

Einführung in Datenbanken

Übung 7: Logik

Prof. Dr. Stefan Brass

PD Dr. Alexander Hinneburg

Martin-Luther-Universität Halle-Wittenberg

Wintersemester 2024/25

<http://www.informatik.uni-halle.de/~brass/db24/>

Inhalt

- 1 Präsenzaufgabe 6
- 2 Logik
- 3 Hausaufgabe 6
- 4 Hausaufgabe 5

Präsenzaufgabe: Variablenbelegungen (1)

- Wählen Sie das Schema „**studentenaufgaben_public**“ im Adminer:

[https://dbs.informatik.uni-halle.de/edb?pgsql=db&username=student_gast&db=postgres&ns=]

- Aufgabe:** Suchen Sie eine FROM-Klausel aus (gern auch mit mehreren Tupelvariablen), und eine WHERE-Bedingung, so dass

```
SELECT COUNT(*)
```

```
FROM _____
```

```
WHERE _____
```

genau den Wert 40 liefert.

Es wird so die Anzahl der Variablenbelegungen gezählt, die die WHERE-Bedingung erfüllen. Die Tabellen-Größen sind: AUFGABEN: 3, STUDENTEN: 4, BEWERTUNGEN: 8.

Es gibt 5 Bewertungen mit ATYP = 'H' und 2 Aufgaben mit ATYP = 'H'.

Mehrere Tupelvariablen über einer Tabelle sind möglich.

Präsenzaufgabe: Variablenbelegungen (2)

STUDENTEN

<u>SID</u>	VORNAME	NACHNAME	EMAIL
101	Lisa	Weiss	...
102	Michael	Grau	NULL
103	Daniel	Sommer	...
104	Iris	Winter	...

AUFGABEN

<u>ATYP</u>	<u>ANR</u>	THEMA	MAXPT
H	1	ER	10
H	2	SQL	10
Z	1	SQL	14

BEWERTUNGEN

<u>SID</u>	<u>ATYP</u>	<u>ANR</u>	PUNKTE
101	H	1	10
101	H	2	8
101	Z	1	12
102	H	1	9
102	H	2	9
102	Z	1	10
103	H	1	5
103	Z	1	7

Präsenzaufgabe: Variablen-Belegungen (3)

Lösung:

- Ziel dieser Aufgabe ist es, zu verstehen, dass
 - SQL alle möglichen Variablenbelegungen betrachtet, und
 - die Anzahl möglicher Kombinationen von Belegungen für mehrere Tupelvariablen das Produkt der Anzahl möglicher Belegungen für die einzelnen Variablen ist.

Sofern man keine Bedingungen hat, die nur bestimmte Kombinationen zulassen. Wenn man Join-Bedingungen hat (die sich auf mehrere Variablen beziehen) werden die möglichen Variablenbelegungen gefiltert, und nur das Ergebnis dieser Filterung wird unter SELECT gezählt.

- Eine Primfaktor-Zerlegung von 40 ist:

$$40 = 2 * 2 * 2 * 5$$

Präsenzaufgabe: Variablen-Belegungen (4)

Lösung, Forts.:

- Wegen $40 = 2 * 2 * 2 * 5$ erreicht man das Ziel also, wenn man vier Tupelvariablen **A**, **B**, **C**, **D** deklariert, wobei es
 - für A 2 mögliche Belegungen gibt,
 - für B auch,
 - für C ebenso, und
 - für D 5.

```
SELECT COUNT(*)
FROM   AUFGABEN A, AUFGABEN B,
        AUFGABEN C, BEWERTUNGEN D
WHERE  A.ATYP = 'H' AND B.ATYP = 'H'
AND    C.ATYP = 'H' AND D.ATYP = 'H'
```

Präsenzaufgabe: Variablen-Belegungen (5)

Lösung, Forts.:

- Alternativen für A, B, C mit 2 möglichen Belegungen:
 - BEWERTUNGEN X mit X.SID = 103 (Aufgabenstellung)
 - AUFGABEN X mit X.ATYP = 'H'
 - AUFGABEN X mit X.ANR = 1
 - STUDENTEN X mit X.SID <= 102
- Alternativen für D mit 5 möglichen Belegungen:
 - BEWERTUNGEN D mit D.ATYP = 'H' (Aufgabenstellung)
 - BEWERTUNGEN D mit D.SID >= 102

Präsenzaufgabe: Variablen-Belegungen (6)

Lösung, Forts.:

- Man muss keine Primfaktoren verwenden: $40 = 8 * 5$

```
SELECT COUNT(*)
FROM   BEWERTUNGEN A, BEWERTUNGEN B
WHERE  B.ATYP = 'H'
```

- Auch möglich: $40 = 2 * 4 * 5$

```
SELECT COUNT(*)
FROM   AUFGABEN A, STUDENTEN S, BEWERTUNGEN B
WHERE  A.ATYP = 'H' AND B.ATYP = 'H'
```

- Ganz anders (natürlich nicht gewünscht, nicht portabel):

```
SELECT COUNT(*)
FROM   GENERATE_SERIES(1, 40)
```

Tabellenfunktion: Erzeugt 40 Zeilen mit Zahlwerten 1, 2, 3, ..., 40.

Inhalt

1 Präsenzaufgabe 6

2 Logik

3 Hausaufgabe 6

4 Hausaufgabe 5

Analyse von Verbunden (1)

Aufgabe:

- Erkennen Sie das Problem in dieser Anfrage?
Ziel ist es, alle Studenten auszugeben, die eine Aufgabe über SQL und eine über ER-Entwurf gelöst haben.

```
SELECT S.VORNAME, S.NACHNAME
FROM   STUDENTEN S, BEWERTUNGEN B,
        AUFGABEN A1, AUFGABEN A2
WHERE  S.SID = B.SID
AND    B.ATYP = A1.ATYP AND B.ANR = A1.ANR
AND    B.ATYP = A2.ATYP AND B.ANR = A2.ANR
AND    A1.THEMA = 'SQL'
AND    A2.THEMA = 'ER'
```

Analyse von Verbunden (2)

Lösung:

- Die Anfrage ist inkonsistent. Das Ergebnis ist immer leer.
- Man braucht zwei Tupelvariablen über BEWERTUNGEN:

```
SELECT S.VORNAME, S.NACHNAME
FROM   STUDENTEN S,
        BEWERTUNGEN B1, BEWERTUNGEN B2,
        AUFGABEN A1, AUFGABEN A2
WHERE  S.SID = B1.SID AND S.SID = B2.SID
AND    B1.ATYP = A1.ATYP AND B1.ANR = A1.ANR
AND    B2.ATYP = A2.ATYP AND B2.ANR = A2.ANR
AND    A1.THEMA = 'SQL'
AND    A2.THEMA = 'ER'
```

Analyse von Verbunden (3)

Aufgabe:

- Was ist das Ergebnis dieser Anfrage?

```
SELECT B.SID, B.PUNKTE
FROM   BEWERTUNGEN B, AUFGABEN A
WHERE  B.ATYP = 'H' AND B.ANR = 1
```

- Die Tupelvariable A ist deklariert, aber nie verwendet.
- Liefert die Anfrage dasselbe Ergebnis wie die Anfrage ohne die Tupelvariable A?

```
SELECT B.SID, B.PUNKTE
FROM   BEWERTUNGEN B
WHERE  B.ATYP = 'H' AND B.ANR = 1
```

Analyse von Verbunden (4)

Lösung:

- Die Tupelvariable A erzeugt Duplikate. Weil es drei Zeilen in der Tabelle **AUFGABEN** gibt, wird jede Bewertung für Hausaufgabe 1 drei Mal gedruckt.
- Bis auf Duplikate geben die beiden Anfragen immer das gleiche Ergebnis aus.

Aufgabe:

- Gilt das hier auch?

```
SELECT A.ANR, A.THEMA
FROM   AUFGABEN A, BEWERTUNGEN B
WHERE  A.ATYP = 'H'
```

Analyse von Verbunden (5)

Lösung:

- Nein, die Tabelle **BEWERTUNGEN** kann leer sein, auch wenn **AUFGABEN** nicht leer ist.
- In dem Fall ist das Anfrageergebnis mit **BEWERTUNGEN** auch leer (es gibt keine Variablenbelegungen).
- Man kann daher nicht sagen, dass sich die beiden Anfragen (mit und ohne **BEWERTUNGEN B**) nur in Duplikaten unterscheiden.

Analyse von Verbunden (6)

Aufgabe:

- Liegt hier ein unnötiger Join vor?

```
SELECT S.SID, B.ATYP, B.ANR, B.PUNKTE  
FROM   BEWERTUNGEN B, STUDENTEN S  
WHERE  B.SID = S.SID AND S.SID > 101
```

- D.h. kann man auf eine der beiden Tupelvariablen verzichten?

Analyse von Verbunden (7)

Lösung:

- Ja, die gegebene Anfrage enthält einen unnötigen Join:

```
SELECT S.SID, B.ATYP, B.ANR, B.PUNKTE
FROM   BEWERTUNGEN B, STUDENTEN S
WHERE  B.SID = S.SID AND S.SID > 101
```

- Sie ist äquivalent zu folgender Anfrage:

```
SELECT B.SID, B.ATYP, B.ANR, B.PUNKTE
FROM   BEWERTUNGEN B
WHERE  B.SID > 101
```

- Es gibt eine Schlüssel-Fremdschlüssel-Beziehung und die Tupelvariable mit dem Schlüssel ist verzichtbar.
- Von ihr werden nur die Schlüsselattribute verwendet.

Analyse von Verbunden (8)

Aufgabe:

- Sind die folgenden zwei Anfragen äquivalent, d.h. liegt auch hier ein unnötiger Join vor?

```
SELECT S.VORNAME, S.NACHNAME  
FROM   STUDENTEN S
```

```
SELECT DISTINCT S.VORNAME, S.NACHNAME  
FROM   STUDENTEN S, BEWERTUNGEN B  
WHERE  S.SID = B.SID
```

Analyse von Verbunden (9)

Lösung:

- Nein, die Anfragen sind nicht äquivalent:
 - Die erste Anfrage liefert alle Studenten.
 - Die zweite Anfrage liefert nur Studierende, die mindestens eine Aufgabe abgegeben haben.
- Auch hier liegt eine Schlüssel-Fremdschlüssel-Beziehung vor.
- Die Tupelvariable auf der Fremdschlüssel-Seite kann man normalerweise nicht einsparen.
- Der Join filtert Tupel ohne Join-Partner heraus.

In der umgekehrten Richtung gibt es dagegen immer einen Joinpartner:
Zu Zeilen in der Tabelle mit dem Fremdschlüssel gibt es immer eine passende Zeile in der Tabelle mit dem Schlüssel (das ist gerade die Integritätsbedingung).

Datenbank: Komponisten, Musik-CDs

- Die Aufgaben dieses Übungsblattes beziehen sich auf die Datenbank mit Informationen über klassische Musik-CDs:

`https://dbs.informatik.uni-halle.de/edb?
pgsql=db&username=student_gast&
db=postgres&ns=komponist_public`

- `KOMPONIST(KNR, NAME, VORNAME°, GEBOREN°, GESTORBEN°)`
- `STUECK(SNR, KNR°→KOMPONIST, TITEL, TONART°, OPUS°)`
- `CD(CDNR, NAME, HERSTELLER°, ANZ_CDS°, GESAMTSPIELZEIT°)`
- `AUFNAHME(CDNR→CD, SNR→STUECK, ORCHESTER°, LEITUNG°)`
- `SOLIST((CDNR, SNR)→AUFNAHME, NAME, INSTRUMENT°)`

Aufgabe 6.1: Zeitgenossen von Mozart (1)

- Gesucht sind alle Komponisten in der Datenbank, die Zeitgenossen von Wolfgang Amadeus Mozart sind, deren Lebenszeit also mit der von Mozart überlappt.

Es sollen dabei auch Komponisten ausgegeben werden, die in dem Jahr gestorben sind, in dem Mozart geboren wurde, bzw. die in dem Jahr geboren wurden, in dem Mozart gestorben ist (wenn die Überlappung also etwas grenzwertig ist).

- „Wolfgang Amadeus Mozart“ selbst soll in der Auflistung nicht erscheinen.

Sie können voraussetzen, dass Name und Vorname des Komponisten zusammen einen Alternativschlüssel darstellen, dass es also nicht zwei verschiedene Einträge mit gleichem Namen und Vornamen gibt.

Aufgabe 6.1: Zeitgenossen von Mozart (2)

- Geben Sie Name, Vorname, Geburts- und Todesjahr der Komponisten aus, sowie das „Alter“ des Komponisten im Geburtsjahr von Mozart.

Das kann natürlich negativ sein, wenn der andere Komponist erst nach Mozart geboren wurde.

- Die Spalte **Alter** nennen Sie bitte in der Ausgabe genau so (auch mit dieser Groß-/Kleinschreibung).
- Sie dürfen nicht die Lebensdaten von Mozart in die Anfrage einsetzen.

Zwar werden die sich nicht mehr ändern (1756–1791), aber der Name „Wolfgang Amadeus Mozart“ soll nur ein Beispiel sein. Ihre Anfrage könnte z.B. später in einem Programm verwendet werden, mit Parametern für den tatsächlichen Namen.

Aufgabe 6.1: Zeitgenossen von Mozart (3)

name	vorname	geboren	gestorben	Alter
Telemann	Georg Philipp	1681	1767	75
Händel	Georg Friedrich	1685	1759	71
Scarlatti	Domenico	1685	1757	71
Locatelli	Pietro	1695	1764	61
Leclair	Lean-Marie	1697	1764	59
Mozart	Leopold	1719	1787	37
Hayden	Joseph	1732	1809	24
Beethoven	Ludwig van	1770	1827	-14

Aufgabe 6.1: Zeitgenossen von Mozart (4)

- Verwenden Sie bitte nur Konstrukte, die in der Vorlesung schon behandelt wurden.
- Insbesondere sind Joins unter **FROM** ausgeschlossen.

In der Klausur gibt es natürlich keine solchen Beschränkungen (sehr ungewöhnliche Konstrukte könnten aber zu Punktabzügen wegen mangelnder Portabilität führen).
- Wenn in der Aufgabenstellung keine spezielle Sortierung verlangt ist (wie hier), dürfen Sie eine beliebige (sinnvolle) Sortierung vornehmen (oder eben nicht sortieren).
- Die obige Tabelle ist mit
ORDER BY GEBOREN, NAME
(am Ende der Anfrage) erzeugt.

Aufgabe 6.2: CDs mit mehreren Komponisten (1)

- Gesucht sind alle CDs, die Stücke von mehr als einem Komponisten enthalten.

Also von mindestens zwei verschiedenen Komponisten.

- Geben Sie jeweils den Namen der CD aus.
- Bei dieser Anfrage werden Sie wahrscheinlich Duplikate erhalten. Verwenden Sie ggf. **SELECT DISTINCT**, um diese zu eliminieren.
- Für die Lösung dieser Aufgabe verwenden Sie bitte kein **GROUP BY** und keine Aggregationsfunktionen wie **COUNT**, selbst wenn Sie das schon kennen sollten.

In der Vorlesung war es noch nicht dran.

Aufgabe 6.2: CDs mit mehreren Komponisten (2)

- Das erwartete Ergebnis ist:

cdnr	name
106	Leopold Mozart: Sinfonia D-Dur ...
111	Mozart/Beethoven: Klassische Ouvertüren
125	Tschaikowsky/Mendelssohn: Violinkonzerte
139	Oboenkonzerte
140	Corelli,Albinoni,Scarlatti,Manfredini,...
142	Schlager um 1500

Aufgabe 6.3: Logische Analyse (1)

- Im folgenden sollen Sie jeweils zwei Anfragen vergleichen, u.a. hinsichtlich Äquivalenz.
- Anfrage 1:

```
SELECT KNR, SNR
FROM   STUECK
WHERE  KNR = 70
```

- Anfrage 2:

```
SELECT K.KNR, S.SNR
FROM   STUECK S, KOMPONIST K
WHERE  S.KNR = 70 AND K.KNR = 70
```

Aufgabe 6.3: Logische Analyse (2)

- Welche der folgenden Aussagen ist korrekt? Wenn mehrere korrekt sein sollten, wählen Sie die erste korrekte Aussage.
 - A. Die beiden Anfragen liefern immer die gleiche Antwort (äquivalent)
 - B. Die beiden Anfragen liefern bis auf Duplikate die gleiche Antwort
 - C. Das Ergebnis von Anfrage 1 ist immer leer (inkonsistent)
 - D. Das Ergebnis von Anfrage 2 ist immer leer (inkonsistent)
 - E. Anfrage 1 liefert immer eine Obermenge ($\supseteq$) von Anfrage 2
 - F. Anfrage 1 liefert immer eine Teilmenge ($\subseteq$) von Anfrage 2
 - G. Keine der Aussagen trifft zu

Bei „B.“ ist gemeint, dass die Tupelmengen (nach Duplikat-Eliminierung) identisch sind. Die Teile „E.“ und „F.“ beziehen sich auch auf die Tupel-Mengen ohne Berücksichtigung eventueller Duplikate.

Aufgabe 6.4: Logische Analyse

- Vergleichen Sie nach dem Schema von Aufgabe 3 die folgenden beiden Anfragen:

- Anfrage 1:

```
SELECT *  
FROM   KOMPONIST  
WHERE  KNR = 70 OR KNR = 80 AND GEBOREN < 1500
```

- Anfrage 2:

```
SELECT *  
FROM   KOMPONIST  
WHERE  KNR IN (70, 80) AND NOT GEBOREN >= 1500
```

Aufgabe 6.5: Logische Analyse

- Vergleichen Sie entsprechend die folgenden beiden Anfragen:

- Anfrage 1:

```
SELECT *  
FROM   STUECK  
WHERE  TONART = 'C-dur'  
AND    (TONART = 'a-moll'  
        OR TITEL LIKE 'Symphon%')
```

- Anfrage 2:

```
SELECT *  
FROM   STUECK  
WHERE  TONART = 'C-dur'  
AND    TITEL LIKE 'Symphon%'
```


5. Übungsblatt: Aufgabe 1 (1)

- Wählen Sie einen Fachvortrag vom Industrietag Informationstechnologie IT², der am 12.11.2024 von 14 bis 18 Uhr stattfindet (Vorträge 14–16 im Raum 3.07).
- Modellieren Sie eine im gewählten Vortrag beschriebene Anwendung, einen relevanten Teil, oder eine Beispiel-Anwendung aus dem Vortrag als relationales Datenbankschema mit etwa 2 bis 3 Tabellen.
 - Geben Sie die Tabellen mit Primär- und Fremdschlüsseln in Kurznotation wie auf **Folie 6-37** an.
 - Geben Sie weiterhin für alle Tabellen Tupel an, die verdeutlichen, was der Inhalten der einzelnen Spalten ist.

Der gesamte Datenbankzustand soll etwa 8–12 Tupel umfassen.

5. Übungsblatt: Aufgabe 1 (2)

- Anforderungen, Forts.:
 - Beschreiben Sie knapp und verständlich, was die Attribute der Tabellen bedeuten sollen.

Soweit es aus den Spaltennamen und den Beispieldaten nicht schon völlig klar ist.
 - Beschreiben Sie weiterhin mindestens einen Anwendungsprozess, der Daten in die Tabellen einfügt, liest oder aktualisiert.
- Sie können die Lösung als .txt oder als PDF-Datei abgeben.
- **Hinweis:** Für den Besuch des IT-Tags müssen (sollten) Sie sich frühzeitig über die Web-Seite anmelden.

[<https://www.uni-halle.de/uzi/veranstaltungen/39it/>]

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (1)

- Ein König lässt eine junge Dame einsperren, weil sie für mehr Umweltschutz demonstriert hat. Ihr Verlobter möchte sie gern befreien. Der König ist ein Fan von Logik-Rätseln, und stellt den Verlobten vor die Wahl zwischen drei Türen:
 - In einem der drei Räume ist seine Herzens-Dame. Wenn er diese Tür öffnet, darf er mit ihr gehen.
 - In einem anderen Raum ist ein hungriger Tiger. Wenn er diese Tür öffnet, wird er höchstwahrscheinlich aufgefressen.
 - Der dritte Raum ist leer. Er überlebt dann zwar, muss aber ohne seine Liebste gehen.

Dieses Logikrätsel stammt aus dem Buch „Dame oder Tiger“ von **Raymond Smullyan**. Sie können Auszüge aus dem titel-gebenden Abschnitt des Buches auf dieser Webseite finden: [<https://emath.de/Referate/Smullyan.pdf>]
Ich habe die Geschichte und das Rätsel leicht verändert.

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (3)

- Wir codieren die Möglichkeiten als Zeilen einer Tabelle:

MOEGLICH		
R1	R2	R3
D	T	
D		T
T	D	
	D	T
T		D
	T	D

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (4)

- Aus den Inschriften und den Zusatzbedingungen ergeben sich folgende logischen Formeln (jeweils zwei Wenn-Dann-Bedingungen pro Schild):

- $R1 = 'D' \rightarrow R1 = ' '$
- $R1 = 'T' \rightarrow \neg R1 = ' '$
- $R2 = 'D' \rightarrow R1 = 'T'$
- $R2 = 'T' \rightarrow \neg R1 = 'T'$
- $R3 = 'D' \rightarrow R1 = 'D'$
- $R3 = 'T' \rightarrow \neg R1 = 'D'$

In SQL gibt es leider kein „ $\rightarrow$ “ (wenn-dann), aber $A \rightarrow B$ ist bekanntlich äquivalent zu $\neg A \vee B$, und kann damit in SQL mit NOT und OR ausgedrückt werden. Wenn ein Tiger im jeweiligen Raum ist, ist die Inschrift des Schildes falsch und wird deswegen mit $\neg$ (NOT in SQL) negiert.

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (5)

- Ihre Aufgabe ist nun, die obigen Bedingungen in eine **WHERE**-Bedingung in SQL zu überführen (alle sechs Bedingungen müssen gelten, also mit **AND** verknüpft werden).
- Sie dürfen, wenn Sie möchten, **NOT** entfernen, indem Sie den jeweiligen Vergleich invertieren.
- Ansonsten bleiben Sie bitte möglichst nahe an den gegebenen Bedingungen (in der Geschichte wäre es ja fatal, wenn irgendein Fehler geschehen würde).

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (6)

- Leider gibt es die Tabelle **MOEGLICH** nicht in unserer Adminer-Installation, aber man kann auf folgende Weise eine lokale Tabelle nur für die eine Anfrage definieren:

```
WITH MOEGLICH(R1, R2, R3) AS
  (VALUES
    ('D', 'T', ' '),
    ('D', ' ', 'T'),
    ('T', 'D', ' '),
    (' ', 'D', 'T'),
    ('T', ' ', 'D'),
    (' ', 'T', 'D'))
SELECT *
FROM MOEGLICH
```

[https://users.informatik.uni-halle.de/~brass/db24/homework/h5_dt.sql]

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (7)

- Für diese Aufgabe ist wesentlich, dass man die folgende Äquivalenz kennt:
 - $A \rightarrow B$ ist äquivalent zu $\neg A \vee B$, bzw. in SQL: **NOT A OR B**.
- Eine „Wenn-Dann-Bedingung“ ist erfüllt gdw.
 - Die Voraussetzung falsch ist, oder
Dann ist egal, ob die Folgerung wahr oder falsch ist. Es ist ja keine „Genau-dann-wenn-Bedingung“.
 - die Folgerung wahr ist.
Dann ist egal, ob die Voraussetzung wahr oder falsch ist.
- Dies ist eine nützliche Äquivalenz, die man sich merken sollte.
Man braucht diese Äquivalenz z.B. für CHECK-Constraints, weil es kein $\rightarrow$ in SQL gibt, aber natürlich schon NOT und OR.

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (8)

- Lösung:

```

SELECT *
FROM   MOEGLICH
WHERE  -- An Raum 1: Dieser Raum ist leer.
      (NOT R1 = 'D' OR R1 = ' ')
AND    (NOT R1 = 'T' OR NOT R1 = ' ')
      -- An Raum 2: Der Tiger ist in Raum 1.
AND    (NOT R2 = 'D' OR R1 = 'T')
AND    (NOT R2 = 'T' OR NOT R1 = 'T')
      -- An Raum 3: Die Dame ist in Raum 1.
AND    (NOT R3 = 'D' OR R1 = 'D')
AND    (NOT R3 = 'T' OR NOT R1 = 'D')

```

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (9)

- Man kann noch jeweils das NOT eliminieren, indem man = in <> umwandelt:

```
SELECT *
FROM   MOEGLICH
WHERE  -- An Raum 1: Dieser Raum ist leer.
        (R1 <> 'D' OR R1 = ' ')
AND    (R1 <> 'T' OR R1 <> ' ')
        -- An Raum 2: Der Tiger ist in Raum 1.
AND    (R2 <> 'D' OR R1 = 'T')
AND    (R2 <> 'T' OR R1 <> 'T')
        -- An Raum 3: Die Dame ist in Raum 1.
AND    (R3 <> 'D' OR R1 = 'D')
AND    (R3 <> 'T' OR R1 <> 'D')
```

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (10)

- Abgegebene Lösung mit gaaanz vielen Klammern:

```
SELECT *  
FROM MOEGlich  
WHERE (( NOT ( R1 = 'D' ) ) OR ( R1 = ' ' ) )  
AND   (( NOT ( R1 = 'T' ) ) OR ( NOT( R1 = ' ' ) ))  
AND   (( NOT ( R2 = 'D' ) ) OR ( R1 = 'T' ) )  
AND   (( NOT ( R2 = 'T' ) ) OR ( NOT( R1 = 'T' ) ))  
AND   (( NOT ( R3 = 'D' ) ) OR ( R1 = 'D' ) )  
AND   (( NOT ( R3 = 'T' ) ) OR ( NOT( R1 = 'D' ) ))
```

- Man muss die Prioritäten der logischen Operatoren lernen:
NOT bindet am stärksten, dann **AND** und zum Schluss **OR**.

So, wie es ist, hat man Schwierigkeiten, die jeweils zugehörigen Klammern zu finden.

5. Übungsblatt, Aufgabe 2: Logik-Rätsel (11)

- Die Lösung ist diese Zeile:

R1	R2	R3
T	D	

- Man kann es sich auch so überlegen:
 - Die Dame kann nicht in Raum 1 sein, weil das Schild von Raum 1 dann nicht stimmen würde („dieser Raum ist leer“).
 - Die Inschrift von Raum 3 („Die Dame ist in Raum 1“) ist sicher falsch. Die Dame kann dort auch nicht sein.
 - Also muss die Dame in Raum 2 sein. Dann ist das Schild wahr und der Tiger in Raum 1.

5. Übungsblatt, Aufgabe 3: Erster Join (1)

- Diese Aufgabe bezieht sich auf die schon bekannte Datenbank mit Informationen über Musik-CDs (klassische Musik). Sie finden Sie im **Adminer**, Schema „**komponist_public**“. Die Webadresse ist:

`https://dbs.informatik.uni-halle.de/edb?
pgsql=db&username=student_gast&
db=postgres&ns=komponist_public`

- Sie benötigen die Tabellen
 - **KOMPONIST**(**KNR**, **NAME**, **VORNAME**[°], **GEBOREN**[°], **GESTORBEN**[°])
 - **STUECK**(**SNR**, **KNR**[°]→**KOMPONIST**, **TITEL**, **TONART**[°], **OPUS**[°])

5. Übungsblatt, Aufgabe 3: Erster Join (2)

- Sie wollen wissen, welche Stücke in der Tonart „F-dur“ von Johann Sebastian Bach und Georg Friedrich Händel in der Datenbank sind.
- Geben Sie Name und Vorname des Komponisten sowie den Titel des Stückes aus.
- Nennen Sie die Spalten der Ausgabe bitte „**Nachname**“, „**Vorname**“ und „**Titel**“ (auch in genau dieser Groß-/Kleinschreibung).
- Tipp: Sie brauchen über beiden Tabellen jeweils eine Tupelvariable.
- Das erwartete Ergebnis steht auf der nächsten Folie.

5. Übungsblatt, Aufgabe 3: Erster Join (3)

Nachname	Vorname	Titel
Händel	Georg Friedrich	Wassermusik, Suite in F-dur
Händel	Georg Friedrich	Concerto grosso op.3 Nr.4
Händel	Georg Friedrich	Concerto grosso op.6 Nr.2
Händel	Georg Friedrich	Concerto grosso op.6 Nr.9
Bach	Johann Sebastian	Brandenburg... Konzert Nr.1
Bach	Johann Sebastian	Brandenburg... Konzert Nr.2

5. Übungsblatt, Aufgabe 3: Erster Join (4)

Hinweise:

- Sie dürfen nur die Informationen im Aufgabentext verwenden (und ein wenig Allgemeinbildung).
 - Sie dürfen z.B. nicht die Komponistennummern der beiden Komponisten erst in der Beispieldatenbank nachschlagen, und dann in die Anfrage einsetzen.
 - Ihre Anfrage muss auch mit anderen Testzuständen funktionieren, nicht nur mit dem gegebenen Beispiel-Zustand.
- Bitte verwenden Sie möglichst nur die in der Vorlesung bereits vorgestellten Konstrukte von SQL.

5. Übungsblatt, Aufgabe 3: Erster Join (5)

- Eine Lösungs-Möglichkeit:

```
SELECT k.name AS "Nachname",  
       k.vorname, s.titel  
FROM   Komponist k, Stueck s  
WHERE  k.knr = s.knr  
AND    (k.name = 'Bach'  
        AND k.vorname = 'Johann Sebastian'  
        OR k.name = 'Händel'  
        AND k.vorname = 'Georg Friedrich')  
AND    s.tonart = 'F-dur'
```

- Wichtig: OR braucht Klammern (bindet schwächer als AND).