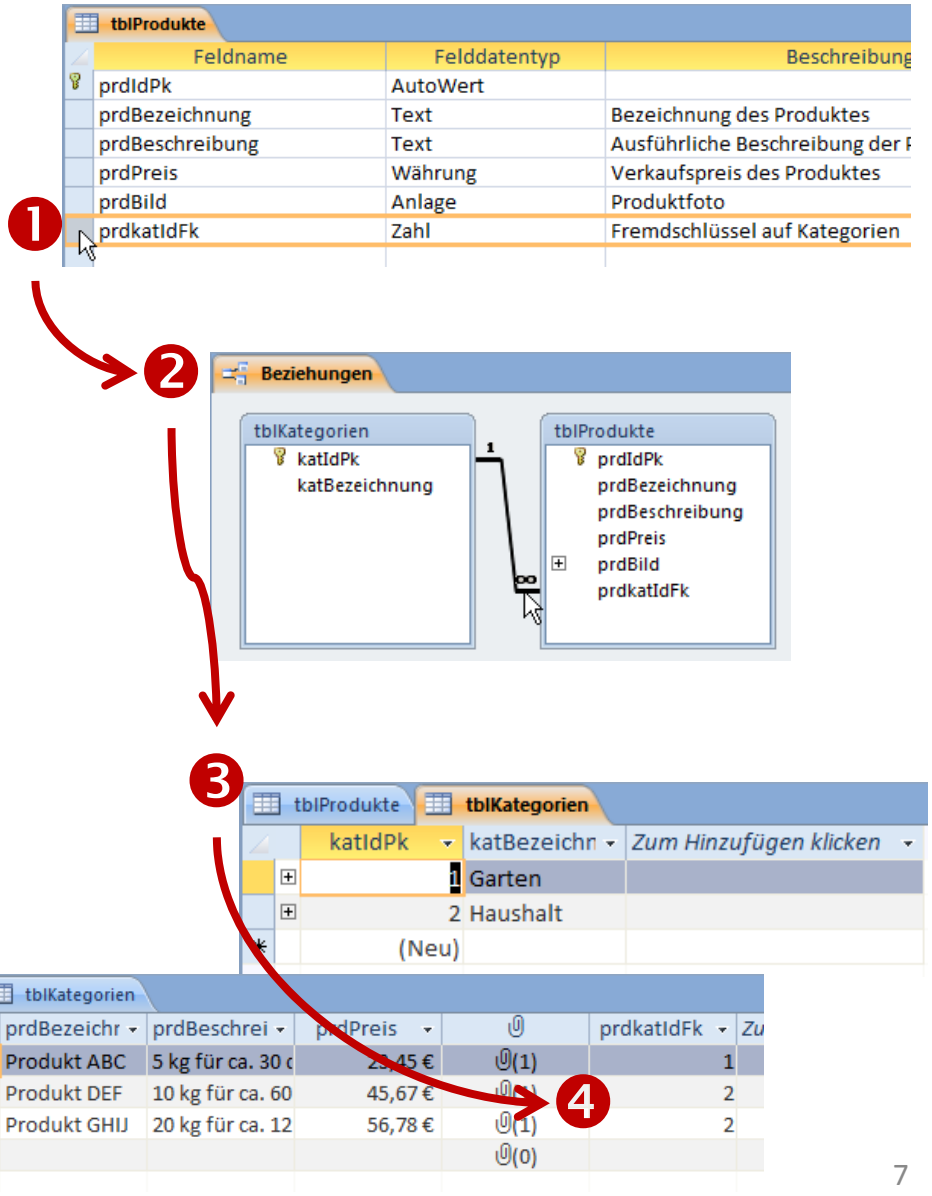
– was beim Entwurf der Relationen und

– beim Anlegen von Daten bedacht werden muss.

Rückblick

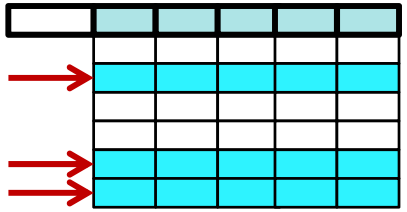
Umsetzung von Beziehungen in MS Access

- Schritt 1: Beziehungen zwischen vorhandenen Relationen werden über zusätzliche Spalten für Fremdschlüssel umgesetzt
- Schritt 2: Anlegen der Beziehung in der Beziehungsansicht, Festlegen der Integritätsbedingungen
- Schritt 3: Erfassen von Daten in der Reihenfolge in der die Beziehung dies erfordert



Rückblick

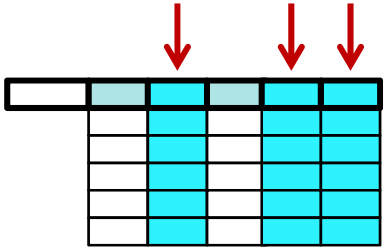
Selektion (Restriktion)



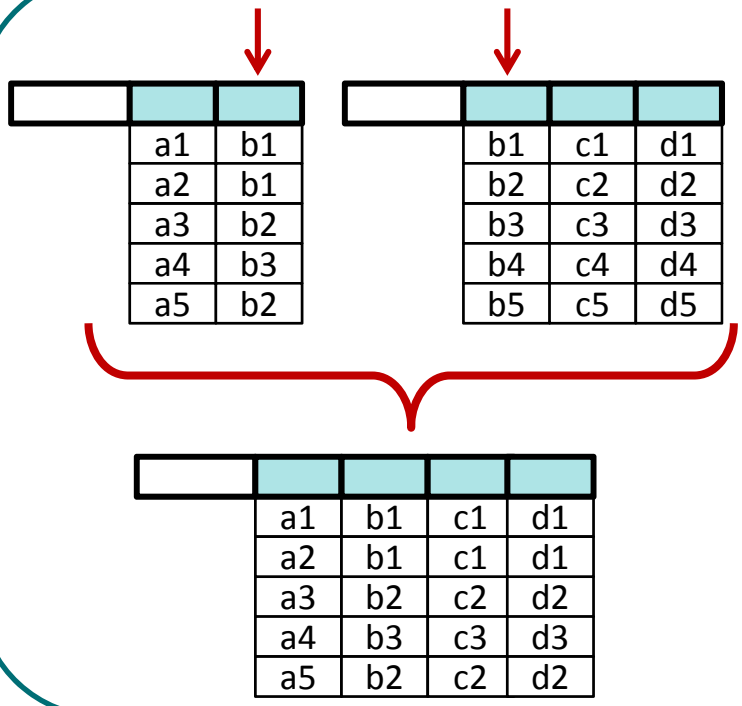
Nur relevante Tupel werden ausgewählt.

Projektion

Nur relevante Attribute werden ausgewählt.



Join



a1	b1	
a2	b1	
a3	b2	
a4	b3	
a5	b2	

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

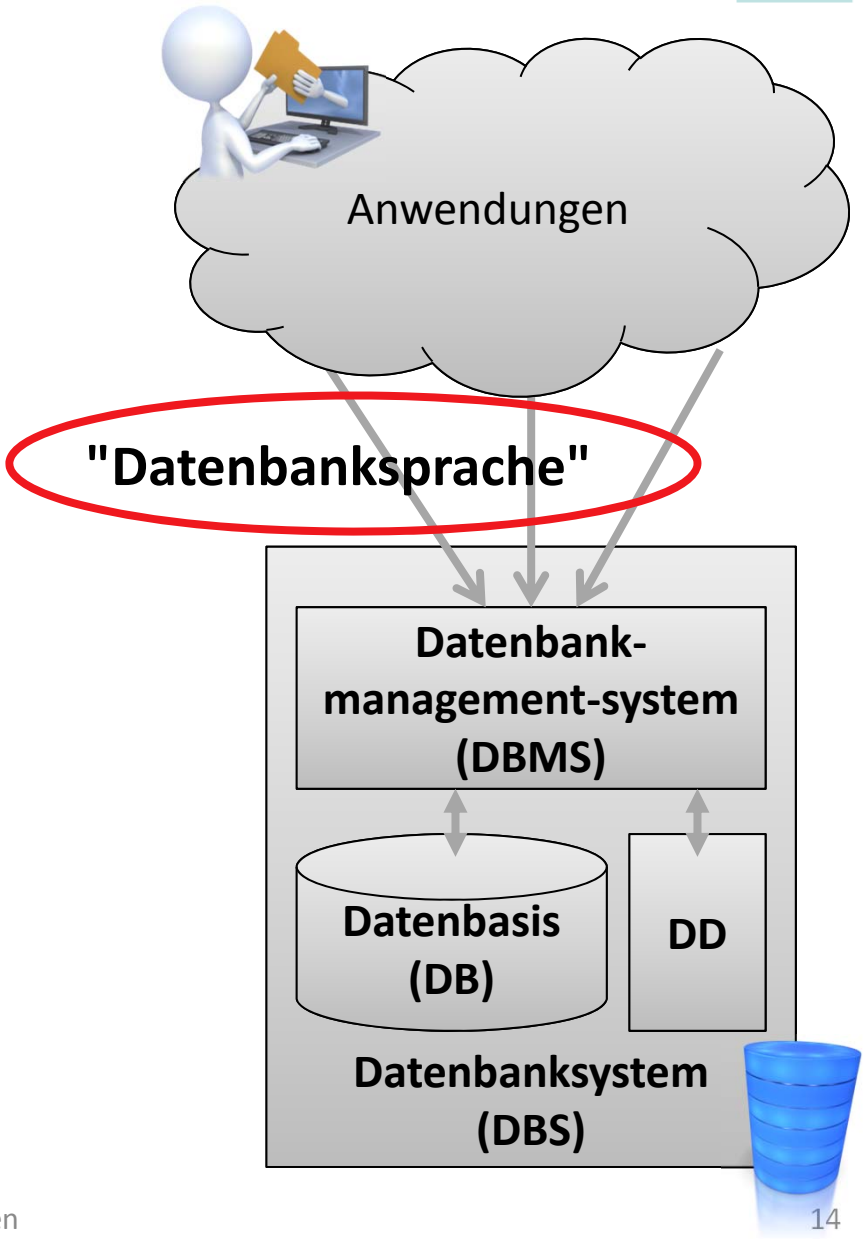
a1	b1	c1	d1
a2	b1	c1	d1
a3	b2	c2	d2
a4	b3	c3	d3
a5	b2	c2	d2

Rückblick

Komponenten eines Datenbanksystems

- Datenbasis (syn. Datenbank, DB)
- Data Dictionary (DD)
- Datenbankmanagement-system (DBMS)
- Datenbanksprache

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



Rückblick

Inhalt

Ziel und Einordnung

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel und Referenzielle Integrität

Relationale Algebra und ihre Operationen

SQL kennenlernen

- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

SQL in MS Access anwenden

- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

Relationale Algebra

Relationale Algebra

– ist Bestandteil des Relationalen Modells

– definiert eine Menge von Operationen, mit denen Relationen verknüpft und abgefragt werden können

Operationen der relationalen Algebra

– werden immer auf Relationen angewandt (Ausgangsrelationen)

– erzeugen als Ergebnis eine neue Relation (Ergebnisrelation)

- ist (im Gegensatz zu Ausgangsrelationen) nicht gespeichert, d.h. nur vorübergehend verfügbar und
- kann durch erneute Ausführung der Operation wieder erzeugt werden

– können nacheinander (geschachtelt) angewandt werden

Operationen der Relationalen Algebra

Mengenoperationen

– Vereinigung

– Schnitt

– Differenz

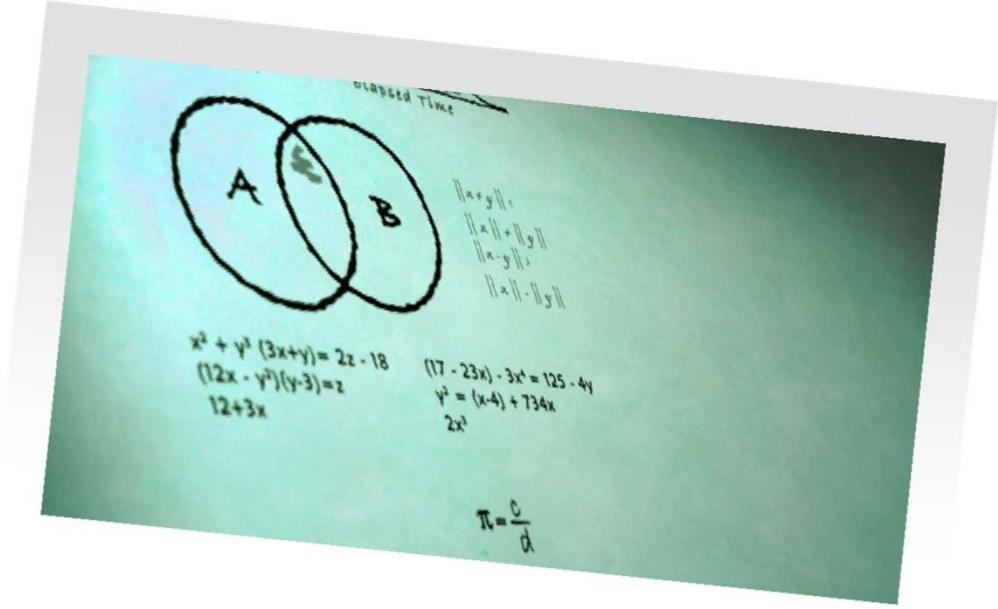
Projektion

Selektion/Restriktion

Kartesisches Produkt

Verbund (Join)

...



Mengenoperationen

Vereinigung

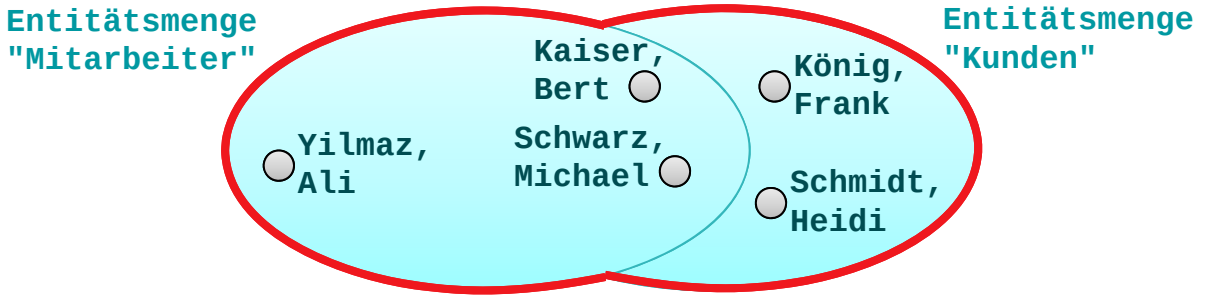
- Ausgangsrelationen müssen die gleichen Attribute mit passenden Datentypen haben (vereinigungskompatibel)
- Ergebnisrelation
  - hat alle Attribute der Ausgangsrelation und
  - umfasst alle Tupel der Ausgangsrelationen, wobei gleiche Tupel nur einmal enthalten sind
- ...

Mengenoperationen

Vereinigung

– ...

– Beispiel: Sowohl Kunden als auch Mitarbeiter sollen eine Weihnachtskarte bekommen. Die Vereinigungsmenge ist zu bilden.



Ausgangsrelationen

Mitarb	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	Yilmaz	Ali

Kunden	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	König	Frank
	Schmidt	Heidi

Vereinigung

Ergebnisrelation

Name	VName
Kaiser	Bert
Schwarz	Michael
Yilmaz	Ali
König	Frank
Schmidt	Heidi

Mengenoperationen

Schnitt

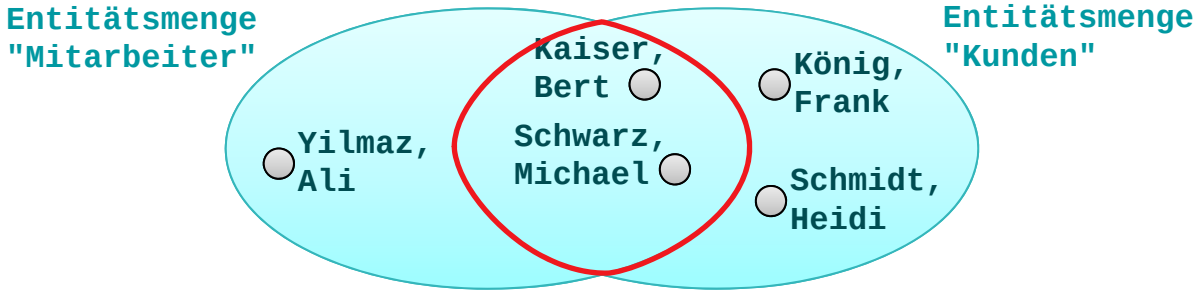
- Ausgangsrelationen müssen die gleichen Attribute mit passenden Datentypen haben (vereinigungskompatibel)
- Ergebnisrelation
  - hat alle Attribute der Ausgangsrelationen,
  - umfasst alle Tupel, die in beiden Relationen enthalten sind
- ...

Mengenoperationen

Schnitt

– ...

– Beispiel: Mitarbeiter, die auch Kunden sind, bekommen 10% Rabatt auf ihre Einkäufe am Jahresende ausbezahlt. Die Schnittmenge ist zu bilden.



Ausgangsrelationen

Mitarb	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	Yilmaz	Ali

Kunden	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	König	Frank
	Schmidt	Heidi

Schnitt

Ergebnisrelation

Name	VName
Kaiser	Bert
Schwarz	Michael

Mengenoperationen

Differenz

- Ausgangsrelationen müssen die gleichen Attribute mit passenden Datentypen haben (vereinigungskompatibel)
- Ergebnisrelation
  - hat alle Attribute der Ausgangsrelationen,
  - umfasst alle Tupel, die in der ersten Relationen enthalten sind und in der zweiten Relation nicht enthalten sind
- ...

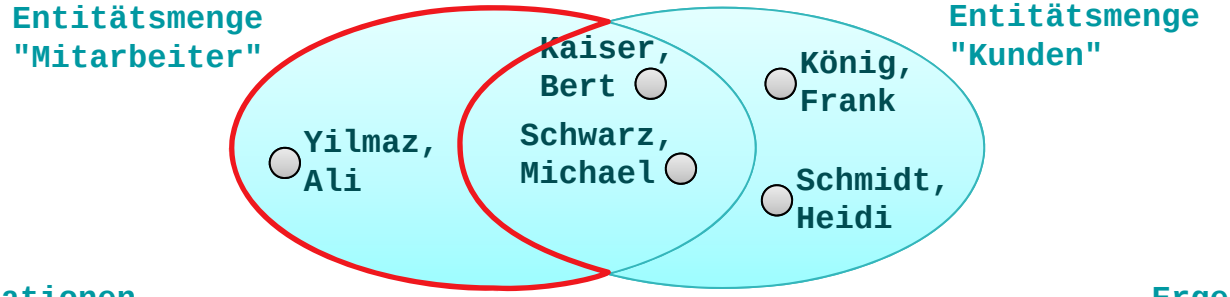
Mengenoperationen

Differenz

– ...

– Beispiel 1

- An alle Mitarbeiter, die noch keine Kunden sind, soll eine Information über den Mitarbeiterrabatt versandt werden. Die Differenz von Mitarbeitern ohne Kunden ist zu bilden.



Ausgangsrelationen

Mitarb	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	Yilmaz	Ali

Kunden	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	König	Frank
	Schmidt	Heidi

Differenz: Mitarbeiter ohne Kunden

Ergebnisrelation

Name	VName
Yilmaz	Ali

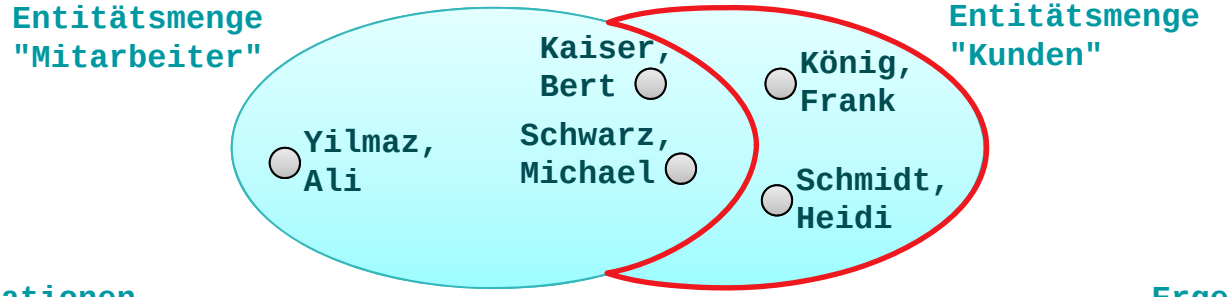
Mengenoperationen

Differenz

– ...

– Beispiel 2

- Für Kunden gibt es einen Weihnachtsrabatt, der nicht für Mitarbeiter gewährt wird. Die Differenz von Kunden ohne Mitarbeiter ist zu bilden.



Ausgangsrelationen

Kunden	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	König	Frank
	Schmidt	Heidi

Mitarb	Name	VName
	Kaiser	Bert
	Schwarz	Michael
	Yilmaz	Ali

Differenz: Kunden ohne Mitarbeiter

Ergebnisrelation

Name	VName
König	Frank
Schmidt	Heidi

Projektion und Selektion (Restriktion)

Projektion

- bestimmt Attribute der Ausgangsrelation, die in der Ergebnisrelation enthalten sein sollen
- Reihenfolge der Attribute in der Ergebnisrelation festgelegt
- entstehen durch Wegfallen von Attributen doppelte Tupel, so werden diese per Definition aus Ergebnisrelation entfernt
- Bespiel: Ort, Name (in dieser Reihenfolge) in Ergebnisrelation

Ausgangsrelationen

Kunden	ID	Name	VName	Plz	Ort	StrasseNr
	9	Kaiser	Bert	12345	Berlin	Bergstr. 8
	8	Schwarz	Michael	12345	Berlin	Schloßstr. 7
	7	Kaiser	Frank	12360	Berlin	Turmstr. 23
	6	Schmidt	Heidi	14482	Potsdam	Badstr. 9

Projektion (Ort,Name)

Ergebnisrelation

Ort	Name
Berlin	Kaiser
Berlin	Schwarz
Potsdam	Schmidt

Projektion und Selektion (Restriktion)

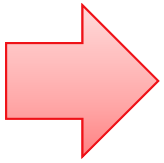
Selektion

- bestimmt die Tupel der Ausgangsrelation, die in der Ergebnisrelation enthalten sein sollen
- Formulierung einer Bedingung, die Attributwerte in Tupeln der Ausgangsrelation erfüllen müssen
- Ergebnisrelation umfasst nur solche Tupel, deren Attributwerte die Bedingung erfüllen
- Beispiel: Alle Kunden aus Berlin.

Ausgangsrelationen

Kunden	ID	Name	VName	Ort
	9	Kaiser	Bert	Berlin
	8	Schwarz	Michael	Berlin
	7	König	Frank	Berlin
	6	Schmidt	Heidi	Potsdam

Selektion
(Ort = Berlin)



Ergebnisrelation

ID	Name	VName	Ort
9	Kaiser	Bert	Berlin
8	Schwarz	Michael	Berlin
7	König	Frank	Berlin

Kartesisches Produkt

Ergebnisrelation hat alle Attribute der Ausgangsrelationen und umfasst alle möglichen Kombinationen von Tupeln der Ausgangsrelationen

Ausgangsrelationen

Namen	Name
	Kaiser
	Schwarz
	König
	Schmidt

VNamen	VName
	Thomas
	Mike
	Susi

Kartesisches Produkt

Ergebnisrelationen

Name	VName
Kaiser	Thomas
Kaiser	Mike
Kaiser	Susi
Schwarz	Thomas
Schwarz	Mike
Schwarz	Susi
König	Thomas
König	Mike
König	Susi
Schmidt	Thomas
Schmidt	Mike
Schmidt	Susi

Verbund (Join)

Allgemeine Form (Theta Join)

Gleichverbund (Equi Join)

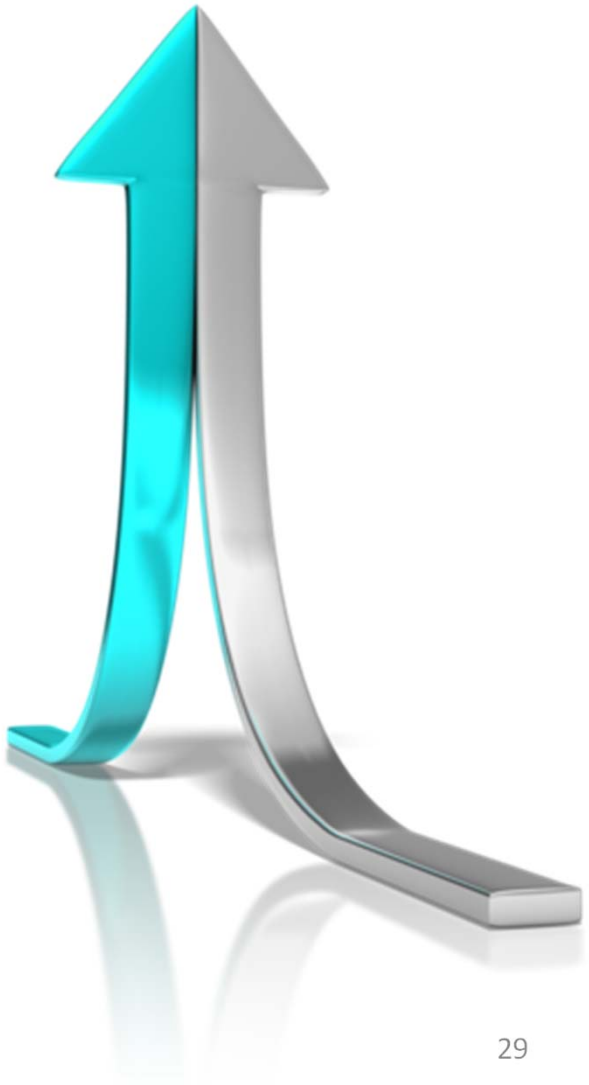
Natürlicher Verbund (Natural Join)

Innerer Verbund (Inner Join)

Äußerer Verbund (Outer Join)

– Linker äußerer Verbund (Left Outer Join)

– Rechter äußerer Verbund (Right Outer Join)



Verbund (Join)

Allgemeine Form (Theta Join)

- Verbindung von Ausgangsrelationen anhand der Werte ausgewählter Attribute
- Formulierung einer Bedingung, die diese Attributwerte erfüllen müssen

Verbund (Join)

Gleichverbund (Equi Join)

- Verbindung von Ausgangsrelationen anhand gleicher Werte der ausgewählten Attribute
- Nur wenn die Werte der ausgewählten Attribute in den Tupeln gleich sind, ist das durch eine Verbindung entstehende neue Tupel in der Ergebnisrelation enthalten
- Häufig wird in der praktischen Anwendung zusätzlich verlangt, dass die am Vergleich beteiligten Attribute gleiche Bezeichnungen haben müssen
- ...

Verbund (Join)

Gleichverbund (Equi Join)

– ...

– Beispiel: Produkte.LID = Lieferanten.LID

Ausgangsrelationen

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

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

Ergebnisrelation

PID	Name	LID	LID	Firma	Ort
123	Multi AB	987	987	Müller AG	Berlin
234	Flexi 123	987	987	Müller AG	Berlin
345	Mega+	876	876	Meier GmbH	Potsdam

Verbund (Join)

Natürlicher Verbund (Natural Join)

- Verbindung von Ausgangsrelationen anhand gleicher Werte ausgewählter Attribute (analog zum Gleichverbund)
- die Attribute, deren Werte verglichen wurden, sind nur einmal in der Ergebnisrelation enthalten
- Sehr häufig wird in der praktischen Anwendung zusätzlich verlangt, dass die am Vergleich beteiligten Attribute gleiche Bezeichnungen haben müssen
- ...

Verbund (Join)

Natürlicher Verbund (Natural Join)

– ...

– Beispiel: Beispiel: Produkte.LID = Lieferanten.LID

Ausgangsrelationen

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

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

Ergebnisrelation

PID	Name	LID	Firma	Ort
123	Multi AB	987	Müller AG	Berlin
234	Flexi 123	987	Müller AG	Berlin
345	Mega+	876	Meier GmbH	Potsdam

Verbund (Join)

Innerer Verbund (Inner Join)

- Verbindung von Ausgangsrelationen anhand der Werte ausgewählter Attribute
- Formulierung einer Bedingung, die diese Attributwerte erfüllen müssen
- Nur wenn die Werte der ausgewählten Attribute in den verglichenen Tupeln die Bedingung erfüllen, ist das durch eine Verbindung entstehende neue Tupel in der Ergebnisrelation enthalten
- Hinweis: Im Folgenden prüft die Bedingung auf Gleichheit. Es muss aber im Inner Join nicht immer Gleichheit sein! Deshalb erscheinen beide in der Bedingung geprüften Spalten in der Ergebnisrelation.

Verbund (Join)

Innerer Verbund (Inner Join)

– ...

– Beispiel: 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

Ergebnisrelation

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

Verbund (Join)

Äußerer Verbund (Outer Join)

- Verbindung von Ausgangsrelationen anhand der Werte ausgewählter Attribute
- Formulierung einer Bedingung, die diese Attributwerte erfüllen müssen
- Im Ergebnis sind mindestens die Tupel einer der beteiligten Relation enthalten und wurden ergänzt, um
  - Attributwerte aus der anderen Relation, wenn die Attributwerte der verglichenen Tupel die Bedingung erfüllen
  - NULL-Werte (leere Attributwerte), wenn die Attributwerte der verglichenen Tupel die Bedingung nicht erfüllen
- Es werden der linke und der rechte äußere Verbund unterschieden (nächste Folie).

Verbund (Join)

Linker äußerer Verbund (Left Outer Join)

- Mindestens die Tupel der linken Relation im Ergebnis vorhanden; wenn sie keinen Partner haben, dann mit NULL
- Beispiel: Produkte.LID = Lieferaten.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
	654	Berg GmbH	Berlin

Mindestens alle linken Tupel

Ergebnisrelation

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

Verbund (Join)

Rechter äußerer Verbund (Right Outer Join)

- Mindestens die Tupel der rechten Relation im Ergebnis vorhanden; wenn sie keinen Partner haben, dann mit NULL
- Beispiel: Produkte.LID = Lieferaten.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
	654	Berg GmbH	Berlin

Ergebnisrelation

PID	Name	LID	ID	Firma	Ort
123	Multi AB	987	987	Müller AG	Berlin
234	Flexi 123	987	987	Müller AG	Berlin
345	Mega+	876	876	Meier GmbH	Potsdam
			765	Bach&Sohn	Cottbus
			654	Berg GmbH	Berlin

Mindestens alle rechten Tupel

Verbund (Join)

Hinweise

- Die vorherigen Darstellungen zeigten immer
  - zwei Tabellen, der Verbund ist aber analog auch mit mehreren Tabellen möglich (indem er zunächst für die ersten zwei Tabellen durchgeführt wird und das Ergebnis mit der nächsten Tabelle verbunden wird usw.)
  - die Verbindung anhand einer Bedingung für Werte von zwei Attributen, mehrere Bedingungen und mehrere Attribute sind auch möglich
- Es gibt weitere Arten des Verbunds, die hier nicht relevant sind
  - Auto-/Self Join einer Relation mit sich selbst (z.B. Person mit ihrem Ehepartner, der ebenfalls als Person gespeichert ist)
  - Semi Join: Natürlicher Verbund ohne die zusätzlichen Attribute der rechten Relation
  - …

Verbund (Join)

Allgemeine Form (Theta Join)

Gleichverbund (Equi Join)

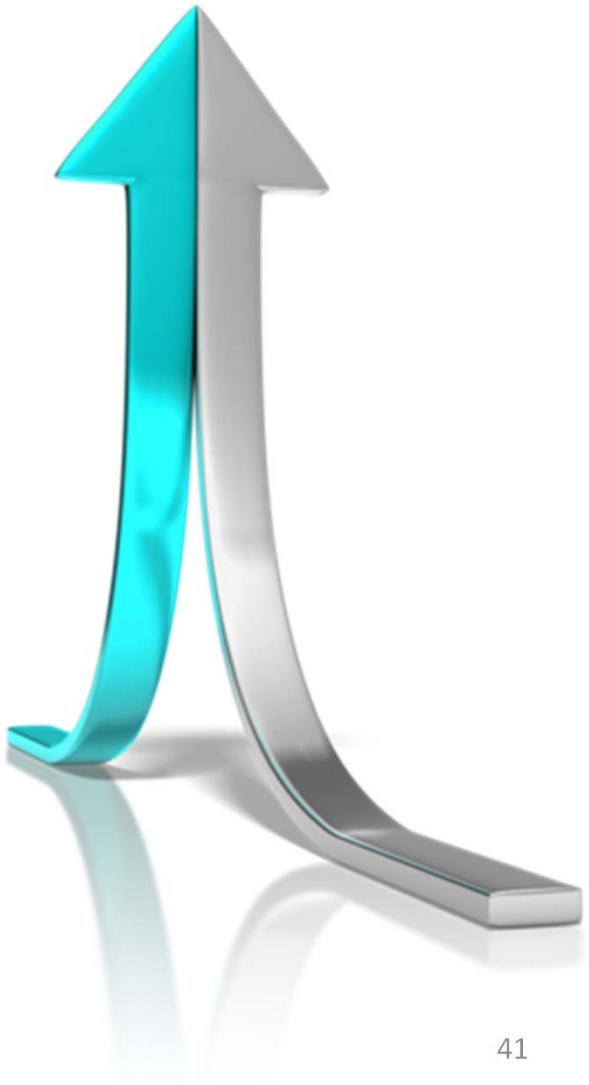
Natürlicher Verbund (Natural Join)

Innerer Verbund (Inner Join)

Äußerer Verbund (Outer Join)

– Linker äußerer Verbund (Left Outer Join)

– Rechter äußerer Verbund (Right Outer Join)



Operationen der Relationalen Algebra

Mengenoperationen

– Vereinigung

– Schnitt

– Differenz

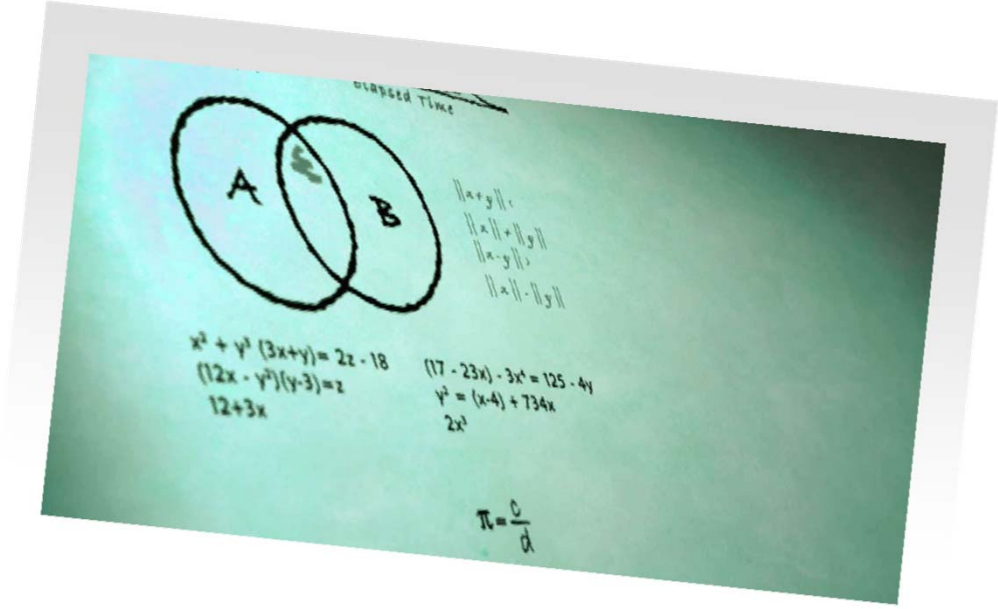
Projektion

Selektion/Restriktion

Kartesisches Produkt

Verbund (Join)

...



Inhalt

Ziel und Einordnung

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel und Referenzielle Integrität

Relationale Algebra und ihre Operationen

SQL kennenlernen

- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

SQL in MS Access anwenden

- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

Bestandteile des SQL-Sprachumfangs

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

- SELECT
- INSERT
- UPDATE
- DELETE

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

Thema von LE 11

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

Thema von LE 11

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

- vorhandene Tupel werden gelesen
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

- neue Tupel werden zur Relation hinzugefügt

Ändern

- vorhandene Tupel werden geändert
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

- vorhandene Tupel werden gelöscht
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Auswählen von Tupeln mit SQL

SQL-Syntax ähnlich der natürlichen Sprache (in Englisch), d.h. es kann bspw. formuliert werden

- "Wähle alles aus Tabelle *Kunden*."
- "Wähle *Name*, *Vorname* aus Tabelle *Kunden*." (Projektion)
- "Wähle alle aus der Tabelle *Kunden* bei denen der *Ort 'Berlin'* ist." (Selektion)
- "Wähle *Gesamtpreis* der *Bestellung* und *Name* sowie *Vorname* des *Kunden*, der die Bestellung bestellt hat." (Join)

Auswählen von Tupeln mit SQL

Allgemeine Abfrage in SQL (Bsp. 1):

– Deutsch: "Wähle alle Spalten aus Tabelle *Kunden*."

– Englisch: "Select all Colums from table *Kunden*."

– SQL: **SELECT * FROM *kunden*;**

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 * FROM Kunden;
```

Ergebnis

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

Auswählen von Tupeln mit SQL

Projektion in SQL (Bsp. 2):

– Deutsch: "Wähle *Name*, *Vorname* aus Tabelle *Kunden*."

– Englisch: "Select *Name*, *Vorname* from table *Kunden*."

– SQL: **`SELECT name, vorname FROM kunden;`**

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

Projektion

```
SELECT Name, Vorname FROM Kunden;
```

Ergebnis

Name	Vorname
Albers	Willi
Boehrs	Vera
Dinkel	Ulrike
Dinkels	Thomas
Esser	Thomas

Auswählen von Tupeln mit SQL

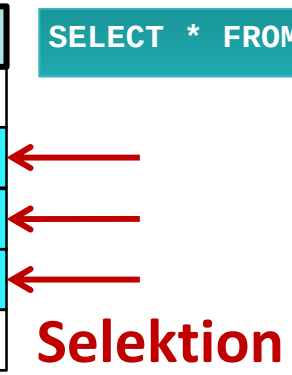
Selektion in SQL (Bsp. 3):

- Deutsch: "Wähle alle Spalte aus der Tabelle *Kunden* bei denen der *Ort 'Berlin'* ist."
- Englisch: "Select all colums from table *Kunden* where *Ort* is *'Berlin'*."
- SQL:

```
SELECT * 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

```
SELECT * FROM Kunden WHERE Ort='Berlin';
```

Ergebnis

KndNr	Name	Vorname	Ort
234	Boehrs	Vera	Berlin
345	Dinkel	Ulrike	Berlin
456	Dinkels	Thomas	Berlin

Auswählen von Tupeln mit SQL

Innerer Verbund (Inner Join) in SQL (Bsp. 5):

- *Name* sowie *Vorname* des *Kunden*, der eine Bestellung bestellt hat und den *Gesamtpreis* der *Bestellung*.
- SQL (Variante 1):

```
SELECT kunden.name, kunden.vorname, bestellungen.gesamt
FROM kunden, bestellungen
WHERE kunden.kndnr = bestellungen.kndnr;
```

Ausgangsrelationen

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

Join

Bestellungen	BstID	KndNr	Gesamt	Datum
	987	456	80€	1.1.2012
	876	345	320€	7.3.2011
	765	234	120€	1.8.2011

```
SELECT
kunden.name,
kunden.vorname
bestellungen.gesamt
FROM
bestellungen, kunden
WHERE
kunden.kndnr =
bestellungen.kndnr;
```

Ergebnis

Name	Vorname	Gesamt
Boehrs	Vera	120€
Dinkel	Ulrike	320€
Dinkels	Thomas	80€

Auswählen von Tupeln mit SQL

Innerer Verbund (Inner Join) in SQL (Bsp. 5):

- *Name* sowie *Vorname* des *Kunden*, der eine Bestellung bestellt hat und den *Gesamtpreis* der *Bestellung*.
- SQL (Variante 2):

```
SELECT kunden.name, kunden.vorname, bestellungen.gesamt
FROM kunden INNER JOIN bestellungen
ON kunden.kndnr = bestellungen.kndnr;
```

Ausgangsrelationen

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

Join

Bestellungen

BstID	KndNr	Gesamt	Datum
987	456	80€	1.1.2012
876	345	320€	7.3.2011
765	234	120€	1.8.2011

```
SELECT
kunden.name,
kunden.vorname,
bestellungen.gesamt
FROM kunden
INNER JOIN
bestellungen
ON kunden.kndnr =
bestellungen.kndnr;
```

Ergebnis

Name	Vorname	Gesamt
Boehrs	Vera	120€
Dinkel	Ulrike	320€
Dinkels	Thomas	80€

Auswählen von Tupeln mit SQL

Linker äußerer Verbund (Left Outer Join) in SQL (Bsp. 5):

– Alle *Lieferadressen* mit *Straße, Ort* und *ggf. vorhandene Bestellungen* mit *ID, Datum*

– SQL:

```
SELECT LAdr.StraßeNr, LAdr.PlzOrt, Bestell.BstId, Bestell.Datum
FROM LAdr LEFT JOIN Bestell
ON LAdr.AdrNr = Bestell.AdrNr;
```

Ausgangsrelationen

Linke Tabelle

LAdr	AdrNr	StraßeNr	PlzOrt
	123	Bergstr 7	14476 Potsdam
	234	Badstr 1a	12345 Berlin
	345	Schloßstr 8	12456 Berlin

Bestell	BstID	AdrNr	Liefern	Gesamt	Datum
	987	234	J	80€	1.1.2012
	876		N	320€	7.3.2011
	765	345	J	120€	1.8.2011

Ergebnisrelation

StraßeNr	PlzOrt	BstID	Datum
Bergstr 7	14476 Potsdam		
Badstr 1a	12345 Berlin	987	1.1.2012
Schloßstr 8	12456 Berlin	765	1.8.2011

Auswählen von Tupeln mit SQL

Rechter äußerer Verbund (Right Outer Join) in SQL (Bsp. 5):

– Alle *Bestellungen* mit *ID, Datum* und *ggf. vorhandene Lieferadressen* mit *Straße, Ort*

– SQL:

```
SELECT LAdr.StraßeNr, LAdr.PlzOrt, Bestell.BstId, Bestell.Datum
FROM LAdr RIGHT JOIN Bestell
ON LAdr.AdrNr = Bestell.AdrNr;
```

Rechte Tabelle

Ausgangsrelationen

LAdr	AdrNr	StraßeNr	PlzOrt
	123	Bergstr 7	14476 Potsdam
	234	Badstr 1a	12345 Berlin
	345	Schloßstr 8	12456 Berlin

Bestell	BstID	AdrNr	Liefern	Gesamt	Datum
	987	234	J	80€	1.1.2012
	876		N	320€	7.3.2011
	765	345	J	120€	1.8.2011

Ergebnisrelation

StraßeNr	PlzOrt	BstID	Datum
Badstr 1a	12345 Berlin	987	1.1.2012
		876	7.3.2011
Schloßstr 8	12456 Berlin	765	1.8.2011

Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung

– Differenz

– Schnitt

Kartesisches Produkt

Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)
  - Mengen A und B werden in einer Menge vereinigt
  - Voraussetzung: Gleiche Spalten (und Datentypen)

– Differenz

– Schnitt

Kartesisches Produkt

Mitarbeiter	Name	Vorname
	Wurst	Hans
	Wurst	Ida
	Vetter	Jan

Kunden	Name	Vorname
	Albers	Willi
	Boehrs	Vera
	Dinkel	Ulrike

```
SELECT * FROM mitarbeiter
UNION
SELECT * FROM kunden;
```

Name	Vorname
Albers	Willi
Boehrs	Vera
Dinkel	Ulrike
Wurst	Hans
Wurst	Ida
Vetter	Jan

Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

– Differenz (EXCEPT)

- Eliminieren der Tupel in A, die auch in B sind
- Voraussetzung: Gleiche Spalten (und Datentypen)

– Schnitt

Kartesisches Produkt

Personen	Name	Vorname
	Albers	Willi
	Boehrs	Vera
	Dinkel	Ulrike
	Wurst	Hans
	Wurst	Ida
	Vetter	Jan

Kunden	Name	Vorname
	Albers	Willi
	Boehrs	Vera
	Dinkel	Ulrike

```
SELECT * FROM personen
EXCEPT
SELECT * FROM kunden;
```

Name	Vorname
Wurst	Hans
Wurst	Ida
Vetter	Jan

Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

– Differenz (EXCEPT)

– Schnitt (INTERSECT)

- Gemeinsame Tupel in A und B
- Voraussetzung: Gleiche Spalten (und Datentypen)

Kartesisches Produkt

Kunden	Name	Vorname
	Wurst	Ida
	Vetter	Jan
	Dinkel	Ulrike
	Wurst	Hans

Mitarbeiter	Name	Vorname
	Albers	Willi
	Boehrs	Vera
	Dinkel	Ulrike
	Wurst	Hans

```
SELECT * FROM kunden
INTERSECT
SELECT * FROM mitarbeiter;
```

Name	Vorname
Dinkel	Ulrike
Wurst	Hans

Auswählen von Tupeln mit SQL

Mengenoperationen

- Vereinigung (UNION)
- Differenz (EXCEPT)
- Schnitt (INTERSECT)

Kartesisches Produkt

- Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```

R1	S1	S2
	A	B
	B	C
	C	D
	D	E

```
SELECT *
FROM
R1, R2;
```

R2	S3	S4
	1	2
	2	3
	3	4

S1	S2	S3	S4
A	B	1	2
A	B	2	3
A	B	3	4
B	C	1	2
B	C	2	3
B	C	3	4
C	D	1	2
C	D	2	3
C	D	3	4
D	E	1	2
D	E	2	3
D	E	3	4

Auswählen von Tupeln mit SQL

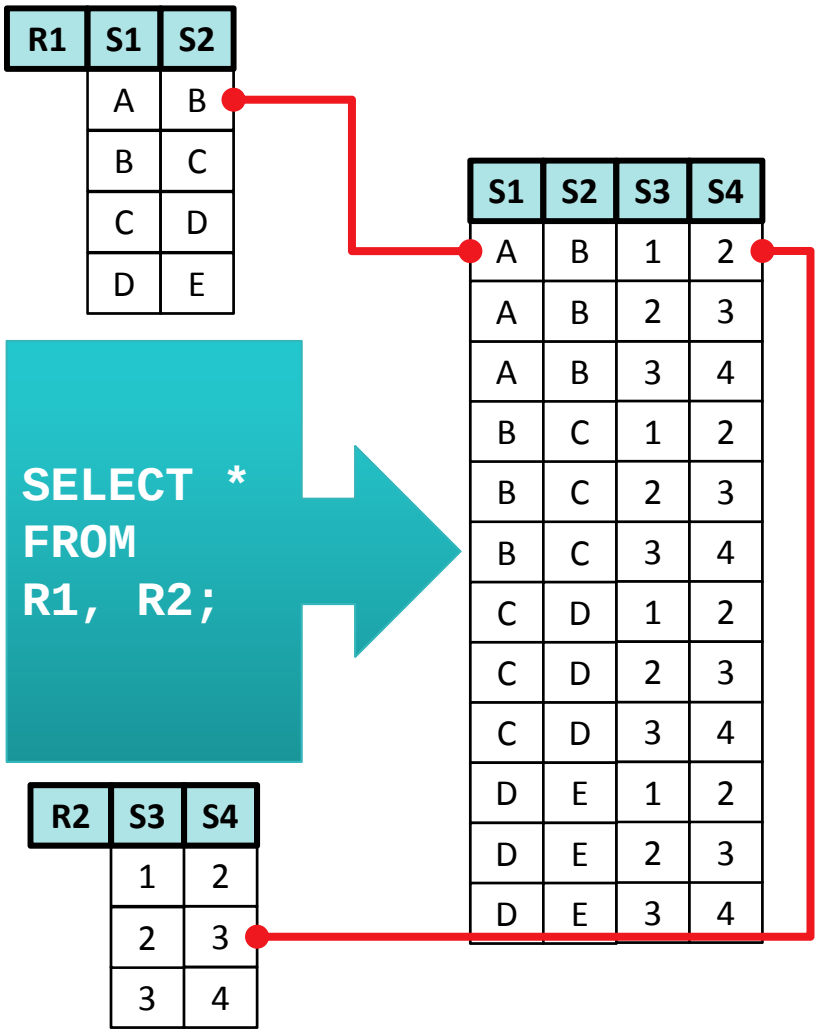
Mengenoperationen

- Vereinigung (UNION)
- Differenz (EXCEPT)
- Schnitt (INTERSECT)

Kartesisches Produkt

- Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

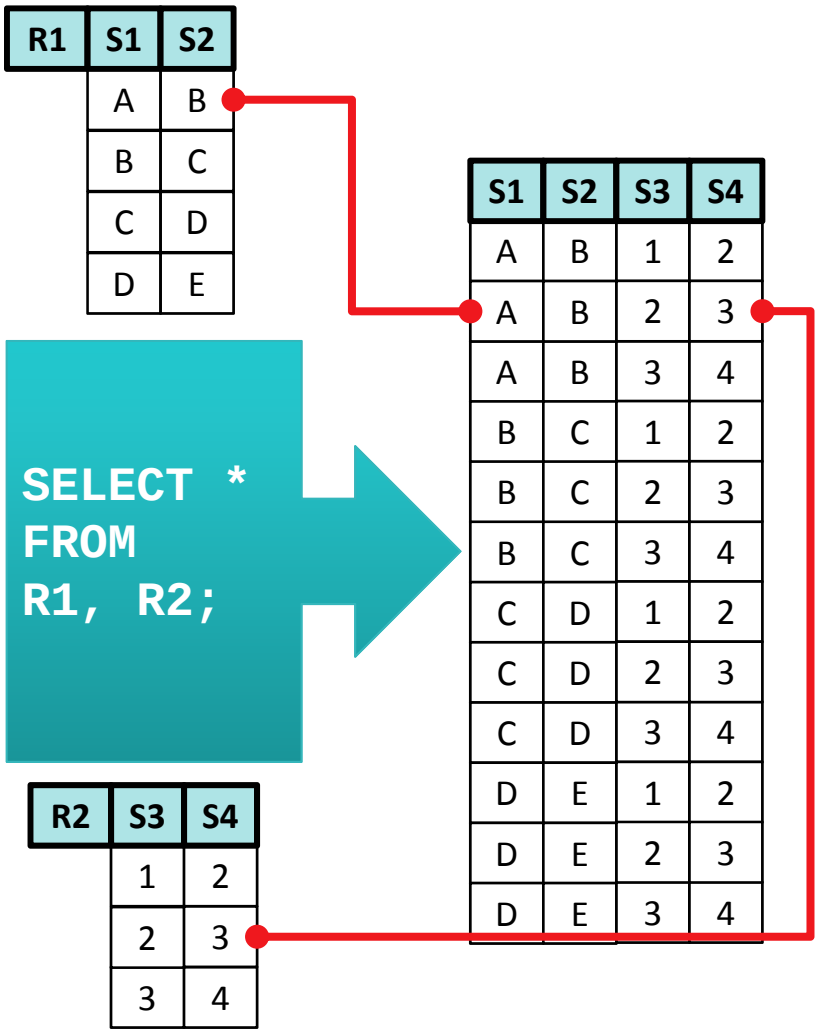
Mengenoperationen

- Vereinigung (UNION)
- Differenz (EXCEPT)
- Schnitt (INTERSECT)

Kartesisches Produkt

- Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

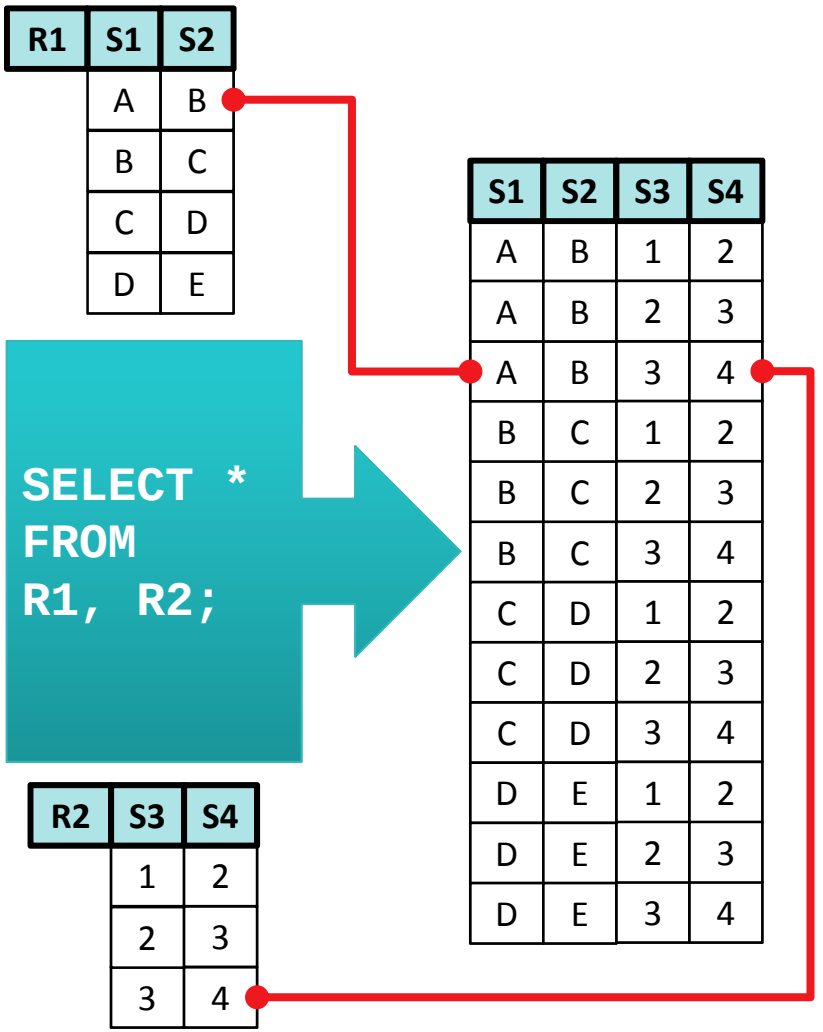
– Differenz (EXCEPT)

– Schnitt (INTERSECT)

Kartesisches Produkt

– Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

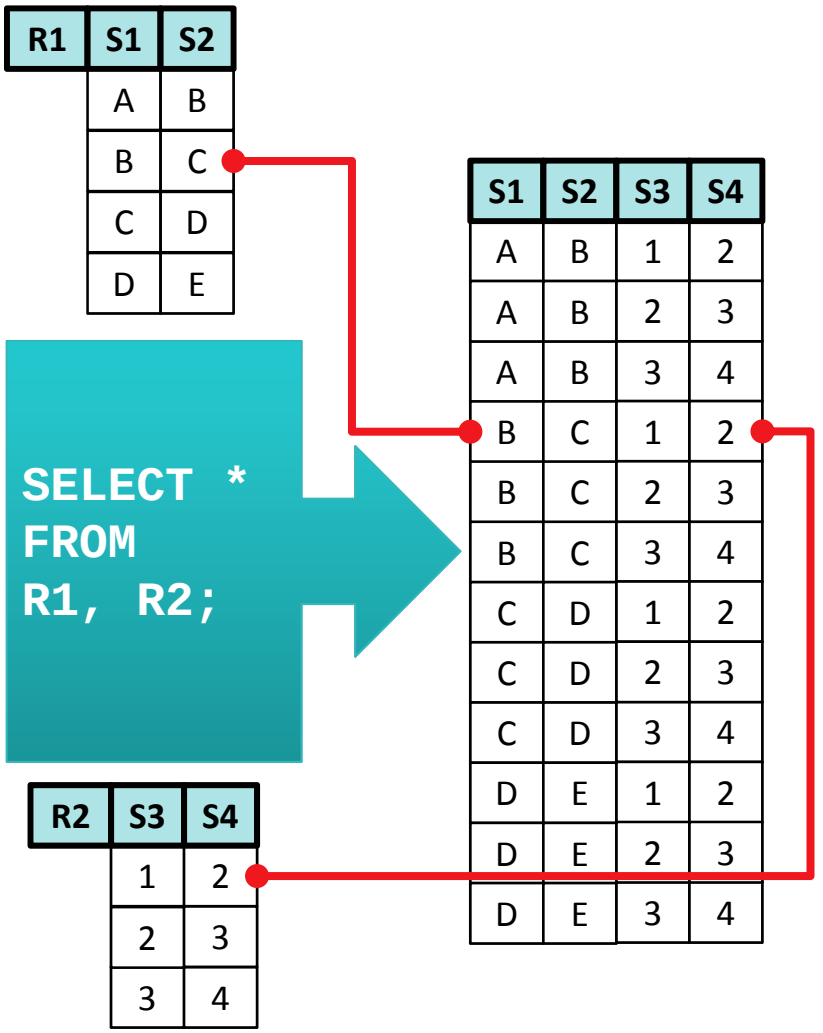
– Differenz (EXCEPT)

– Schnitt (INTERSECT)

Kartesisches Produkt

– Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

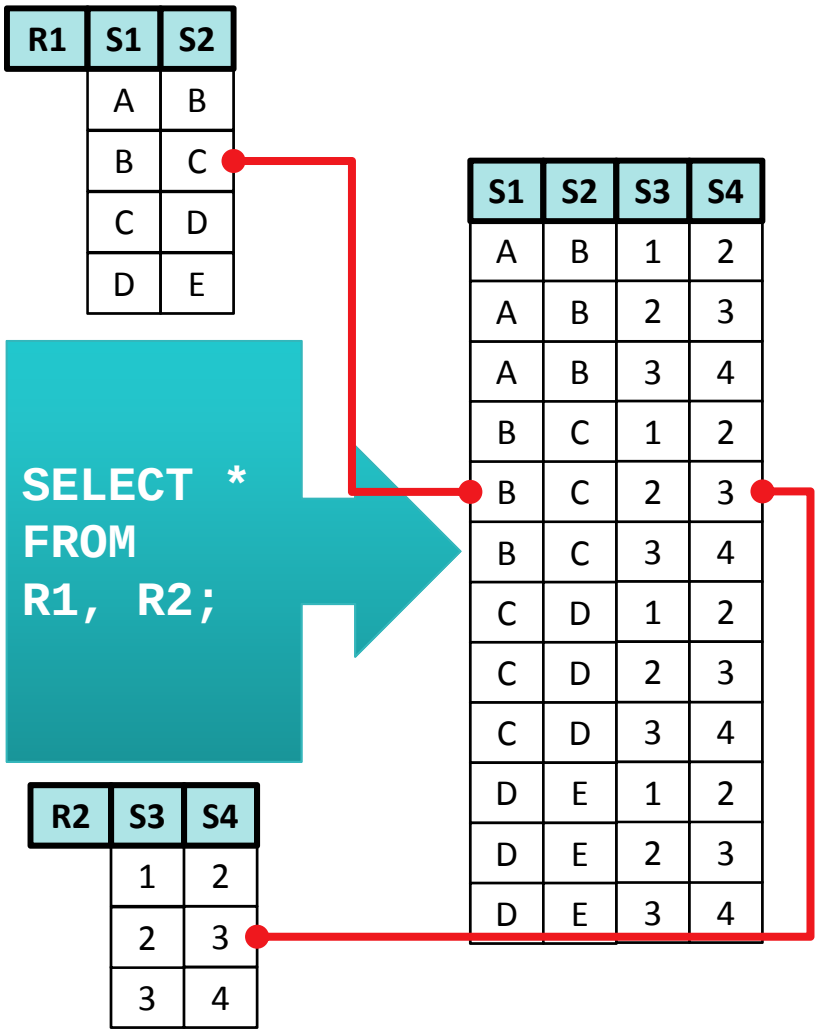
– Differenz (EXCEPT)

– Schnitt (INTERSECT)

Kartesisches Produkt

– Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

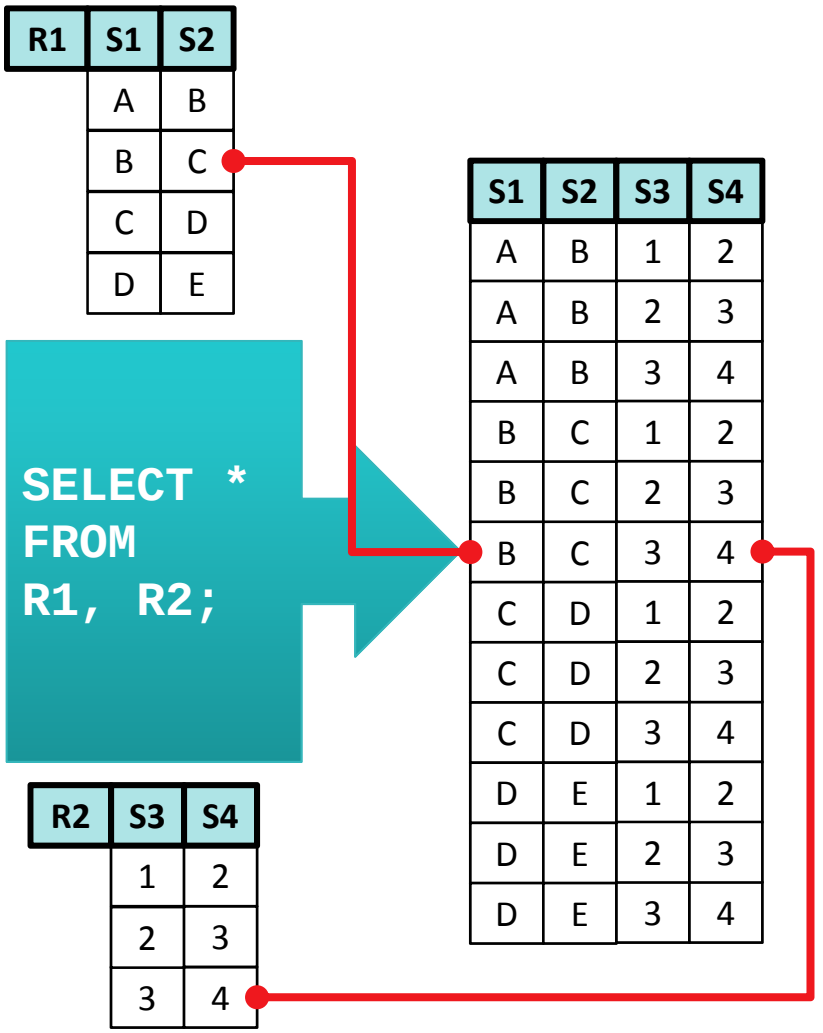
– Differenz (EXCEPT)

– Schnitt (INTERSECT)

Kartesisches Produkt

– Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

– Differenz (EXCEPT)

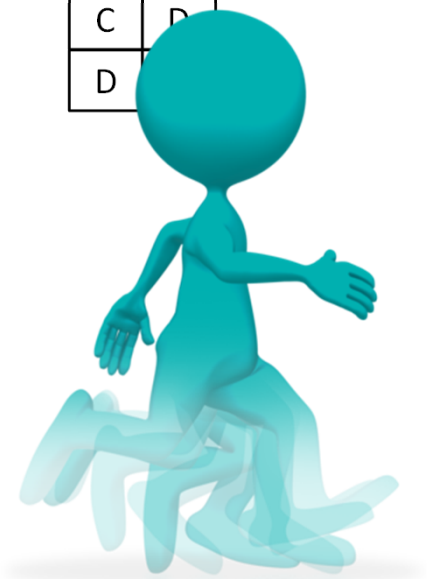
– Schnitt (INTERSECT)

Kartesisches Produkt

– Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```

R1	S1	S2
	A	B
	B	C
	C	D
	D	



R2	S3	S4
	1	2
	2	3
	3	4

S1	S2	S3	S4
A	B	1	2
A	B	2	3
A	B	3	4
B	C	1	2
B	C	2	3
B	C	3	4
C	D	1	2
C	D	2	3
C	D	3	4
D	E	1	2
D	E	2	3
D	E	3	4

Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung (UNION)

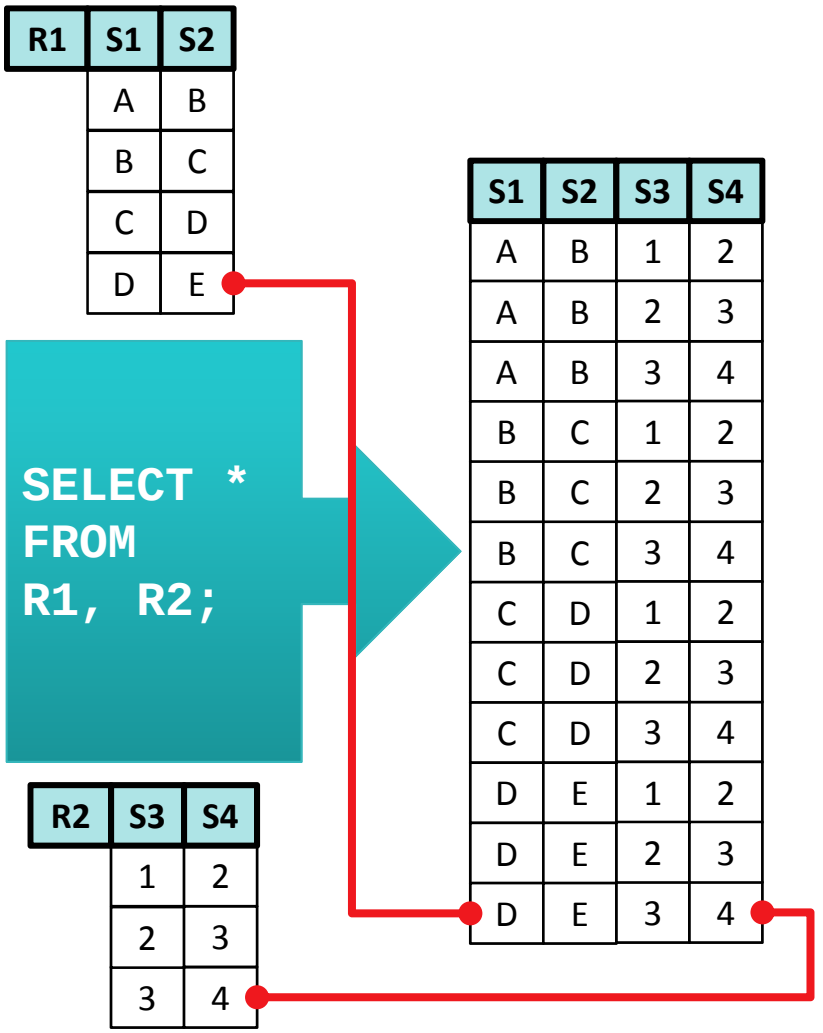
– Differenz (EXCEPT)

– Schnitt (INTERSECT)

Kartesisches Produkt

– Kombination jedes Elements von Menge A mit jedem Element von B

```
SELECT *
FROM R1, R2;
```



Auswählen von Tupeln mit SQL

Mengenoperationen

– Vereinigung

– Differenz

– Schnitt

Kartesisches Produkt

Weiterführendes SQL zum Auswählen von Tupeln

SELECT ... WHERE <Bedingung>

- Bedingungen mit NOT und verknüpft mit AND und OR möglich
- Bedingungen können Prüfung umfassen
  - gleich =
  - ungleich <> bzw. !=
  - größer als > bzw. kleiner als <
  - größer gleich >= bzw. kleiner gleich <=
  - nicht größer als !> bzw. nicht kleiner als !<
- Beispiele
  - ```
    SELECT name FROM kunden
      WHERE plz >= 12000 AND plz <= 15000
            AND name='Thomas';
    ```
    ```

Weiterführendes SQL zum Auswählen von Tupeln

SELECT ... WHERE <Bedingung>, weitere Operatoren für die Bedingung

– IN-Operator

- ```
  SELECT * FROM kunden
    WHERE name IN ('Wurst', 'Vetter', 'Dinkel')
  ```

– BETWEEN-Operator

- ```
  SELECT * FROM bestellungen
    WHERE gesamtpreis BETWEEN 500 AND 1000;
  ```

– LIKE-Operator

- ```
  SELECT * FROM kunden
    WHERE name LIKE 'Dink';
  ```

– NULL-Operator

- ```
  SELECT * FROM kunden
    WHERE email IS NOT NULL;
  ```

Weiterführendes SQL zum Auswählen von Tupeln

Alias für Tabellen verwenden

```
SELECT a.*, u.* FROM artikel AS a, umsatz AS u
```

Sortieren der Ergebnismenge (absteigend, engl. descending)

```
SELECT b.gesamt, b.datum FROM bestellungen as b
ORDER BY b.gesamt DESC;
```

Berechnungen in Abfragen und Sortierung (aufsteigend, engl. ascending)

```
SELECT einzelpreis, stueckzahl, einzelpreis *
stueckzahl AS gesamt
FROM bestellpositionen
ORDER BY gesamt ASC;
```

Entfernen doppelter Einträge (z.B. gleiche Names-Kombinationen bei Kunden)

```
SELECT DISTINCT name, vorname FROM kunden;
```

Überblick über die SELECT-Anweisung[1)]

```
SELECT[DISTINCT]
      <Spaltenname>
      <Berechnung>
      <Konstante>
      <einer der obigen Ausdrücke> As Alias
      [, weitere der obigen Ausdrücke]
FROM <Ausdruck, der Tabelle zurückgibt> As Alias
[WHERE ...]
[GROUP BY ...]
[HAVING ...]
[UNION [ALL]]
[Weitere SELECT-Anweisung, welche dieselbe Zahl
von
      Spalten und Datentypen liefert]
[ORDER BY [Order-By-Ausdruck] ASC | DESC]]
      [, weitere Sortierungen]
```

1) nach [1]

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

- vorhandene Tupel werden gelesen
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

- neue Tupel werden zur Relation hinzugefügt

Ändern

- vorhandene Tupel werden geändert
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

- vorhandene Tupel werden gelöscht
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

- vorhandene Tupel werden gelesen
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

- neue Tupel werden zur Relation hinzugefügt

Ändern

- vorhandene Tupel werden geändert
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

- vorhandene Tupel werden gelöscht
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Einfügen von Tupeln mit SQL

SQL-Syntax ähnlich der natürlichen Sprache (in Englisch), d.h. es kann bspw. formuliert werden

- "Füge in Tabelle Kunden die Werte 345, 'Simon', 'Jakob' ein."
- "Füge in Tabelle Kunden für die Spalte Vorname, Name, Kundenummer die Werte 'Simon', 'Jakob', 345 ein."

Einfügen von Tupeln mit SQL

Einfügen in vorgegebener Spaltenreihenfolge (Bsp. 1):

- Deutsch: "Füge in Tabelle Kunden die Werte 345, 'Simon', 'Jakob' ein."
- Englisch: "Insert into table *Kunden* values 345, 'Simon', 'Jakob'."
- SQL:

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

Ausgangsrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera

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

Ergebnisrelation

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera
	345	Simon	Jakob

Einfügen von Tupeln mit SQL

Einfügen in abweichender Spaltenreihenfolge (Bsp. 1):

- Deutsch: "Füge in Tabelle Kunden für die Spalte Vorname, Name, Kundenummer die Werte 'Simon', 'Jakob', 345."
- Englisch: "Insert into table *Kunden* in columns Vorname, Name, Kundenummer values 'Simon', 'Jakob', 345."
- 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

Kunden	KndNr	Name	Vorname
	123	Albers	Willi
	234	Boehrs	Vera
	345	Jakob	Simon

Weiterführendes SQL zum Einfügen von Tuplen

Wenn Primärschlüssel auf Relation (als AutoWert) definiert, muss beim INSERT kein Wert für den Schlüssel angegeben werden

```
INSERT INTO personen (vorname, name)
VALUES ('Barbara', 'Beispiel');
```

Kombination von INSERT INTO mit dem Ergebnis einer Auswahl mittels SELECT

```
INSERT INTO personen (vorname, name)
SELECT vorname, name FROM kunden
WHERE ort='Berlin';
```

Überblick über die INSERT-Anweisung[1)]

Entweder

```
INSERT INTO
  <Tabellenname>(<Spaltenname> [,weitere Spalte])
VALUES
 (<Wert für die erste Spalte> [, weitere Werte])
```

Oder

```
INSERT INTO
  <Tabellenname>(<Spaltenname> [, weitere Spalte])
SELECT <erste Spalte> [, weitere Ausgaben]
     [FROM ...]
     [WHERE ...]
     [GROUP BY ...]
     [HAVING ...]
```

1) siehe [1]

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

– vorhandene Tupel werden gelesen

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

– neue Tupel werden zur Relation hinzugefügt

Ändern

– vorhandene Tupel werden geändert

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

– vorhandene Tupel werden gelöscht

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

– vorhandene Tupel werden gelesen

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

– neue Tupel werden zur Relation hinzugefügt

Ändern

– vorhandene Tupel werden geändert

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

– vorhandene Tupel werden gelöscht

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Ändern von Tupeln mit SQL

SQL-Syntax ähnlich der natürlichen Sprache (in Englisch), d.h. es kann bspw. formuliert werden

– "Ändere in der Tabelle Kunden den Wert der Spalte Name in 'Albers', wo die Kundennummer '234' ist."

Ändern von Tupeln mit SQL

Änderungen (Bsp. 1):

– Deutsch: "Ändere in der Tabelle *Kunden* den Wert der Spalte *Name* in 'Albers', wo die *Kundennummer* '234' ist."

– Englisch: "Update in table *Kunden* value of column *Name* to 'Albers' where *Kundennummer* is '234'."

– 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

Überblick über die UPDATE-Anweisung[1)]

```
UPDATE <Tabelle>
 SET <Name einer Spalte> = <Ausdruck aus Spalten,
      Konstanten, Funktionen>
      [, weitere Spaltennamen = Ausdruck]
 WHERE <Bedingung>
```

1) siehe [1]

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

– vorhandene Tupel werden gelesen

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

– neue Tupel werden zur Relation hinzugefügt

Ändern

– vorhandene Tupel werden geändert

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

– vorhandene Tupel werden gelöscht

– ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

- vorhandene Tupel werden gelesen
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

- neue Tupel werden zur Relation hinzugefügt

Ändern

- vorhandene Tupel werden geändert
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

- vorhandene Tupel werden gelöscht
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen von Tupeln mit SQL

SQL-Syntax ähnlich der natürlichen Sprache (in Englisch), d.h. es kann bspw. formuliert werden

– "Lösche alles aus der Tabelle Kunden."

– "Lösche aus der Tabelle Kunden wo der Name 'Albers' ist."

Ändern von Tupeln mit SQL

Alle Zeilen löschen (Bsp. 1):

– Deutsch: "Lösche alles aus der Tabelle *Kunden*."

– Englisch: "Delete all from table *Kunden*."

– SQL: **`DELETE FROM kunden;`**

Ausgangsrelation

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

```
DELETE FROM kunden;
```

Ergebnisrelation

Kunden	KndNr	Name	Vorname

Ändern von Tupeln mit SQL

Alle Zeilen, die einer Bedingung genügen löschen (Bsp. 2):

- Deutsch: "Lösche aus der Tabelle *Kunden* wo der *Name* 'Albers' ist."
- Englisch: "Delete from table *Kunden* where *Name* is 'Albers'."
- 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

Überblick über die DELETE-Anweisung[1)]

```
DELETE FROM <Tabelle>
[WHERE <Bedingung>];
```

1) siehe [1]

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

- vorhandene Tupel werden gelesen
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

- neue Tupel werden zur Relation hinzugefügt

Ändern

- vorhandene Tupel werden geändert
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

- vorhandene Tupel werden gelöscht
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Arbeiten mit Relationen und Tupeln per SQL

Auswählen

- vorhandene Tupel werden gelesen
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel bereitgestellt werden

Einfügen

- neue Tupel werden zur Relation hinzugefügt

Ändern

- vorhandene Tupel werden geändert
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Löschen

- vorhandene Tupel werden gelöscht
- ggf. nach bestimmten Kriterien festgelegt, welche Tupel zu ändern sind

Inhalt

Ziel und Einordnung

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel
- Integritätsregeln
- Operationen auf Relationen und Tupeln

Grundlagen des Relationales Datenmodell (Teil 3) - SQL

- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

Arbeiten mit dem Relationalen Modell (Teil 3) - SQL in MS Access

- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

Zusammenfassung

SQL (Structured Query Language) als Datenbanksprache

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): dient zum Erzeugen, Verändern und Löschen der Strukturen, die für die Speicherung der Daten benutzt werden (z.B. Tabellen, Spalten)
- Data Control Language (DCL): dient zum Einrichten, Festlegen und Entziehen von Zugriffsrechten für Benutzer und Gruppen auf den Strukturen und auf Aktionsmöglichkeiten der Datenbank

Thema von LE 11

Thema von LE 11

Inhalt

Ziel und Einordnung

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel
- Integritätsregeln
- Operationen auf Relationen und Tupeln

Grundlagen des Relationales Datenmodell (Teil 3) - SQL

- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

Arbeiten mit dem Relationalen Modell (Teil 3) - SQL in MS Access

- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

SQL SELECT-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

- SELECT
- INSERT
- UPDATE
- DELETE

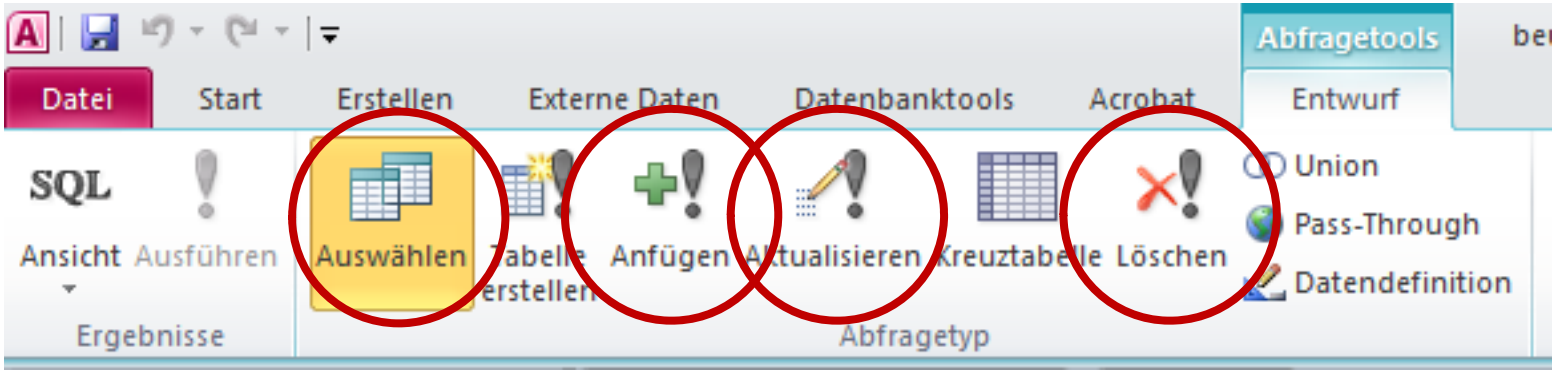
SQL SELECT-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

Grafische Abfragen und SQL-Abfragen

MS Access stellt dem Sprachumfang von SQL vergleichbare grafische Abfragen bereit

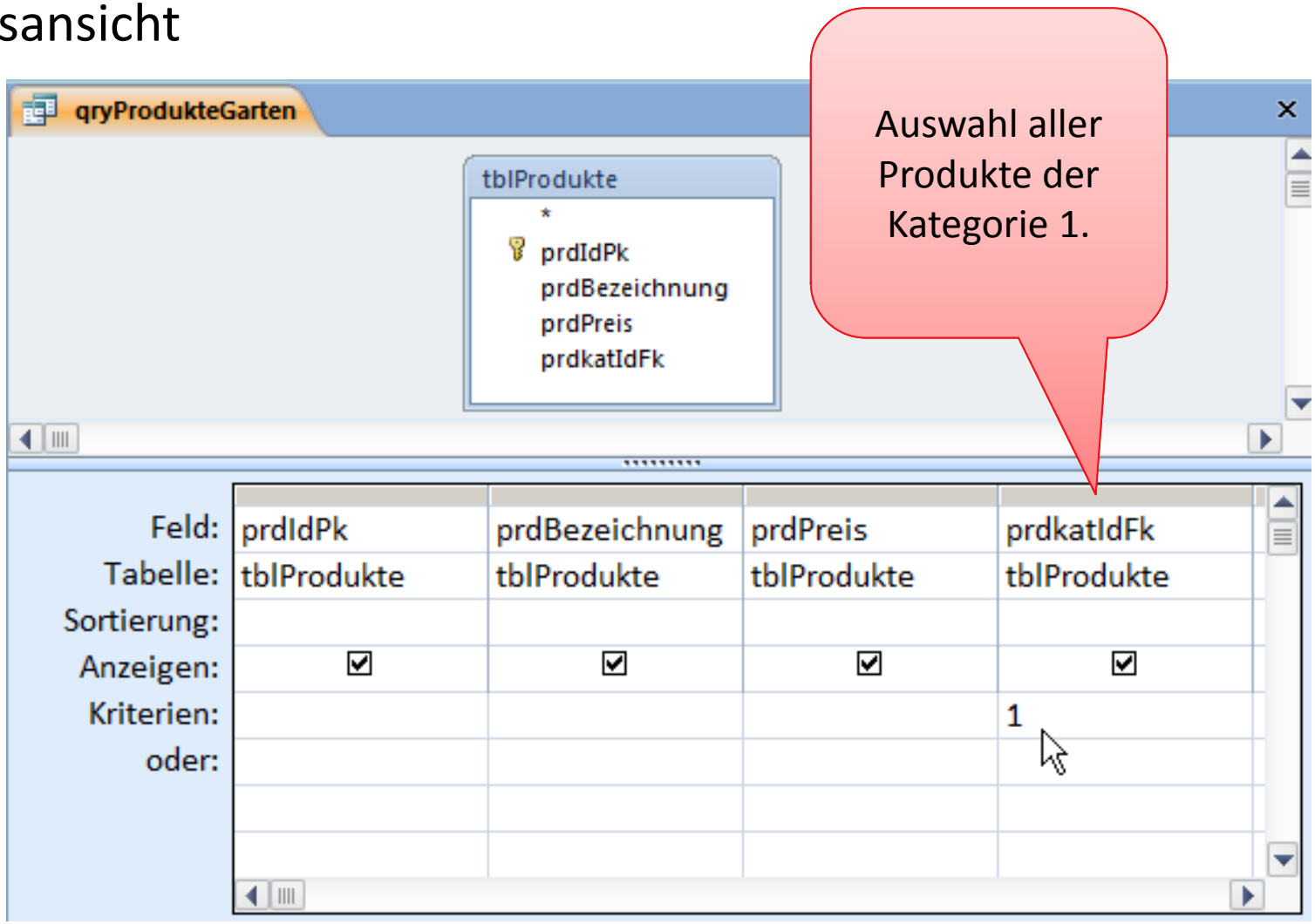
– Auswahlabfrage für Restriktion, Projektion und Join
  - entspricht: SELECT
  - vgl. letzte LE
– Anfügeabfrage: entspricht INSERT
– Aktualisierungsabfrage: entspricht UPDATE
– Löschabfrage: entspricht DELETE



SQL-Ansicht grafischer Abfragen

Jede Abfrage kann in mehreren Ansicht dargestellt werden

– Entwurfsansicht



SQL-Ansicht grafischer Abfragen

Jede Abfrage kann in mehreren Ansicht dargestellt werden

– Entwurfsansicht

– Datenblattansicht

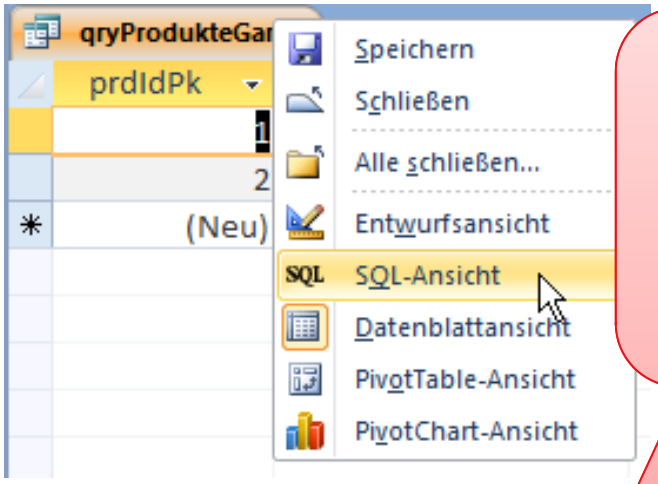
qryProdukteGarten

prdIdPk	prdBezeichnung	prdPreis	prdkatIdFk
1	Harke "Boris"	5,99 €	1
2	Rasenbesen "Thomas"	6,99 €	1
(Neu)			

SQL-Ansicht grafischer Abfragen

Jede Abfrage kann in mehreren Ansicht dargestellt werden

- Entwurfsansicht
- Datenblattansicht
- SQL-Ansicht



Auswahl aller Produkte der Kategorie 1.

qryProdukteGarten

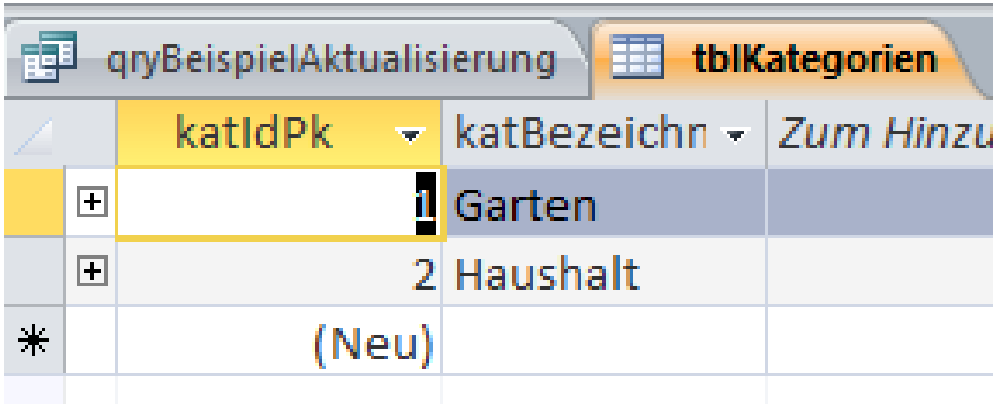
```
SELECT tblProdukte.prdIdPk, tblProdukte.prdBezeichnung,
tblProdukte.prdPreis, tblProdukte.prdkatIdFk
FROM tblProdukte
WHERE (((tblProdukte.prdkatIdFk)=1));
```

SQL-Ansicht grafischer Abfragen

SQL-Ansicht steht auch für Anfüge-, Aktualisierungs- und Löschabfragen zur Verfügung

Beispiel

– Aktualisierung der Kategorien-bezeichnung "Haushalt" in "Haushalt/Balkon"

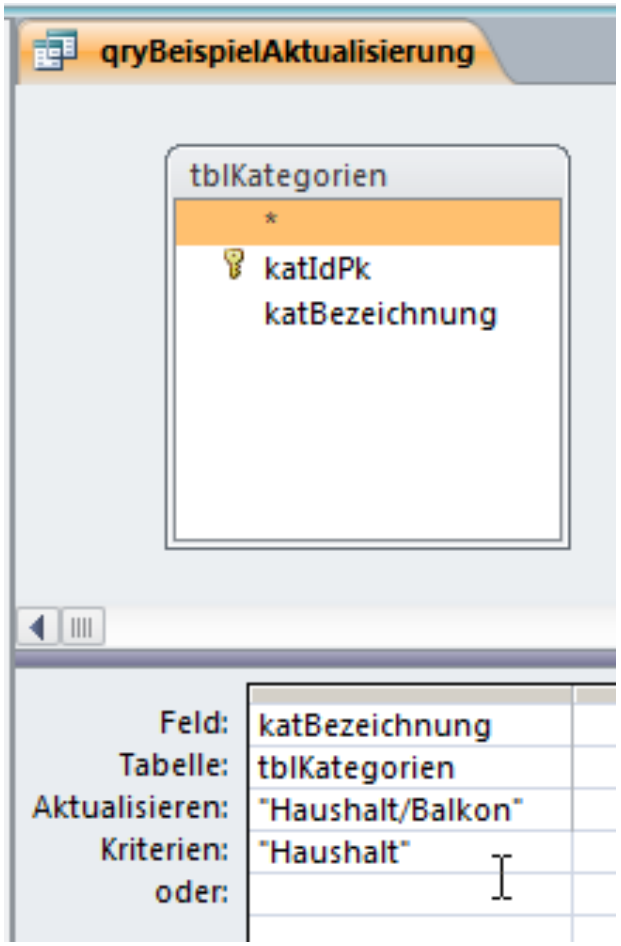


SQL-Ansicht grafischer Abfragen

SQL-Ansicht steht auch für Anfüge-, Aktualisierungs- und Löschabfragen zur Verfügung

Beispiel

– Aktualisierung der Kategorien-bezeichnung "Haushalt" in "Haushalt/Balkon"

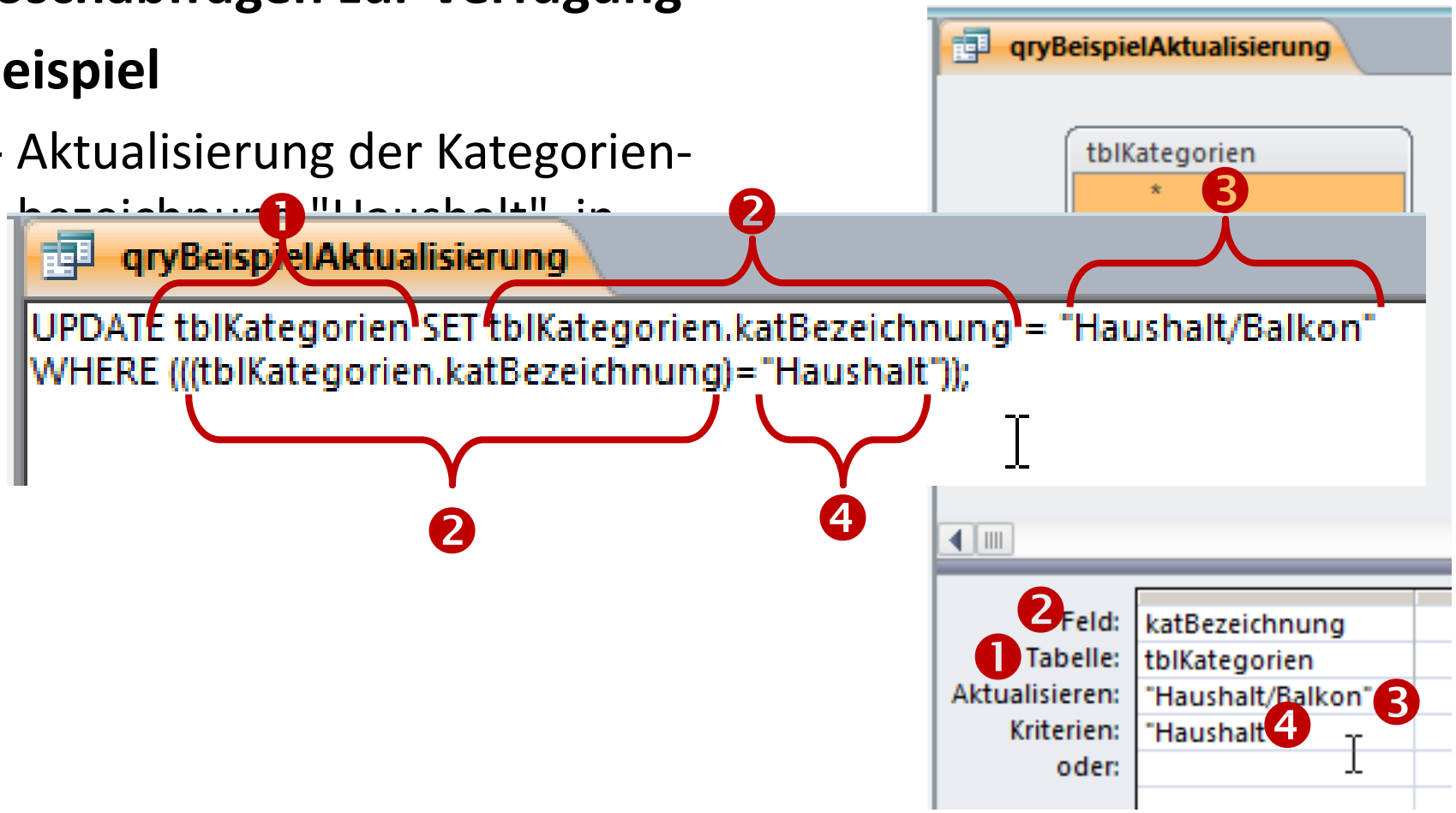


SQL-Ansicht grafischer Abfragen

SQL-Ansicht steht auch für Anfüge-, Aktualisierungs- und Löschabfragen zur Verfügung

Beispiel

– Aktualisierung der Kategorien-

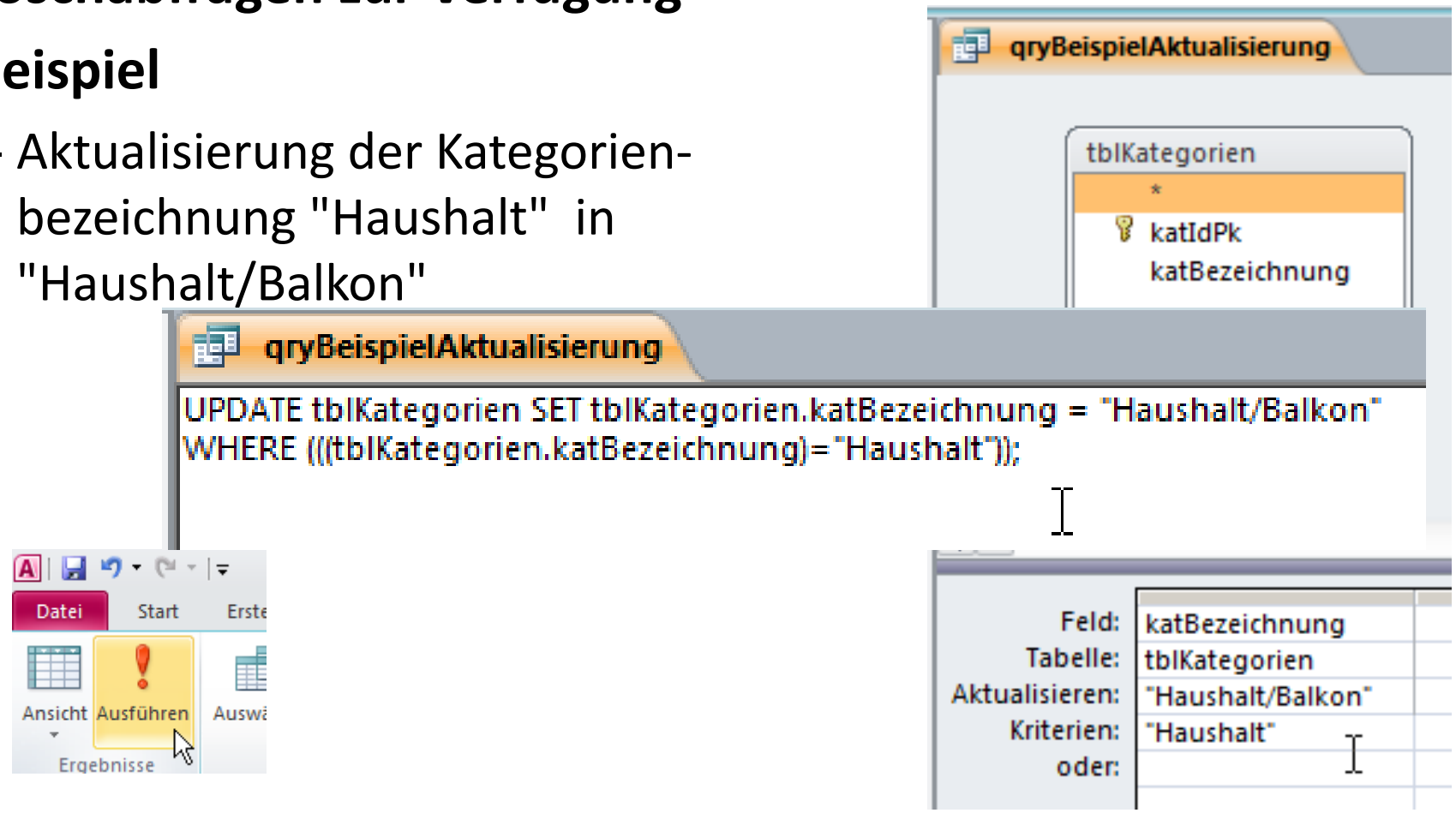


SQL-Ansicht grafischer Abfragen

SQL-Ansicht steht auch für Anfüge-, Aktualisierungs- und Löschabfragen zur Verfügung

Beispiel

– Aktualisierung der Kategorien-bezeichnung "Haushalt" in "Haushalt/Balkon"

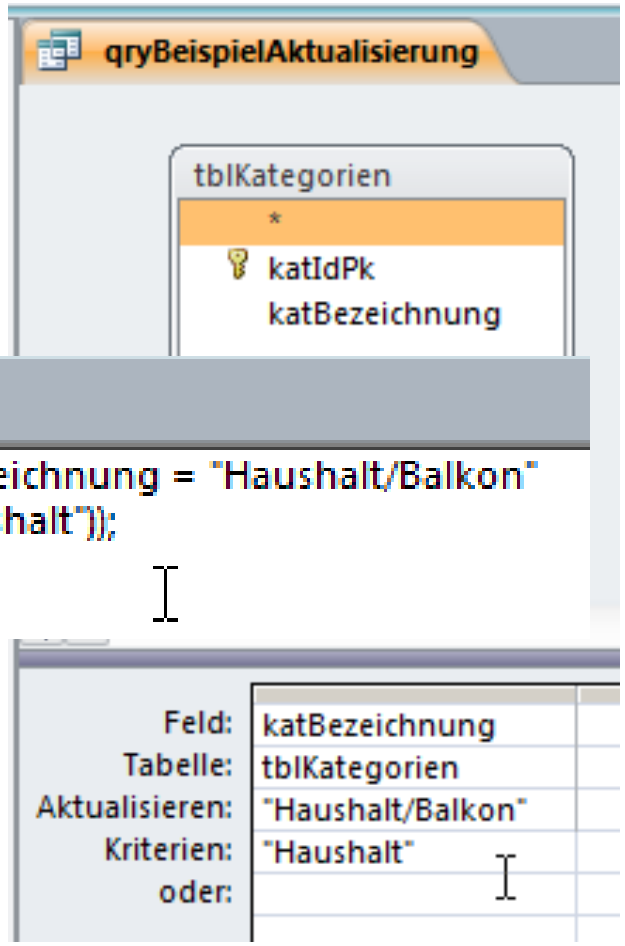


SQL-Ansicht grafischer Abfragen

SQL-Ansicht steht auch für Anfüge-, Aktualisierungs- und Löschabfragen zur Verfügung

Beispiel

– Aktualisierung der Kategorien-bezeichnung "Haushalt" in "Haushalt/Balkon"

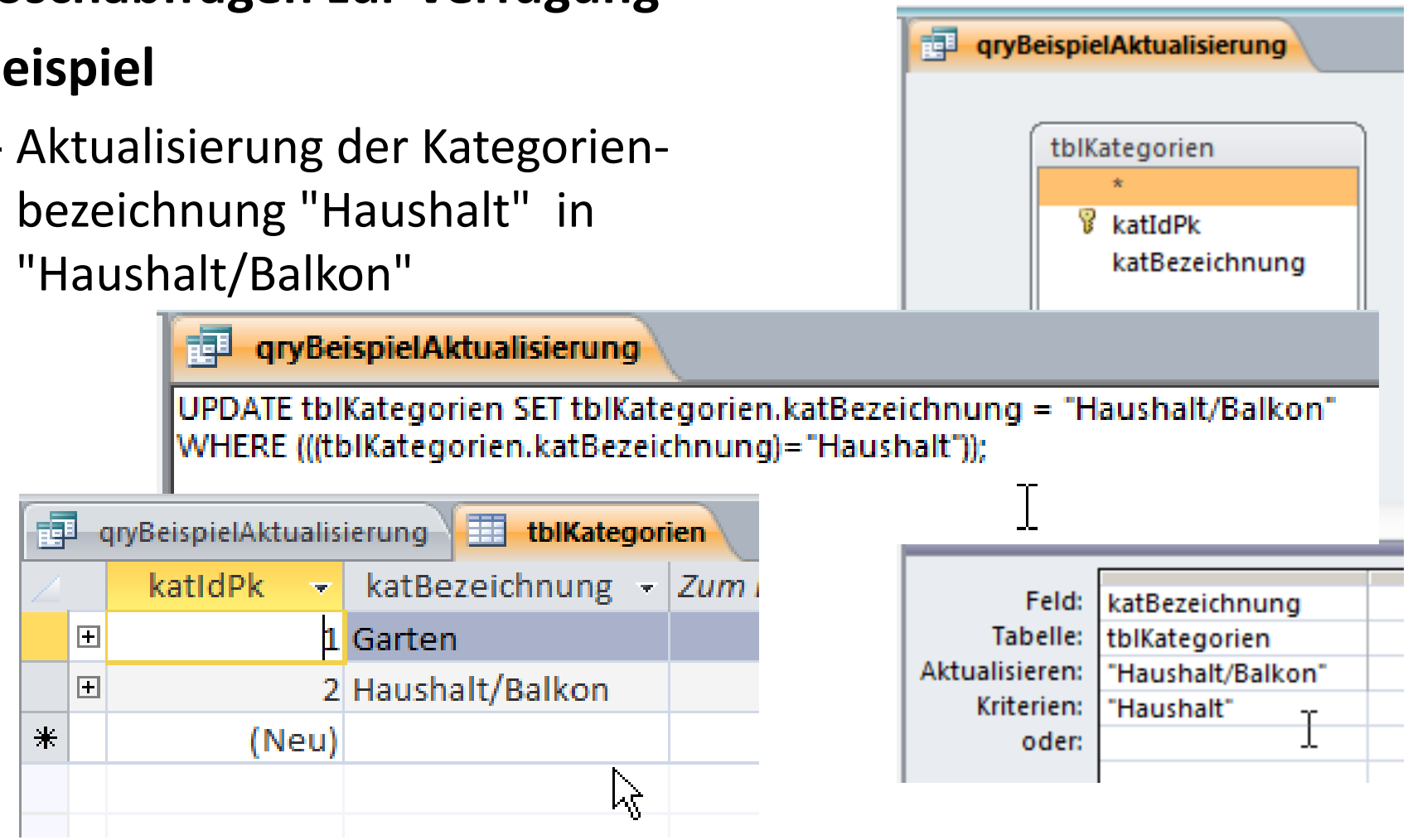


SQL-Ansicht grafischer Abfragen

SQL-Ansicht steht auch für Anfüge-, Aktualisierungs- und Löschabfragen zur Verfügung

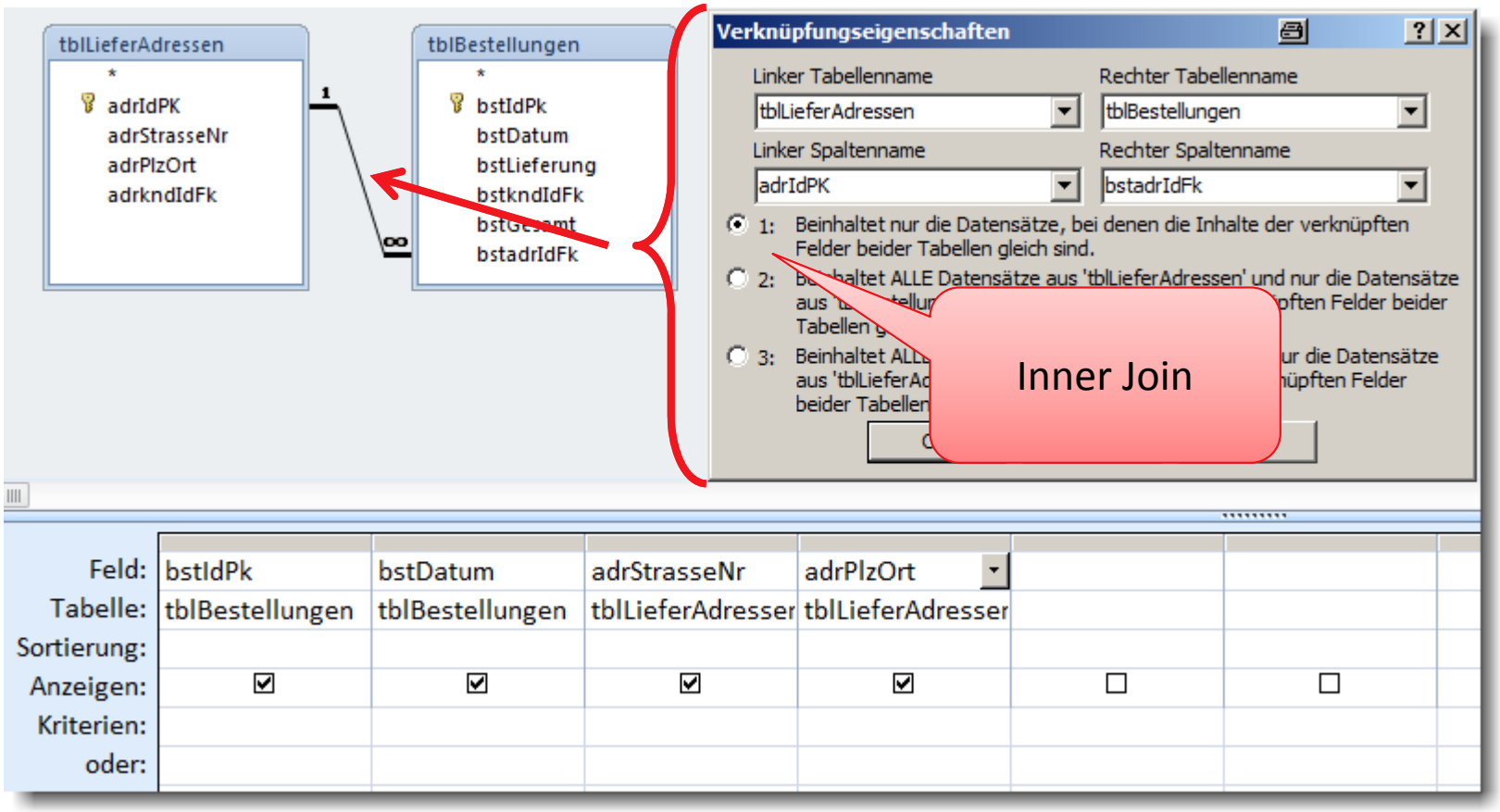
Beispiel

– Aktualisierung der Kategorien-bezeichnung "Haushalt" in "Haushalt/Balkon"



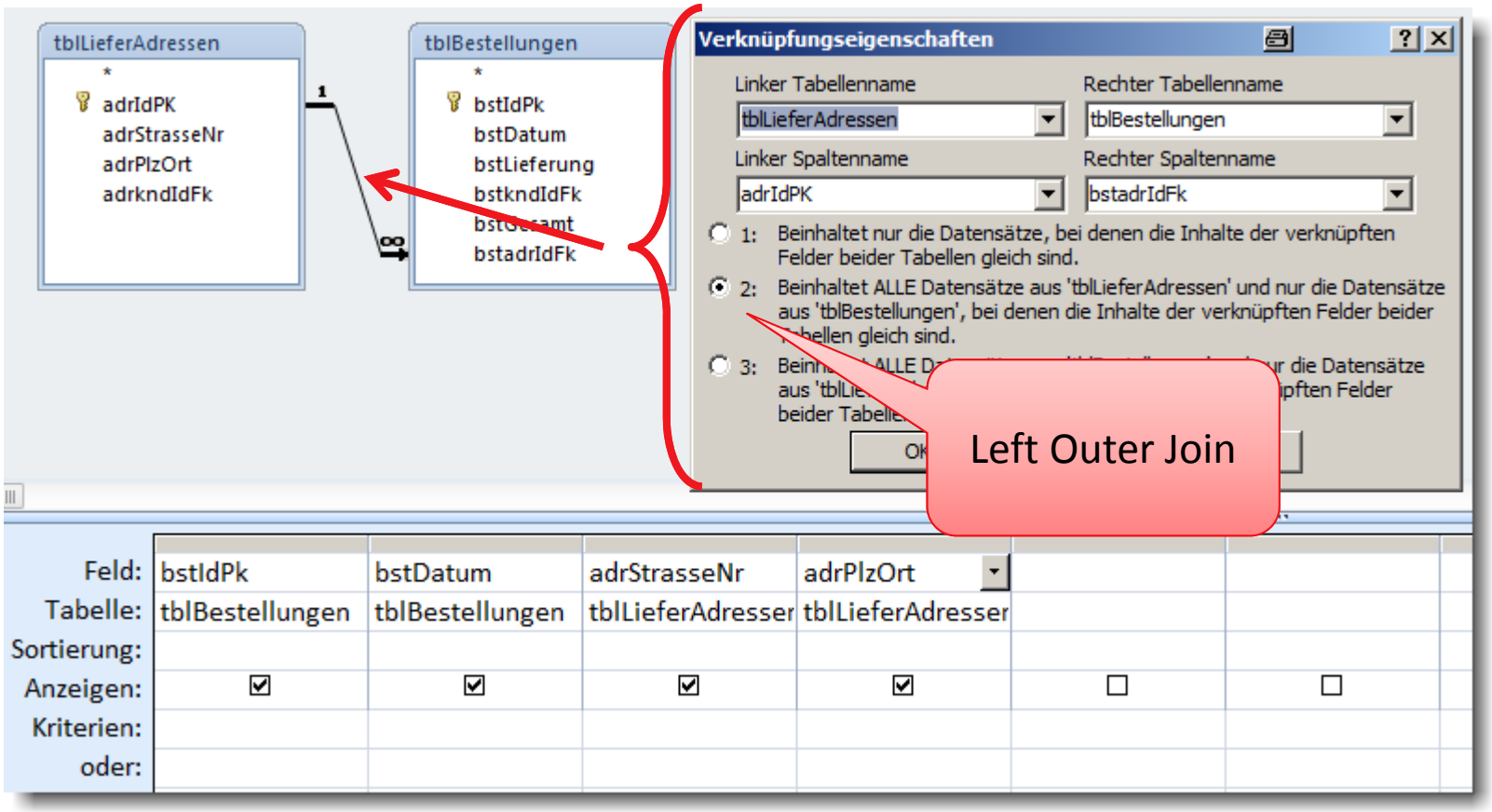
SQL-Ansicht grafischer Abfragen

Verbund (Join) in der Entwurfs- und SQL-Ansicht von Auswahlabfragen



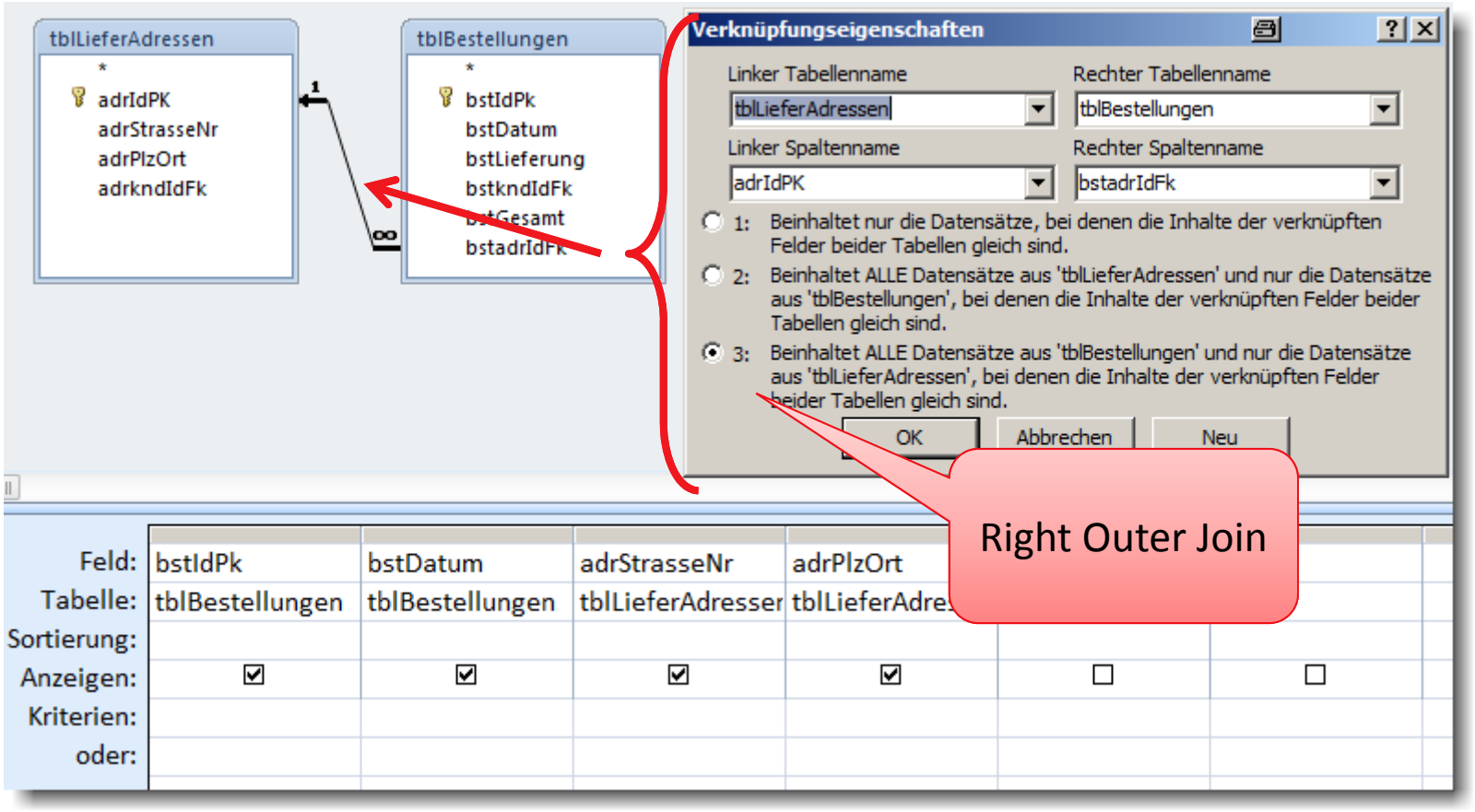
SQL-Ansicht grafischer Abfragen

Verbund (Join) in der Entwurfs- und SQL-Ansicht von Auswahlabfragen



SQL-Ansicht grafischer Abfragen

Verbund (Join) in der Entwurfs- und SQL-Ansicht von Auswahlabfragen

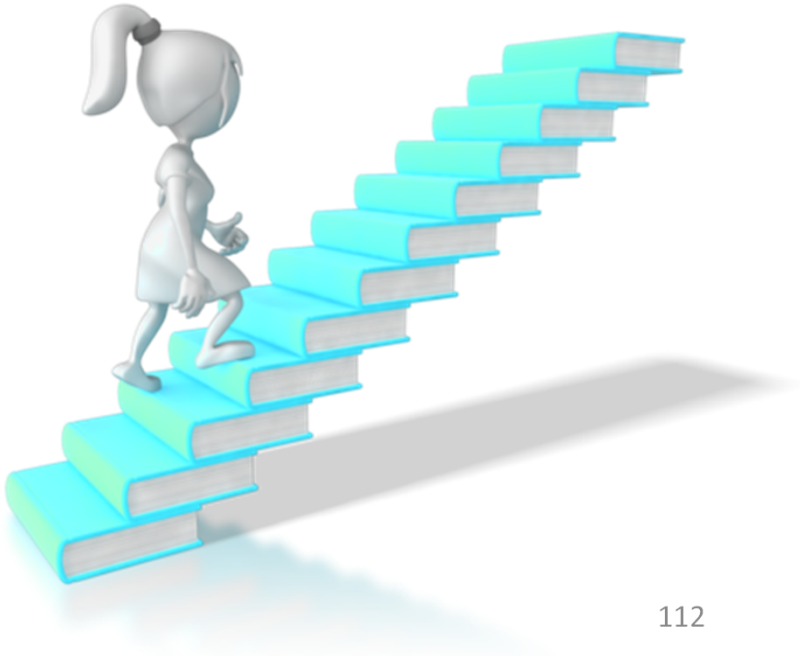


SQL-Ansicht grafischer Abfragen

Verzicht auf grafische Abfragen, jetzt kommt SQL!

Generelle Vorgehensweise

1. Erstellen eines neuen Abfrageentwurfs
2. Schließen des Dialogs zum Hinzufügen von Tabellen
3. Sofortiges Wechseln in SQL-Ansicht
4. Erstellen der SQL-Anweisung bzw. SQL-Abfrage
5. Speichern als Abfrage (Präfix: qry)
6. Ausführen der Abfrage
   - Klick auf Symbol für Ausführen (bei SQL-Anweisungen)
   - Wechsel in Datenblattansicht (bei SQL-Abfragen)



SQL-Ansicht grafischer Abfragen: Demo D06.03

D06.03

– Erstellen von drei SQL-Anweisungen
  1. zum Hinzufügen eines neuen Benutzers
  2. zum Ändern des Passworts des hinzufügten Benutzers
  3. zum Löschen des geänderten Benutzers

– Vorgehensweise
  - Erstellen eines neuen Abfrageentwurfs
  - Schließen des Dialogs zum Hinzufügen von Tabellen
  - Sofortiges Wechseln in SQL-Ansicht
  - Speichern als Abfrage
  - Ausführen der Abfrage

– Warum kann die Anweisung Nr. 1 nicht mehrfach hintereinander ausgeführt werden?

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

- SELECT
- INSERT
- UPDATE
- DELETE

SQL SELECT-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

SQL SELECT-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

SQL-Abfragen als Datenquelle für Formulare

– nur SELECT

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

SQL-Abfragen als Datenquelle für Formulare

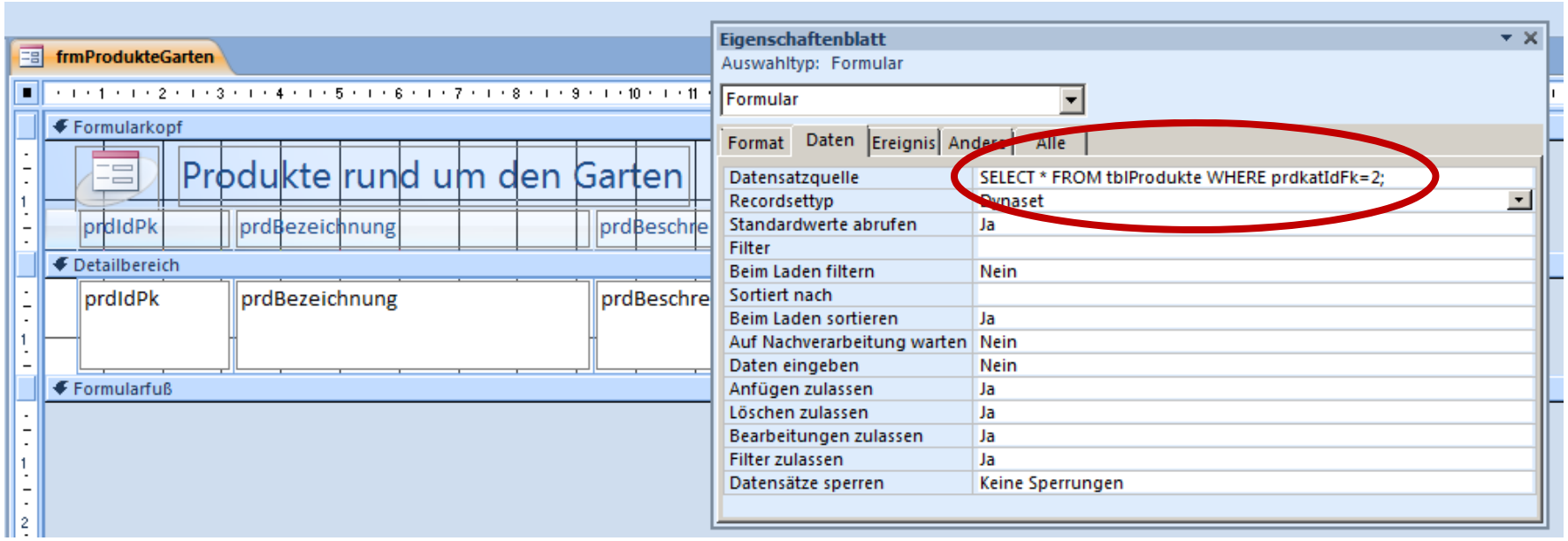
Schritt 1

– SQL-Abfrage überlegen

Schritt 2

– Formular erstellen

– SQL-Abfrage als Datensatzquelle eintragen



SQL-Abfragen als Datenquelle für Formulare

Schritt 1

– SQL-Abfrage überlegen

Schritt 2

– Formular erstellen

– SQL-Abfrage als Datensatzquelle eintragen
oder
SQL-Abfrage per Programmierung als Datensatzquelle festlegen

```
' Datensatzquelle als Eigenschaft des aktuellen Formulars
' kann so verändert werden
Me.RecordSource = "SELECT * FROM tblProdukte " & _
                  "WHERE prdkatIdFk=" & bytAktKategorie & ";"

' Aktualisierung des Formulars (Neuladen) erforderlich
Me.Requery
```

SQL in MS Access: Demo 06.01

D06.01 (Teil 1)

- Vorhanden ist das Formular eines Produktkatalogs, dass
  - bisher alle Produkte zeigt und
  - die Datensatzquelle tblProdukte verwendet
- Es soll zukünftig nur die Produkte der Kategorie 4 (Lebensmittel) zeigen
- Wie muss die SQL-Abfrage lauten?
- Ändern der Datensatzquelle, um die SQL-Abfrage zu verwenden
- Prüfen des Formularinhalts (nur noch Lebensmittel)
- …

SQL in MS Access: Demo 06.01

D06.01 (Teil 2)

- …
- Formular im Kopfbereich um zwei Schaltflächen "Alle Produkte" und "Getränke" erweitern
- zwei SQL-Abfragen überlegen
  - alle Produkte auswählen
  - nur Getränke (Produktkategorie 4) auswählen
- Ereignisprozedur für Klicken anlegen
  - Variable für SQL-Abfrage deklarieren (String)
  - Variable mit SQL-Abfrage initialisieren
  - Datensatzquelle des Formulars die SQL-Anweisung zuweisen
- Formular erproben: Umschalten zwischen allen Produkten und nur Getränken möglich

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

- SELECT
- INSERT
- UPDATE
- DELETE

SQL SELECT-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

SQL in MS Access

SQL-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

SQL-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

- INSERT
- UPDATE
- DELETE
- kein SELECT!

Vordefinierte VBA-Funktionen für SQL

Einfügen, Ändern und Löschen mit SQL durch Nutzung von

```
' Standardfunktionen aufrufen, als Parameter
' String mit SQL-Abfrage übergeben
Call CurrentDb.Execute("<SQL-Anweisung>")
Call DoCmd.RunSQL("<SQL-Anweisung>")
```

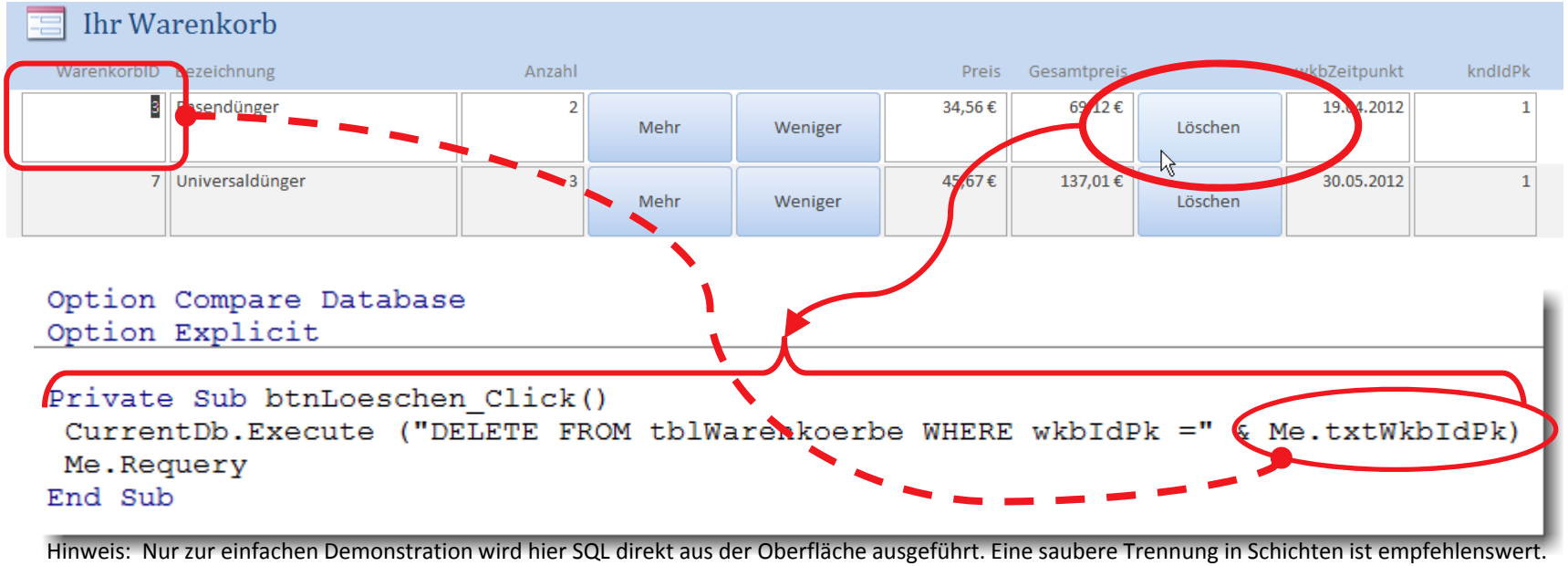
Um Auswirkungen in einem Formular sichtbar zu machen

```
' Aktualisierung des Formulars (Neuladen)
' erforderlich
Call Me.Requery
```

Vordefinierte VBA-Funktionen für SQL

Beispiel

- Löschen des aktuellen Warenkorb-Eintrags über neue Schaltfläche
- ID des aktuellen Eintrags kann mit Me.<FeldMitPrimärschlüssel> abgefragt werden
- Wird in SQL-Anweisung "eingebaut" und Anweisung wird ausgeführt
- Neuladen des Formulars notwendig



Hinweis: Nur zur einfachen Demonstration wird hier SQL direkt aus der Oberfläche ausgeführt. Eine saubere Trennung in Schichten ist empfehlenswert.

Vordefinierte VBA-Funktionen für SQL: Demo 06.02

D06.02

- Schaltfläche zum Löschen von Produkten im Formular des Warenkorbs einfügen
- Bei Klicken soll der in der aktuellen Zeile des Formulars dargestellte Kunde gelöscht werden
  - SQL-Abfrage überlegen und
  - in Ereignisprozedur ausführen

SQL in MS Access

SQL in der SQL-Ansicht von grafischen "Abfragen" zum Auswählen, Einfügen, Ändern und Löschen

SQL-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

- INSERT
- UPDATE
- DELETE
- kein SELECT!

SQL in MS Access

SQL-Abfragen als Datenquelle für Formulare

Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL

Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen

Inhalt

Ziel und Einordnung

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel
- Integritätsregeln
- Operationen auf Relationen und Tupeln

Grundlagen des Relationales Datenmodell (Teil 3) - SQL

- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

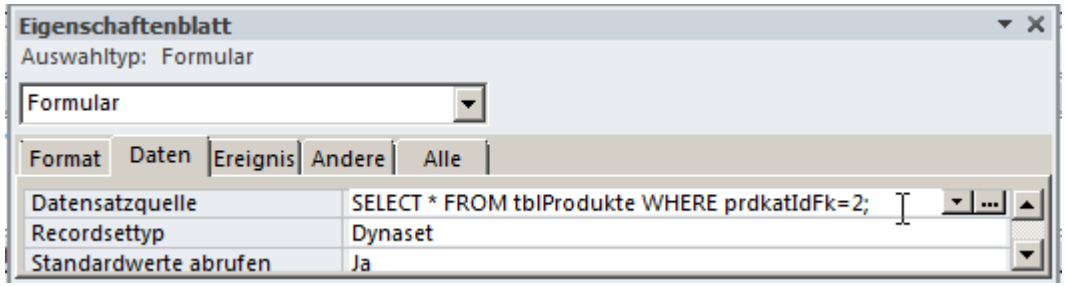
Arbeiten mit dem Relationalen Modell (Teil 3) - SQL in MS Access

- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

Zusammenfassung

SQL-Abfragen können als Datenquelle für Formulare verwendet werden



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

```
' Generelle Syntax
Call CurrentDB.Execute(<strSQL-Anweisung>)
```

Grafische Abfragen und SQL

- werden von MS Access in SQL übersetzt
- SQL-Ansicht zeigt das generierte SQL
- Ausführung SQL ist direkt über den entsprechenden Abfragetyp in der SQL-Ansicht möglich

Inhalt

Ziel und Einordnung

Rückblick

- Beziehungen im Relationen Modell
- Fremdschlüssel
- Integritätsregeln
- Operationen auf Relationen und Tupeln

Grundlagen des Relationales Datenmodell (Teil 3) - SQL

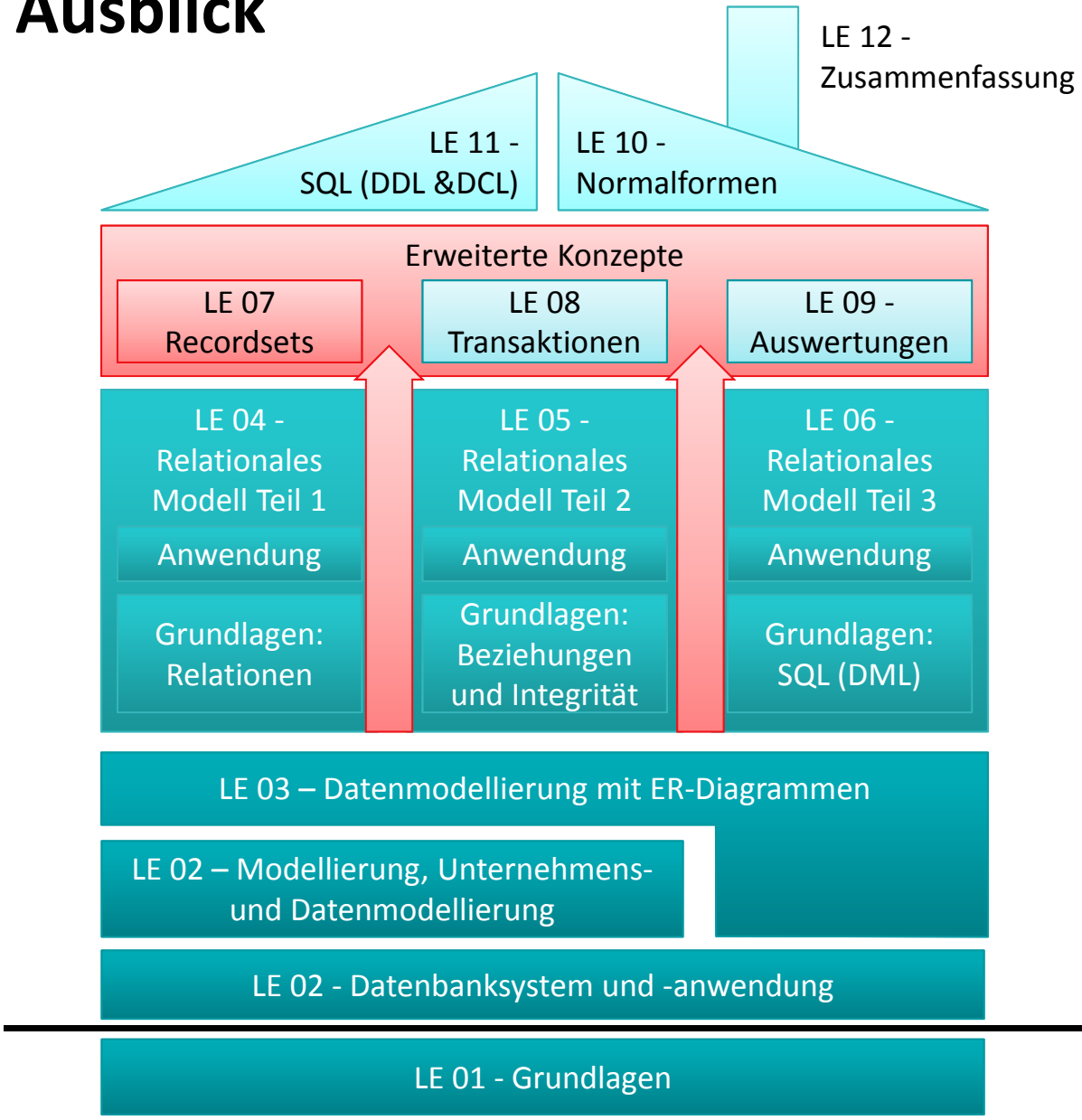
- Bestandteile
- Sprachumfang zum Auswählen, Einfügen, Ändern und Löschen
- Zusammenfassung

Arbeiten mit dem Relationalen Modell (Teil 3) - SQL in MS Access

- Formulare und SQL-Auswahlabfragen
- Vordefinierte VBA-Funktionen zum Einfügen, Ändern und Löschen mit SQL
- Zusammenhang und Abgrenzung grafischer Abfragen von SQL-Abfragen
- Zusammenfassung

Ausblick

Ausblick


LE 12 -
Zusammenfassung
LE 11 -
SQL (DDL &DCL)
LE 10 -
Normalformen
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

Quellen

Literatur

– [1] Jürgen Auer: Sql-Tutorial. http://www.sql-und-xml.de/sql-tutorial/

Abbildungen

– [1] Computer History Museum: Hall of Fellows, Donald Chamberlin; http://www.computerhistory.org/fellowawards/hall/bios/Donald,Chamberlin/
– [2] Homepage of Don Chamberlin; http://www.almaden.ibm.com/cs/people/chamberlin/

BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Wirtschaftsinformatik 2
LE 06 – Relationales Modell (Teil 3)
SQL

Prof. Dr. Thomas Off

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