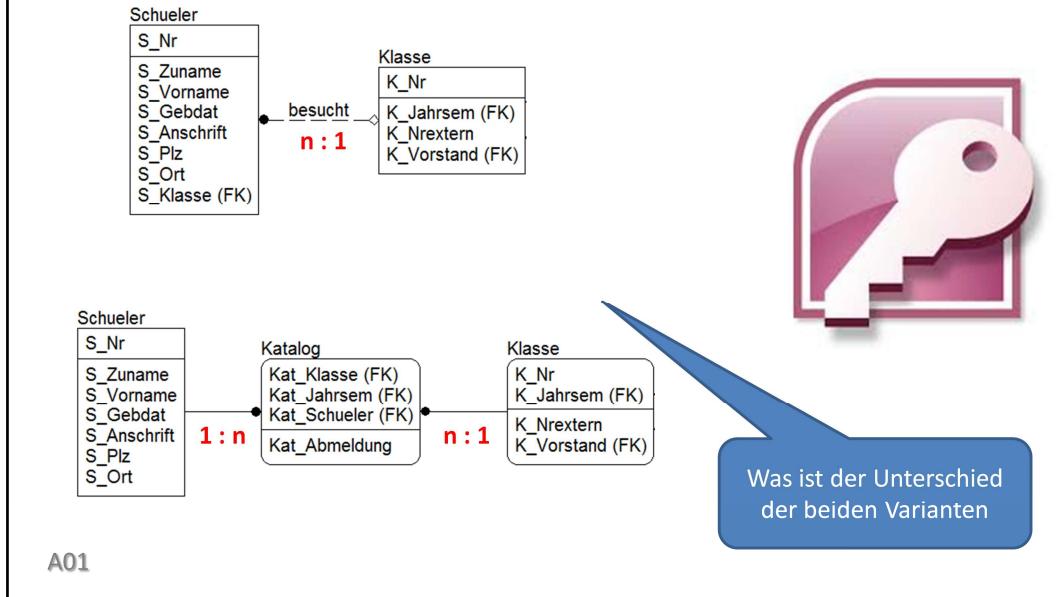


DBIS2 – Datenbanken und Informationssysteme



Die Startfolie enthält wieder eine Modellfrage (bezüglich unserer aktuellen Datenbank)

Die bisherige Datenbank enthielt nur die Klassen des aktuellen Jahres,
daher war die Beziehung von Klasse zu Schüler 1:n

Im Detail:

1 Klasse hat n Schüler,
1 Schüler besucht aber nur exakt eine Klasse

Möchte man aber die Klassen mehrerer Jahre speichern, dann

- ändert sich der PK der Klasse auf Klassennummer und Schuljahr (K_Nr, K_Jahrsem)
- Ein Schüler kann jetzt (im Lauf der Jahre) mehrere Klassen besuchen
- Daher ist die Beziehung Klasse-Schueler nicht mehr 1:n sondern n:m
- n:m Beziehungen kann man zwar auf Modelle zeichnen, aber nicht in Datenbanken abbilden
--- hier bedarf es einer Zwischentabelle
- Tabelle Katalog dient nun der Verbindung zwischen Klasse und Schueler

Und das ist der Unterschied zwischen oberen und unterem Modellausschnitt



**1) ABFRAGEN (INNER JOIN UND
OUTER JOIN)**

**2) „ABFRAGEN“ WELCHE
DATEN VERÄNDERN**

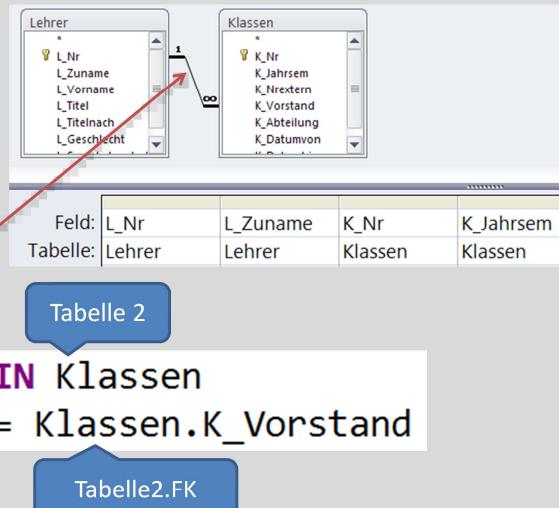
Wiederholung

3

Wiederholung Abfragen mit mehreren Tabellen



- **SQL - Stärke liegt im JOIN mehrerer Tabellen**
- Es können 2 oder mehr Tabellen im FROM kombiniert werden



DIE SPENGERGASSE

DBIS2 - Datenbank und Informationssysteme

4

```
SELECT Lehrer.L_Nr, Lehrer.L_Zuname, Klassen.K_Nr, Klassen.K_Jahrsem
FROM Lehrer INNER JOIN Klassen
  ON Lehrer.L_Nr = Klassen.K_Vorstand
ORDER BY Lehrer.L_Nr;
```

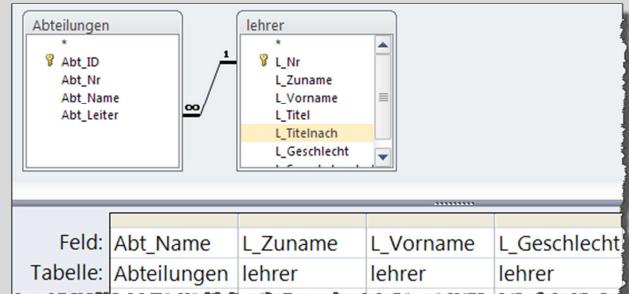
INNER JOIN wird benutzt um die Tabellen zu verbinden

ON Klausel definiert, welche Datensätze aus den beiden Tabellen kombiniert werden, praktisch wird hier immer eine existierende Beziehung verwendet, also tabelle1.PrimaryKey = tabelle2.ForeignKey

Wiederholung Abfragen mit mehreren Tabellen



- Abteilungen mit Namen des AV



```
SELECT Abt_Name, L_Zuname, L_Vorname, L_Geschlecht
FROM lehrer INNER JOIN Abteilungen
    ON lehrer.L_Nr = Abteilungen.Abt_Leiter
```

--Gebe alle Abteilungen mit Namen und Geschlecht des AV aus

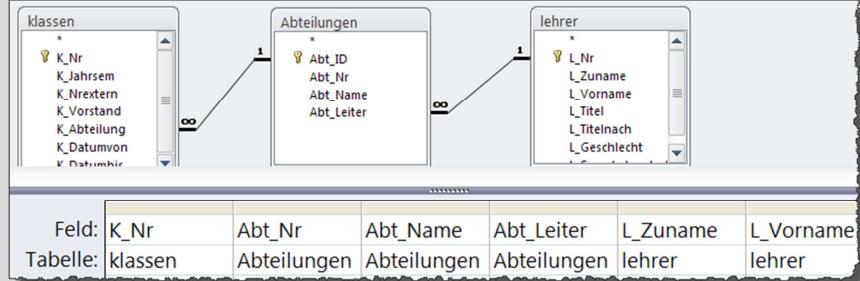
```
SELECT Abt_Nr, Abt_Name, Abt_Leiter, L_Zuname, L_Vorname, L_Geschlecht
FROM Abteilungen INNER JOIN lehrer ON Abteilungen.Abt_Leiter = Lehrer.L_Nr
```

Anmerkung: Wodurch ist die Anzahl der Ausgabesätze bestimmt?

Wiederholung Abfragen mit mehreren Tabellen



- Klassen mit dem Namen des AV



```

SELECT K_Nr, Abt_Nr, Abt_Name, Abt_Leiter, L_Zuname, L_Vorname
FROM (klassen INNER JOIN Abteilungen
      ON Klassen.K_Abteilung = Abteilungen.Abt_ID)
      INNER JOIN lehrer ON Abteilungen.Abt_Leiter = Lehrer.L_Nr
  
```

--Gebe alle Klassen mit dem Namen des AV aus

```

SELECT K_Nr, Abt_Nr, Abt_Name, Abt_Leiter, L_Zuname, L_Vorname
FROM (klassen INNER JOIN Abteilungen ON Klassen.K_Abteilung = Abteilungen.Abt_ID)
      INNER JOIN lehrer ON Abteilungen.Abt_Leiter = Lehrer.L_Nr
  
```

-- das setzen der Klammern im FROM ist aufgrund einer Access Laune notwendig

-- (nur wenn man den From per Hand schreibt)

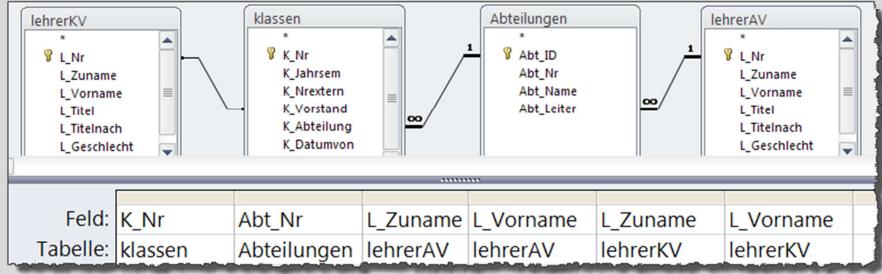
Die Angabe sagt zwar nichts von der Abteilung, es ist aber notwendig sie im Befehl zu verwenden, weil für den Av Namen keine direkte Beziehung Klasse-Lehrer möglich ist

Anmerkung: Wodurch ist die Anzahl der Ausgabesätze bestimmt?

Wiederholung Abfragen mit mehreren Tabellen



- Klassen mit Namen des AV und es KV



```
SELECT klassen.K_Nr, Abteilungen.Abt_Nr,
       lehrerAV.L_Zuname, lehrerAV.L_Vorname, lehrerKV.L_Zuname, lehrerKV.L_Vorname
FROM ((klassen INNER JOIN Abteilungen ON Klassen.K_Abteilung = Abteilungen.Abt_ID)
      INNER JOIN lehrer as lehrerAV ON Abteilungen.Abt_Leiter = LehrerAV.L_Nr)
      INNER JOIN lehrer as lehrerKV ON Klassen.K_Vorstand = LehrerKV.L_Nr)
```

Umbenennen der Tabelle lehrer, nötig weil sie 2 mal für verschiedene Zwecke vorkommt

-- Ja, dieser ist eigentlich zu schwer für die 2. Klasse

-- Gebe alle Klassen mit dem Namen des KV und des AV aus

SELECT K_Nr, Abt_Nr,

 lehrerAV.L_Zuname as AVzuname, lehrerAV.L_Vorname as AVvorname,
 lehrerKV.L_Zuname as KVzuname, lehrerKV.L_Vorname as KVvorname

FROM ((klassen INNER JOIN Abteilungen ON Klassen.K_Abteilung = Abteilungen.Abt_ID)

 INNER JOIN lehrer as lehrerAV ON Abteilungen.Abt_Leiter = LehrerAV.L_Nr)

 INNER JOIN lehrer as lehrerKV ON Klassen.K_Vorstand = LehrerKV.L_Nr

-- in der Entwurfsansicht ist dieser Befehl eher leichter, man muss nur lehrer

-- 2 mal reingeben und die überschüssigen JOIN Linien entfernen

-- im Handbetrieb muss man "lehrer as lehrerAV" die Tabelle umbenennen, damit man

-- im ON und SELECT die richtige Tabelle ansprechen kann

-- logisch verwendet dieser Befehl die Tabellen lehrerKV, Klassen, Abteilungen, lehrerAV

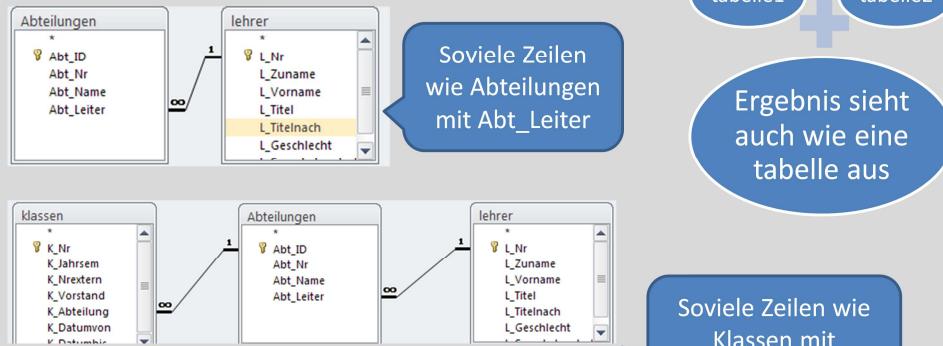
-- zufällig haben lehrerKV und lehrerAV die gleichen Feldnamen, weshalb man

-- immer die Notation tabellenname.feldname verwenden muss

Wiederholung Abfragen mit mehreren Tabellen



- Was bestimmt die Anzahl der Ausgabesätze beim JOIN
→ **Die Einträge in den FK Feldern**

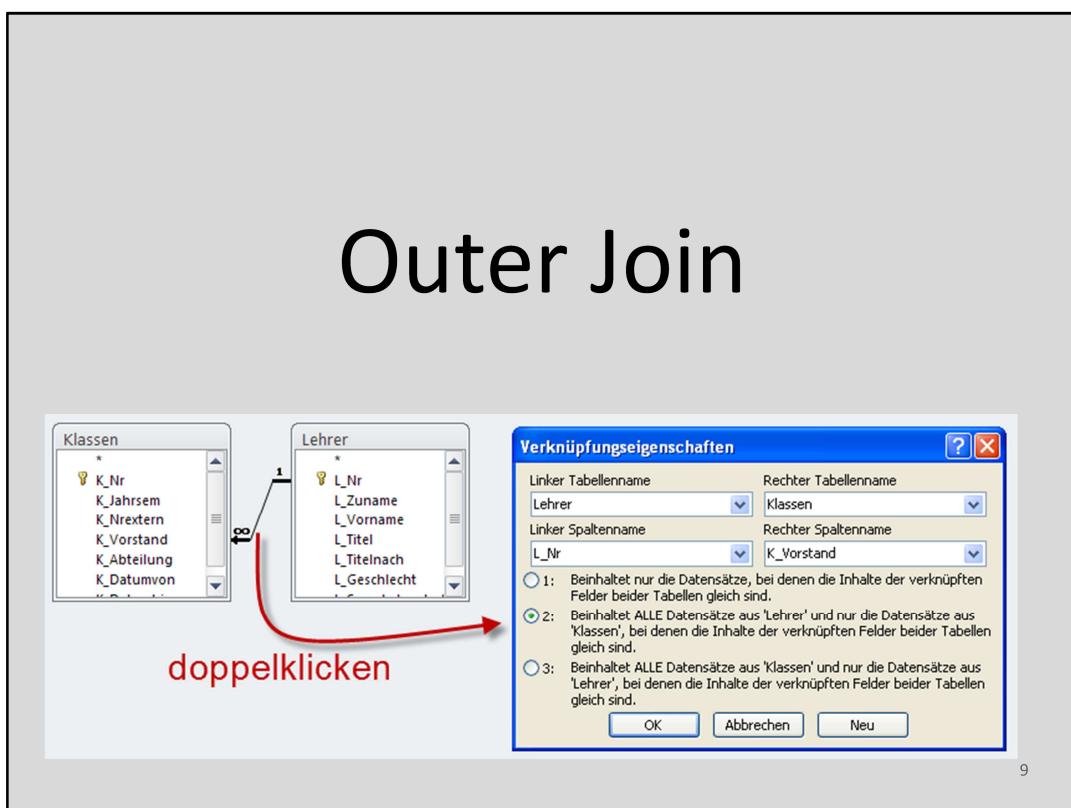


- Das kann man ändern, wenn statt INNER JOIN einen OUTER JOIN verwendet wird

Weil man Abfragen wohl immer schreibt um eine bestimmte Ausgabe zu bekommen

- ist es gut, dass das Ergebnis einer Abfrage aussieht wie eine Tabelle und auch fast so verwendet werden kann
- das wir wissen, welche Datensätze wir zu erwarten haben (Fremdschlüssel bestimmt!)

Outer Join



OUTER JOIN ist der Überbegriff für mehrere Schreibvarianten

LEFT JOIN (auch LEFT OUTER JOIN)
RIGHT JOIN (auch RIGHT OUTER JOIN)

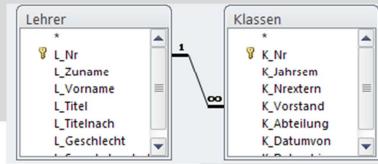
Nicht in Access verfügbar und auch selten benötigt
wäre noch FULL OUTER JOIN

Abfragen in Access OUTER JOIN



- Liefere die Klassenvorstände mit Namen (Sicht aus der Klasse)

```
FROM Lehrer INNER JOIN Klassen  
    ON Lehrer.L_Nr = Klassen.K_Vorstand
```



- Liefere alle Lehrer mit Info ob sie KV sind (Sicht vom Lehrer)

```
FROM Lehrer LEFT JOIN Klassen  
    ON Lehrer.L_Nr = Klassen.K_Vorstand
```



Im 2. Befehl ist mit mehr Datensätzen zu rechnen
(es sei denn jeder Lehrer wäre auch KV)

OUTER JOIN wird erreicht indem man statt dem Wort INNER
LEFT oder RIGHT verwendet

In der Entwurfsansicht erscheint ein Pfeil beim Join-Strich,
mittels doppelklick auf den Join-Strich kann man den OUTER JOIN einstellen

```
SELECT Lehrer.L_Nr, Lehrer.L_Zuname, Klassen.K_Nr, Klassen.K_Jahrsem  
FROM Lehrer LEFT JOIN Klassen  
    ON Lehrer.L_Nr = Klassen.K_Vorstand  
ORDER BY Lehrer.L_Nr
```

Abfragen in Access OUTER JOIN



- 61 Lehrer, aber nur 27 Klassen
→ 27 Ausgabesätze

```
FROM Lehrer INNER JOIN Klassen  
ON Lehrer.L_Nr = Klassen.K_Vorstand
```



- 61 Lehrer, nur 27 Klassen
→ aber jetzt 66 Ausgabesätze

```
FROM Lehrer LEFT JOIN Klassen  
ON Lehrer.L_Nr = Klassen.K_Vorstand
```



alle Lehrer sind jetzt in der Ausgabe, auch wenn sie nicht KV sind, 5 Lehrer sind KV in 2 Klassen

HAE, PE, SKO, SO, SWH sind die 5 Lehrer, welche KV in 2 Klassen sind, Diese haben je 2 Ausgabezeilen (weil sie ja in 2 verschiedenen Klassen in K_Vorstand vorkommen)

Lehrer, die nicht KV sind haben in ihrer Ausgabezeile in K_Nr und K_Jahrsem den Wert NULL stehen

```
SELECT Lehrer.L_Nr, Lehrer.L_Zuname, Klassen.K_Nr, Klassen.K_Jahrsem  
FROM Lehrer LEFT JOIN Klassen  
ON Lehrer.L_Nr = Klassen.K_Vorstand  
ORDER BY Lehrer.L_Nr;
```

Abfragen in Access OUTER JOIN



- Wenn man für eine gewünschte Ausgabe 2 Tabellen braucht hängt es von der Fragestellung ab welche JOIN Variante besser ist.
Steht die Tabelle mit dem Fremdschlüssel im Vordergrund, dann einen INNER JOIN
Steht die Tabelle mit dem Primärschlüssel im Vordergrund, dann einen OUTER JOIN
- LEFT oder RIGHT ?
Egal, man weist auf die Tabelle, aus der man alle Datensätze sehen will (auf die PK Tabelle)



Z.B. bei Schuljahre → Klassen

will ich zu den Klassen den Schuljahresbeginn dann INNER JOIN (welcher immer performanter ist)

will ich alle Schuljahre ausgeben (auch wenn sie keine Klassen haben) dann OUTER JOIN

LEFT oder RIGHT ?

das hängt nur davon ab in welcher Reihenfolge man die Tabellen schreibt

normalerweise will man alle Datensätze aus jener Tabelle, wo der PK (der Beziehung) steht

LEFT = die links geschriebene Tabelle, RIGHT = die rechts geschriebene Tabelle

Verdrehst man die Tabellen im From muss man eben auch das LEFT oder RIGHT richtig setzen

```
SELECT Lehrer.L_Nr, Lehrer.L_Zuname, Klassen.K_Nr, Klassen.K_Jahrsem
FROM Lehrer LEFT JOIN Klassen
  ON Lehrer.L_Nr = Klassen.K_Vorstand
ORDER BY Lehrer.L_Nr;
```

-- ist identisch mit

```
SELECT Lehrer.L_Nr, Lehrer.L_Zuname, Klassen.K_Nr, Klassen.K_Jahrsem
FROM Klassen RIGHT JOIN Lehrer
  ON Lehrer.L_Nr = Klassen.K_Vorstand
ORDER BY Lehrer.L_Nr;
```

Abfragen in Access OUTER JOIN



- Eine sehr häufige Fragestellung, für OUTER JOINS ist Zeige die Datensätze einer PK Tabelle, bei denen der PK nicht im Fremdschlüssel vorkommt,

z.B.

Welche Lehrer sind keine KV's?



```
SELECT L_Nr, L_Zuname, K_Nr, K_Vorstand  
FROM Lehrer LEFT JOIN Klassen  
ON Lehrer.L_Nr = Klassen.K_Vorstand  
WHERE K_Vorstand IS NULL
```

Bei jeder in der Datenbank vorkommenden Beziehung kann man diese Frage stellen

Welche Lehrer sind keine KV's (welche PKs aus Lehrer (L_Nr) kommen nie in K_Vorstand (dem zugehörigen FK) vor

entscheidend dafür ist ein WHERE Zweig mit Fremdschlüssel IS NULL

```
SELECT Lehrer.L_Nr, Lehrer.L_Zuname, Klassen.K_Nr, Klassen.K_Jahrsem  
FROM Lehrer LEFT JOIN Klassen  
ON Lehrer.L_Nr = Klassen.K_Vorstand  
where K_Vorstand IS NULL
```

Allgemein formuliert würde das Muster für diesen Befehl so lauten:

```
SELECT felder_aus_PK-Tabelle  
FROM PK-Tabelle LEFT JOIN FK-Tabelle  
ON PK-Tabelle.pk = FK-Tabelle.fk  
WHERE fk IS NULL
```

Abfragen in Access Übungen für Sie schuldb1_V3.mdb



- Zeige Lehrernamen mit dem Text Ges_Lehrerlehrerin
(überlegen ob inner, outer join)
- Welche Klassen haben keine Schüler
- Zeige alle Lehrer, gebe daneben aus ob sie Abteilungsleiter sind (Abt_Name angeben)
- Welche Religionsbekenntnisse sind nie bei Schülern eingetragen
- Welche Schüler haben kein Religionsbekenntnis
(Achtung Fangfrage)

Abfragen, die Daten ändern können

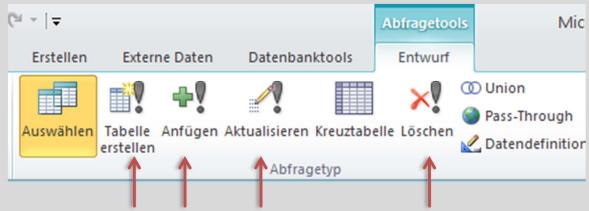
Insert Update Delete

15

Verändernde Abfragen in Access



- Im Abfrageentwurf sieht man weitere Abfragetypen
- Man erstellt zuerst eine normale (Auswahl) Abfrage und ändert diese dann in eine andere ab
- Will man z.B. mit den Lehrerdaten der Klassenvorstände eine neue Tabelle erstellen, dann entwickelt man zuerst die Auswahlabfrage und klickt dann auf „Tabelle erstellen“ und gibt den Namen der neuen Tabelle z.B. „tempKvdaten“ an



Wie Sie sicher schon früher bemerkt haben sieht das Ergebnis einer Abfrage aus wie eine Tabelle,

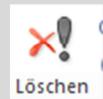
Es ist daher sehr leicht das Ergebnis in einer neuen Tabelle abzuspeichern
Einfach `INTO name_neuetabelle` vor den `FROM` schreiben

```
SELECT Lehrer.*  
INTO tempKVdaten  
FROM Lehrer INNER JOIN Klassen  
ON Lehrer.L_Nr = Klassen.K_Vorstand
```

Verändernde Abfragen in Access



- Das Hinzufügen neuer Daten wird man eher direkt im Datenblatt der Tabelle erledigen, es ist aber auch mit Befehlen möglich. Zuerst muss eine normale Abfrage sinnvolle neue Daten liefern dann → Die Zieltabelle wird gewählt und Zielfelder werden in der „Anfügen an“ Zeile angegeben
- Eher einfach ist es eine bestehende Abfrage (mit einer Tabelle im FROM) in eine Löschabfrage umzuwandeln, aber Vorsicht die gerade angezeigten Daten sind recht flott für immer gelöscht!
Versuchen Sie mal Ihre Klasse zu löschen



Hinzufüge Abfragen (INSERT Befehl aus SQL) üben wir im Moment nicht, weiter unten findet sich ein Muster

Ein praktischer Einsatz wäre gegeben, wenn wir beim Datenimport (z.B. aus Excel) nicht direkt in die Zieltabelle einfügen sondern Zwischenschritte erledigen müssen

Um Daten zu löschen (DELETE Befehl aus SQL) erstellt man zuerst eine Abfrage (wenn geht mit nur einer Tabelle)

welche genau die Daten anzeigt, die gelöscht werden sollen

Dann klickt man auf Löschabfrage und auf ausführen (das rote Rufzeichen)

--Z.B.

```
SELECT K_Nr FROM Klassen WHERE K_Nr = "2CHIF"
```

-- daraus wird dann (hier ohne unnötige Zeichen dargestellt)

```
DELETE FROM Klassen WHERE K_Nr = "2CHIF,"
```

-- führt man den Befehl aus so erhält man aber eine Fehlermeldung
„Schlüsselverletzung“

-- tja, man kann eben dank der Beziehungen (und der referenziellen Integrität)
-- keine Datensätze löschen, deren PK anderswo in FKs vorkommt

Verändernde Abfragen in Access



- Aktualisierungsabfragen (UPDATE Befehl)
braucht man schon eher. Aufgabe:
Ändern Sie bei Ihrem eigenen Datensatz (nur bei diesem)
die Adresse auf sinnvolle Werte
- Zuerst eine Abfrage erstellen, die nur Ihren Datensatz als
Ausgabe hat (WHERE pk = wert wäre sinnvoll), dabei die
Adressfelder ausgeben
- Auf Aktualisieren drücken und in der „Aktualisieren“ Zeile
die neuen
Werte
angeben

Feld:	S_Nr	S_Zuname	S_Strasse	S_Hausnummer	S_Postleitzahl	S_Postort
Tabelle:	Schueler	Schueler	Schueler	Schueler	Schueler	Schueler
Aktualisieren:			"Spengergasse"	"20"	1050	"Wien"
Kriterien:	2					

DIE SPENGERGASSE

DER WEG ZUM ERFOLG

DBIS2 - Datenbank und Informationssysteme

18

-- Die Auswahlabfrage könnte also so aussehen (nur wenn Sie Bittermann sind)

```
SELECT S_Nr, S_Zuname, S_Strasse, S_Hausnummer, S_Postleitzahl, S_Postort
FROM Schueler
WHERE S_Nr = 2
```

-- wird zu

```
UPDATE Schueler
SET Schueler.S_Strasse = "Spengergasse",
    Schueler.S_Hausnummer = "20",
    Schueler.S_Postleitzahl = 1050,
    Schueler.S_Postort = "Wien"
WHERE S_Nr = 2
```

-- der WHERE Zweig ist bei Update und Delete Befehlen enorm wichtig,
-- wird er versehentlich weggelassen, dann verändern/löschen Sie die ganze Tabelle

Verändernde Abfragen in Access



- Abhängig vom WHERE Zweig werden auch mehrere Datensätze verändert

Setzen Sie alle 2. HIF Klassen in ein anderes Schuljahr

```
UPDATE Klassen SET K_Jahrsem = 20090 WHERE K_Nr LIKE "2?HIF"
```

Setzen Sie alle Klassen aus dem Schuljahr 20090 wieder zurück ins 20100 Schuljahr

```
UPDATE Klassen SET K_Jahrsem = 20100 WHERE K_Jahrsem = 20090
```

- Hinweis: Bei falschem (oder gar fehlendem) WHERE Zweig kann sofort enormer Schaden angerichtet werden

Abfragen in Access Übungen für Sie schuldb1_V3.mdb



- Stelle das Feld S_Staatsbuergerschaft in Schueler auf österreichische Staatsbürgerschaft, falls es leer ist.
- Entferne aus der Tabelle Religionen den Eintrag für Rel_Name=„----Sonstige----“
- Füge zur Tabelle Schueler ein Feld namens S_Alter hinzu und befülle es mit dem Alter in Tagen (Berechnung Tagesdatum – S_Gebdat)
- Was ist der Zweck des folgenden Befehls:

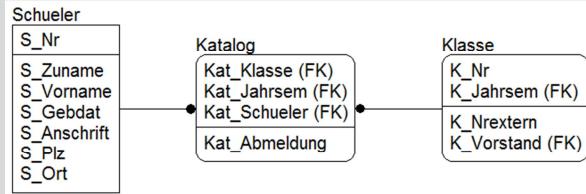
```
UPDATE Schuljahre INNER JOIN Klassen
    ON Schuljahre.Sja_Nr = Klassen.K_Jahrsem
    SET Klassen.K_Datumvon = Sja_Datumvon
    where Klassen.K_Datumvon IS NULL
```

TagesDatum ohne Zeit = Date() Datum()
Aktueller Timestamp (Tagesdatum mit Zeit) = Now Jetzt

Abfragen in Access Übungen für Sie schuldb1_V3.mdb



- Auf der 1. Folie wird eine Tabelle Katalog als Auflösung der n:m Beziehung zwischen Schueler und Klasse vorgestellt:



- Erstellen Sie diese Tabelle mit den Daten des aktuellen Klassenbesuchs und setzen Sie anschliessend den passenden Primary Key (3 Felder)
- Entfernen Sie die bisherige Schueler-Klassen Beziehung, verändern Sie den PK von Klasse und erstellen Sie die beiden neuen Beziehungen