Einführung in Datenbanken

Übung 14: ER-Diagramme

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. Organisatorisches
2. Präsenzaufg. 11
3. ER-Modell
4. Hausaufgabe 12
5. Präsenzaufg. 12

Organisatorisches (1)

- Die Klausur (zuletzt 47 Punkte) enthält üblicherweise:
  - SQL, Logik Ankreuzaufgaben, Relationale Algebra.

    Zuletzt: SQL: $5 * 5$ Punkte, Logik: $5 * 1$ Punkte, RA: 4 Punkte.
  - Eine Aufgabe zum ER-Entwurf (ER-Diagramm zeichnen)

    Sie können es sich aussuchen: Auf Papier oder am Rechner.

    In der letzten Klausur gab es 7 Punkte für diese Aufgabe.
  - Eine Aufgabe zum logischen Entwurf.

    Übersetzung vom ER-Modell ins relationale Modell.

    Fester Algorithmus: Man kann das in ca. 2 Stunden so lernen, dass man die volle Punktzahl bekommt. Leider bekamen bisher viele Studierende bei weitem nicht die volle Punktzahl.

    In der letzten Klausur gab es 6 Punkte für diese Aufgabe.
  - Eine Bonus-Aufgabe zu funktionalen Abhängigkeiten/BCNF.

    In der letzten Klausur gab es 4 Bonuspunkte für diese Aufgabe.

Organisatorisches (2)

- „Bonuspunkte“ bedeutet, dass die Punkte nicht für die Berechnung der vollen Punktzahl (100%) mitzählen.
  - Ab 95% ist die 1.0 garantiert.
    Meist lag die Grenze bei ca. 90%.
  - Ab 60% ist das Bestehen garantiert.
    Meist lag die Grenze bei ca. 50%.
- Neuer Aufgabentyp mit `CREATE TABLE` wäre möglich.
- Es gibt nur noch einen Vorlesungstermin.

  Eine Vorlesung ist durch den Feiertag „Heilige Drei Könige“ ausgefallen.
- Voraussichtlich gibt es eine zusätzliche Video-Aufnahme zum Thema „Relationale Normalformen“ (für die BCNF-Bonusaufgabe).

Inhalt

1 Organisatorisches

2 Präsenzaufg. 11

3 ER-Modell

4 Hausaufgabe 12

5 Präsenzaufg. 12

Präsenzaufgabe: Vereinigung vs. Verbund (1)

- Zwei alternative Darstellungen der Punkte für HA und Zwischen- und Endklausur der Studenten sind:

Resultate_1

STUDENT	H	Z	E
Jim Ford	95	60	75
Ann Lloyd	80	90	95

Resultate_2

STUDENT	ATYP	PROZENT
Jim Ford	H	95
Jim Ford	Z	60
Jim Ford	E	75
Ann Lloyd	H	80
Ann Lloyd	Z	90
Ann Lloyd	E	95

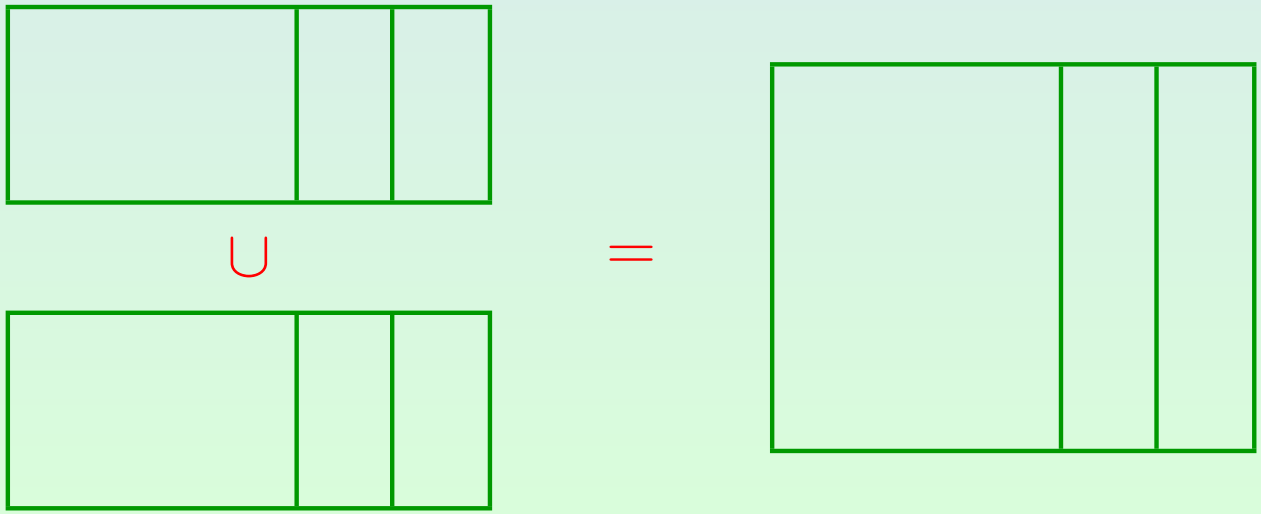
- Für die Umrechnung von `Resultate_1` in `Resultate_2` braucht man: ☐ Vereinigung ☐ Verbund

  D.h. eine Anfrage, die nur `Resultate_1` verwendet, und `Resultate_2` liefert.

Präsenzaufgabe: Vereinigung vs. Verbund (2)

Lösung:

- `Resultate_1(STUDENT, H, Z, E)`
- `Resultate_2(STUDENT, ATYP, PROZENT)`
- In der Ausgabe-Spalte `PROZENT` sollen Werte stehen, die in der Eingabe in mehreren verschiedenen Spalten (`H`, `Z`, `E`) stehen. Das fordert `UNION`:



Präsenzaufgabe: Vereinigung vs. Verbund (3)

Lösung in SQL:

- Berechung von `Resultate_2` aus `Resultate_1`:

```
SELECT STUDENT, 'H' AS ATYP, H AS PROZENT
FROM   Resultate_1
UNION  ALL
SELECT STUDENT, 'Z' AS ATYP, Z AS PROZENT
FROM   Resultate_1
UNION  ALL
SELECT STUDENT, 'E' AS ATYP, E AS PROZENT
FROM   Resultate_1
```

- So werden aus jeder Zeile von `Resultate_1` drei Zeilen im Ergebnis.

Präsenzaufgabe: Vereinigung vs. Verbund (4)

Lösung in Relationaler Algebra:

- Berechung von `Resultate_2` aus `Resultate_1`:

```
pi STUDENT, ATYP<-'H', PROZENT<-H (Resultate_1)
union
pi STUDENT, ATYP<-'Z', PROZENT<-Z (Resultate_1)
union
pi STUDENT, ATYP<-'E', PROZENT<-E (Resultate_1)
```

- In mathematischer Schreibweise:

$$\pi_{\texttt{STUDENT,ATYP}\leftarrow\texttt{'H'},\texttt{PROZENT}\leftarrow\texttt{H}}(\texttt{Resultate_1})$$
$$\cup\ \pi_{\texttt{STUDENT,ATYP}\leftarrow\texttt{'Z'},\texttt{PROZENT}\leftarrow\texttt{Z}}(\texttt{Resultate_1})$$
$$\cup\ \pi_{\texttt{STUDENT,ATYP}\leftarrow\texttt{'E'},\texttt{PROZENT}\leftarrow\texttt{E}}(\texttt{Resultate_1})$$

Präsenzaufgabe: Vereinigung vs. Verbund (5)

Zum Ausprobieren:

- Link zu RelaX (für Anfragen in relationaler Algebra):
  [http://dbis-uibk.github.io/relax/calc/gist/e0a76d711d449d67914a2b188b7ea137]
- Link zum Adminer (für SQL-Anfragen in PostgreSQL):
  [https://dbs.informatik.uni-halle.de/edb?pgsql=db&username=student_gast&db=postgres&ns=resultate_public]

Umgekehrte Richtung (1)

- Zwei Darstellungen der Gesamtleistung für Hausaufgaben, Zwischenklausur, Endklausur:

Resultate_1

STUDENT	H	Z	E
Ann Lloyd	80	90	95
Jim Ford	95	60	75

Resultate_2

STUDENT	ATYP	PROZENT
Ann Lloyd	H	80
Ann Lloyd	Z	90
Ann Lloyd	E	95
Jim Ford	H	95
Jim Ford	Z	60
Jim Ford	E	75

- Nun die umgekehrte Transformation: `Resultate_2` ist gegeben, und wir wollen `Resultate_1` berechnen.

Umgekehrte Richtung (2)

- `Resultate_1(STUDENT, H, Z, E)`
- `Resultate_2(STUDENT, ATYP, PROZENT)`
- Aufg.: Umrechnung von `Resultate_2` in `Resultate_2`.
- Man braucht drei Zeilen von `Resultate_2` (für die drei verschiedenen `ATYP`-Werte), um eine Zeile von `Resultate_1` zu produzieren. Das fordert einen Join:

- Der Join macht die Tabellen breiter (mehr Spalten), die Vereinigung macht sie höher (mehr Zeilen).

Umgekehrte Richtung (3)

- Berechnung von `Resultate_1` aus `Resultate_2`:

```
SELECT H.STUDENT, H.PROZENT AS H,
       Z.PROZENT AS Z, E.PROZENT AS E
FROM   Resultate_2 H,
       Resultate_2 Z,
       Resultate_2 E
WHERE  H.STUDENT = Z.STUDENT
AND    Z.STUDENT = E.STUDENT
AND    H.ATYP = 'H'
AND    Z.ATYP = 'Z'
AND    E.ATYP = 'E'
```

- Also Selbstverbund mit jeweils drei Tupeln von `Resultate_2` (für die drei Aufgabentypen).

Umgekehrte Richtung (4)

- Lösung mit Teilanfragen:

```
WITH
H AS    (SELECT student, prozent as H
         FROM   Resultate_2
         WHERE  atyp = 'H'),
Z AS    (SELECT student, prozent as Z
         FROM   Resultate_2
         WHERE  atyp = 'Z'),
E AS    (SELECT student, prozent as E
         FROM   Resultate_2
         WHERE  atyp = 'E')
SELECT * FROM H NATURAL JOIN Z NATURAL JOIN E
```

Umgekehrte Richtung (5)

- Mit Unteranfragen unter SELECT:

```
SELECT h.student, h.prozent AS H,
       (SELECT z.prozent
        FROM   Resultate_2 z
        WHERE  z.atyp = 'Z'
        AND    z.student = h.student) AS Z,
       (SELECT e.prozent
        FROM   Resultate_2 e
        WHERE  e.atyp = 'E'
        AND    e.student = h.student) AS E
FROM   Resultate_2 h
WHERE  h.atyp = 'H'
```

- Im Beispiel-Zustand ist das Ergebnis gleich, aber ist die Anfrage äquivalent (Ja/Nein-Umfrage)?

Inhalt

1 Organisatorisches

2 Präsenzaufg. 11

3 ER-Modell

4 Hausaufgabe 12

5 Präsenzaufg. 12

Phasen des DB-Entwurfs (1)

- Was sind die drei klassischen Phasen des DB-Entwurfs?

A. ER-Entwurf,
relationaler Entwurf,
`CREATE TABLE` Statements.

B. Konzeptioneller Entwurf,
Logischer Entwurf,
Physischer Entwurf

C. Anforderungs-Analyse,
Entwurf,
Implementierung

D. Inception,
Elaboration,
Construction

Phasen des DB-Entwurfs (2)

- Man sollte Datenbank-Schemata (mit mehr als zwei Tabellen) zunächst im ER-Modell entwerfen, und dann ins relationale Modell übersetzen.
- Welchen Grund finden Sie am überzeugensten?

A. Das ER-Modell hat eine übersichtliche graphische Notation.

   Auch etwas kompakter als eine graphische Darstellung eines relationalen Schemas mit Kanten für Fremdschlüssel.

B. Man ist nicht gleich auf ein bestimmtes DBMS festgelegt.

   Nicht einmal auf ein relationales DBMS. Blick auf reale Welt, nicht DB.

C. Es gibt Konstrukte, die das relationale Modell nicht hat.

   Z.B. Subklassen, aber erst in DB-Entwurfsvorlesung im Master.

D. Ein objektorientierter Entwurf ist heute Standard.

E. Mich überzeugt keiner dieser Gründe.

UML Klassendiagramme (1)

- Kennen Sie UML Klassendiagramme?
  A. Ja, gut.
  B. Es war dran, aber ich erinnere mich nur verschwommen.
  C. Ich habe davon gehört, aber es war nicht oder nur extrem kurz dran.
  D. Nie gehört.
- Kennen Sie ER-Diagramme?
  A. Ja, gut.
  B. Etwas.
  C. Nein.
- Wenn ja, woher? Welche Notation?

UML Klassendiagramme (2)

ER-Diagramm in Barker-Notation:

UML Klassendiagramm:

UML Klassendiagramme (3)

- Was haben UML Klassendiagramme, was das ER-Modell nicht hat?
  A. Attribute
  B. Methoden
  C. Schlüssel
  D. Relationships mit Kardinalitätsangaben (ggf. umbenannt)
- Was hat das ER-Modell, was UML Klassendiagramme (ohne Erweiterungen) nicht haben?
  A. Attribute
  B. Methoden
  C. Schlüssel

Zeichnen von ER-Diagrammen

- Handschriftlich, abfotographieren/scannen.
- Graphviz: Open Source Graph Visualization Software
  - Homepage: [http://www.graphviz.org/]
  - DOT Language: [https://graphviz.org/doc/info/lang.html]
  - User's Manual: [https://www.graphviz.org/pdf/dotguide.pdf]
  - Online Schnittstelle z.B.: [http://graphviz.it/] [https://dreampuf.github.io/GraphvizOnline/]
  - Erstes Beispiel: [http://www.informatik.uni-halle.de/~brass/db21/lab/dozent.dot]
  - Kommandozeile: `dot -Tpdf -o dozent.pdf dozent.dot`
- Oracle SQL Developer Data Modeler: s.u.

DOT Spezifikation für Graphviz (1)

```
// Gerichteter Graph ohne Duplikat-Kanten:
strict digraph {
nodesep=1
splines=false
rankdir=LR

// Knoten fuer Entity-Typ mit Beschriftung in HTML:
DOZENT [shape=rect style=rounded label=<
<table border="0" cellborder="0" cellpadding="0">
<tr><td align="center" cellpadding="5" colspan="2"
>DOZENT</td></tr>
<tr><td>#</td><td align="left">Name</td></tr>
<tr><td>*</td><td align="left">Tel</td></tr>
</table>
>]; // Semikolon ist optional
```

DOT Spezifikation für Graphviz (2)

```
VORLESUNG [shape=rect style=rounded label=<
<table border="0" cellborder="0" cellpadding="0">
<tr><td align="center" cellpadding="5" colspan="2"
>VORLESUNG</td></tr>
<tr><td>#</td><td align="left">Nr</td></tr>
<tr><td>*</td><td align="left">Titel</td></tr>
</table>
>];

//Subgraph (begrenzt Default-Setzung):
{   // Default Attribut-Setzung für Knoten:
    node[shape=point style=invis height=0];
    // Knoten in der Mitte des Relationships:
    mitte;
}
```

DOT Spezifikation für Graphviz (3)

```
// Positionierung (fuer komplexe Diagramme):
// { rank=same; DOZENT mitte VORLESUNG }
// Würde zu Positionierung übereinander führen.

// Kanten für Relationship:
DOZENT -> mitte [arrowhead=none style=dashed
                 label="hält"]
mitte -> VORLESUNG [arrowhead=crow style=solid
                    label="wird gehalten\nvon"]

}
```

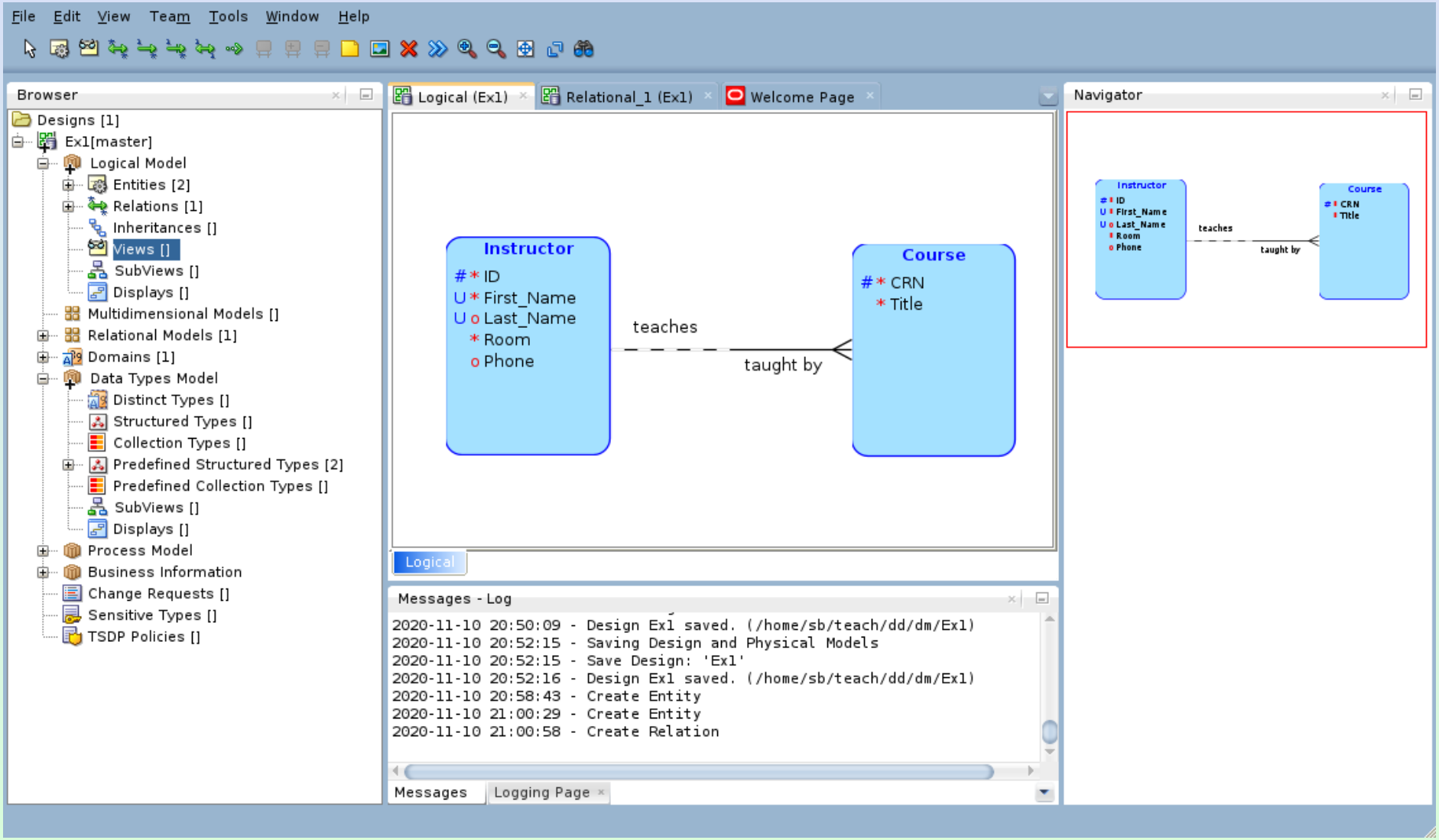
Oracle SQL Developer Data Modeler (1)

- Kostenloses Werkzeug zum Datenbankentwurf:
  [https://www.oracle.com/de/database/technologies/appdev/datamodeler.html]
- Wird in der Vorlesung „Datenbanken IIA: DB-Entwurf“ für das Projekt verwendet.
- Benötigt eventuell etwas Einarbeitung.

  Z.B. gibt es keinen Menupunkt „Neues Projekt“. Wenn Sie das Werkzeug starten, gibt es schon ein leeres Projekt. Durch Abspeichern geben Sie dem Projekt einen Namen.
- Das Werkzeug kann viel mehr, als nur ER-Diagramme in der Barker-Notation zeichnen.

  Z.B. auch die Übersetzung in das relationale Modell. Es können `CREATE TABLE` Anweisungen für verschiedene DBMS erzeugt werden, leider nicht für PostgreSQL. Eine gewisse Kontrolle und Nachbearbeitung ist also nötig.

Oracle SQL Developer Data Modeler (2)



Zeichnen von Diagrammen: Weitere Möglichkeiten

- [draw.io] → [https://app.diagrams.net/]

  Schauen Sie sich auch die geschwungenen Kurven in der Palette für ER-Diagramme an. Dort verbergen sich passende Linien für Relationships.

- yEd: [https://www.yworks.com/products/yed]

- LaTeX pur oder mit Ti*k*Z

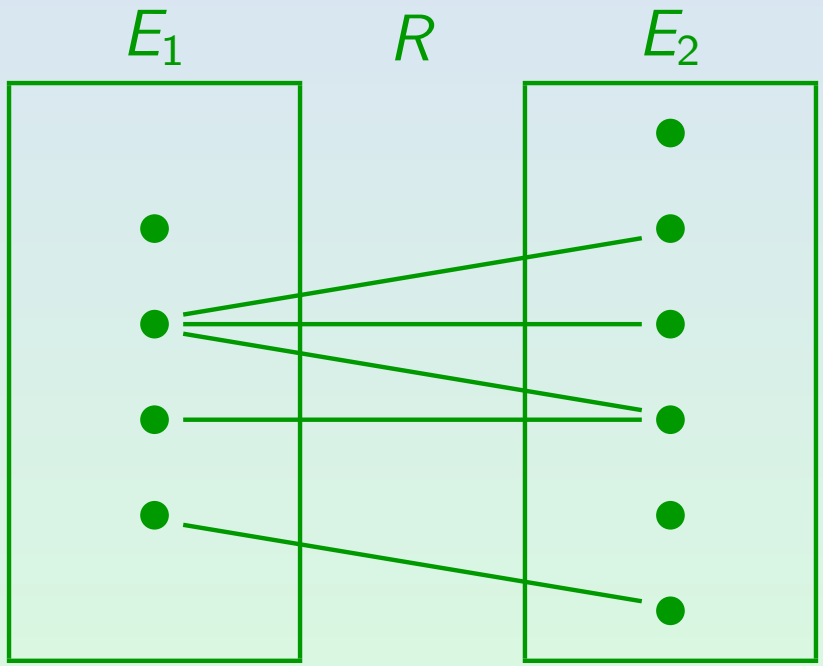
  TikZ&PGF Manual: [https://texdoc.org/serve/tikz/0]
  Claudio Fiandrino: „Drawing ER diagrams with TikZ“ (leider andere Notation):
  [https://www.guitex.org/home/images/ArsTeXnica/AT015/drawing-ER-diagrams-with-TikZ.pdf]
  Beispiel: [https://texample.net/tikz/examples/entity-relationship-diagram/]

- Dia: [http://dia-installer.de/]

  [https://sourceforge.net/projects/dia-installer/]

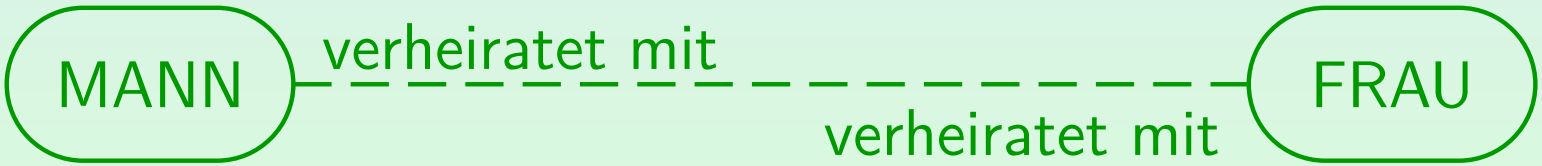
Kardinalitäten: Motivation (1)

Allgemeines Relationship:



Kardinalitäten: Motivation (2)

- Normalerweise gibt es keine Einschränkung, wie oft ein Entity an einem Relationship teilnimmt.
- Ein Entity kann mit einem, mit mehreren oder mit keinem Entity eines anderen Typs verbunden sein.
- Oft weiß man jedoch, wie viele E_2-Entities zu einem E_1-Entity in Beziehung stehen:



Kardinalitäten: Maximum (1)

Klassifikation über Maximalzahl verbundener Entities:

- Eins-zu-eins-Beziehungen („1:1“):
  In beiden Richtungen ist ein Entity mit höchstens einem Entity des anderen Typs verbunden.
- Eins-zu-viele-Beziehungen („1:n“):
  In einer Richtung kann es mehrere Entities geben, in der anderen Richtung nur eins.
- Viele-zu-viele-Beziehungen („n:m“):
  In beiden Richtungen kann ein Entity mit mehreren Entities des anderen Typs in Beziehung stehen.

Kardinalitäten: Maximum (2)

- In der Barker-Notation wird die Klassifizierung der Relationships in „1:1“, „1:n“ und „n:m“ über sogenannte „Krähenfüße“ an den Enden ausgedrückt:

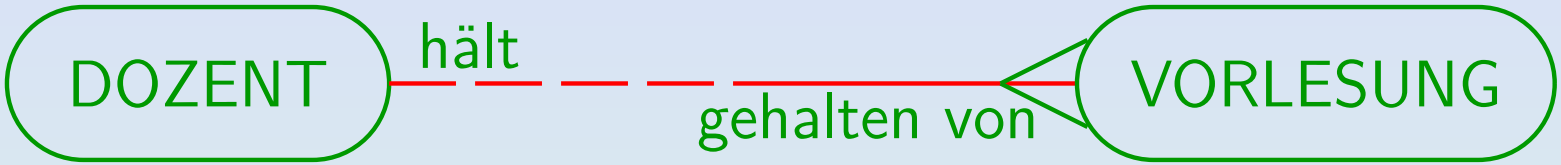
- Dies ist eine „Eins-zu-viele Beziehung“:
  - Ein Dozent kann viele Vorlesungen halten.
  - Jede Vorlesung wird nur von einem Dozenten gehalten (weil auf Dozentenseite kein Krähenfuß).
- Der Krähenfuß drückt intuitiv die Aufspaltung aus.

  Die Integritätsbedingung (Einschränkung) ist jeweils das Fehlen des Krähenfußes. Die viele-zu-viele Beziehung ist der allgemeinste Fall.

Kardinalitäten: Minimum (1)

- Umgekehrt kann man auch die minimale Anzahl von Entities des anderen Typs angeben, zu dem ein Entity eine Beziehung haben muss.
- Die Barker-Notation unterstützt nur 0 und 1:
  - Wenn Entities des Typs E_1 nicht an der Beziehung teilnehmen müssen, wird die Hälfte der Verbindungslinie auf der Seite von E_1 gestrichelt.
  - Wenn Entities vom Typ E_1 zu mindestens einem Entity des Typs E_2 in Beziehung stehen müssen, wird die Linie auf der E_1-Seite durchgezogen.

Kardinalitäten: Minimum (2)



- Ein Dozent muss hier nicht unbedingt eine Vorlesung halten.

  Die gestrichelte Linie auf seiner Hälfte zeigt, dass die Teilnahme von „DOZENT“-Entities an der Beziehung optional ist.

- Aber eine Vorlesung muss immer eine Beziehung zu einem Dozenten haben.

  Die durchgezogene Hälfte auf der Vorlesungs-Seite zeigt, dass die Teilnahme von „VORLESUNG“-Entities an der Beziehung verpflichtend ist (ob das in der Realität immer so ist, ist eine andere Frage).

Kardinalitäten: Minimum (3)

- Selbstverständlich können auch beide Hälften der Verbindungslinie gestrichelt oder beide durchgezogen sein:

- Hier ist nicht verlangt, dass ein Student mindestens eine Vorlesung hört. Es ist auch nicht verlangt, dass eine Vorlesung mindestens einen Teilnehmer hat.

  Eine Vorlesung ohne Teilnehmer wird gestrichen, wenn das Semester begonnen hat. Aber wenn sie angekündigt wird, ist völlig normal, dass sie zunächst keine Teilnehmer hat (temporäre Situationen bedenken!).

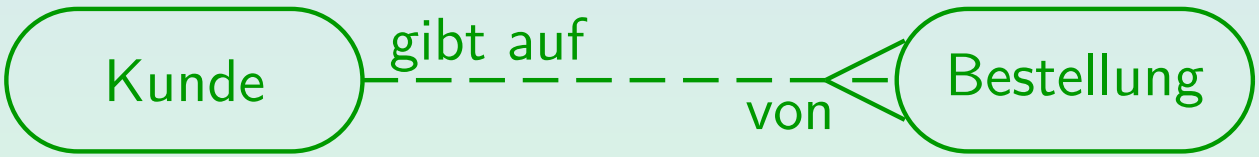
- Die gestrichelte Linie ist der allgemeinste Fall (keine Einschränkung).

Aufgabe zu Kardinalitäten (1)

- Legen Sie die Kardinalitäten für eine Beziehung zwischen „Bestellung“ und „Kunde“ fest.

  Es gibt keine Sammelbestellungen. Jede Bestellung ist von nur einem Kunden. Kunden können mehrere Bestellungen aufgeben. Neben normalen Kunden enthält die DB auch solche, die noch nichts bestellt haben, sondern nur einen Produktkatalog erhalten haben. Fragen Sie, wenn noch etwas unklar ist.

A.



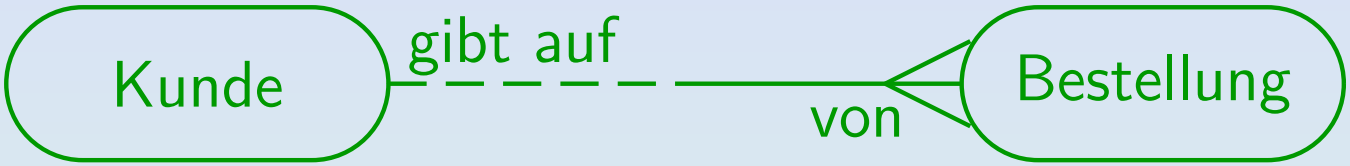
B.

C.

D.

Aufgabe zu Kardinalitäten (2)

- Richtig war Lösung C:



- Ein Kunde kann mehrere Bestellungen aufgeben.
- Ein Kunde muss keine Bestellungen aufgeben.

  Die DB enthält auch „Kunden“, die nur einen Katalog bestellt haben.

- Es gibt keine Sammelbestellungen, d.h. jede Bestellung ist von höchstens einem Kunden.
- Es gibt keine anonymen Bestellungen, d.h. jede Bestellung ist von mindestens einem Kunden.

  Zusammen also ist jede Bestellung von genau einem Kunden.

Aufgabe zum ER-Entwurf

Definieren Sie ein ER-Schema (Diagramm) für die folgende Anwendung:

- Informationen über Forscher auf dem Gebiet „Datenbanken“ sollen gespeichert werden.
- Für jeden Forscher wird Nachname, Vorname, E-Mail-Adresse und Homepage (URL) benötigt.
- Auch der derzeitige Arbeitgeber wird benötigt (Annahme: alle Forscher arbeiten an einer Uni).
- Für jede Universität sollen der Name, die URL und das Land gespeichert werden.

Inhalt

1 Organisatorisches

2 Präsenzaufg. 11

3 ER-Modell

4 Hausaufgabe 12

5 Präsenzaufg. 12

Schema der Vitamin-Datenbank

- Schema vit_public im Adminer:
  - Stoff_Kategorie(Kat, Bezeichnung, Sort_Nr)

    Z.B. Vitamin, Mineralstoff, Spurenelement, ...
  - Stoff(Vit, Einheit, Tagesdosis°, Kat → Stoff_Kategorie)
  - Praeparat(Pid, Name, Hersteller, PZN°, Anz°, Einheit°, Tagesdosis°, Gewicht°, Preis°, glutenfrei°, lactosefrei°)
  - Inhalt(Pid → Praeparat, Vit → Stoff, Menge, Prozent°, Anmerkung°)
  - Zutat(Pid → Praeparat, Seq, Name, Anmerkung°)

Links:

- Link zu RelaX (für Anfragen in relationaler Algebra):
  [http://dbis-uibk.github.io/relax/calc/gist/aeb39915904003f10caa6e53bbc5f19a]

  Falls Sie die Daten im für RelaX nötigen Format sehen wollen (das ist aber nur interessant, wenn Sie eigene Datenbanken als „Gist“ anlegen wollen):
  [https://gist.github.com/aeb39915904003f10caa6e53bbc5f19a]
  Die Daten sind in einer Art vereinfachtem CSV-Format: Wenn ein Spaltenwert nicht nur ein einzelnes Wort oder eine einzelne Zahl ist, muss er in `"..."` geschrieben werden.

- Link zum Adminer (für SQL-Anfragen in PostgreSQL):
  [https://dbs.informatik.uni-halle.de/edb?pgsql=db&username=student_gast&db=postgres&ns=vit_public]

Hausaufgabe 12.1: Einfache RA-Anfrage (1)

- Schreiben Sie folgende Anfrage in relationaler Algebra (in der Syntax von RelaX):
  Welche Präparate enthalten mindestens 7 mg Zink?

  Sie können voraussetzen, dass die Einheit für Zink „mg“ ist, und brauchen das nicht zu testen. Geben Sie Name und Hersteller des Präparates und die Menge an Zink aus.

- Das erwartete Ergebnis ist:

Name	Hersteller	Menge
'A-Z Komplett'	'Abtei'	10
'Kardiodrink'	'MensSana'	10

RelaX zeigt den Relationennamen mit an, also `Praeparat.Name`, `Praeparat.Hersteller` und `Inhalt.Menge`.

Hausaufgabe 12.1: Einfache RA-Anfrage (2)

Lösung:

- Diese Anfrage kann mit dem Muster „Join-Selektion-Projektion“ gelöst werden:

```
pi Name, Hersteller, Menge(
    sigma Vit = 'Zink' and Menge >= 7(
        Praeparat join Inhalt))
```

$$\pi_{\text{Name,Hersteller,Menge}}\Big(\sigma_{\text{Vit}=\text{'Zink'} \wedge \text{Menge} \geq 7}\big(\text{Praeparat} \bowtie \text{Inhalt}\big)\Big)$$

- Es wird hier am einfachsten ein natürlicher Verbund verwendet („`join`“ ohne Bedingung).

  Praktischerweise heißt die Spalte für den Join in beiden Tabellen `Pid` (Schlüssel in `Präparat` und Fremdschlüssel in `Inhalt`). Außerdem ist `Pid` die einzige gemeinsame Spalte von `Praeparat` und `Inhalt`.

Hausaufgabe 12.2: Selbstverbund in RA (1)

- Schreiben Sie folgende Anfrage in relationaler Algebra (in der Syntax von RelaX):
  Geben Sie alle Präparate aus, die sowohl Lutein enthalten, als auch einen Inhaltsstoff mit einer Anmerkung, in der „Beta-Carotin“ als Substring vorkommt.

  > Vitamin A ist öfters teilweise als Provitamin Beta-Carotin enthalten.
  > Das steht dann in der Anmerkung. Sie brauchen `Vit = 'A'` nicht zu testen.
  > Die beiden Stoffe sollen sich gegenseitig in der Aufnahme behindern.

- Geben Sie Name und Hersteller des Präparates aus.

- Das erwartete Ergebnis ist:

Praeparat.Name	Praeparat.Hersteller
'Kardiodrink'	'MensSana'

Hausaufgabe 12.2: Selbstverbund in RA (2)

Hinweise zu RelaX:

- RelaX versteht auch `LIKE`-Bedingungen.
- Sie könnten für diese Anfrage z.B. `intersect` verwenden.
- Alternativ können Sie mit

  ```
  (rho A Inhalt) x (rho B Inhalt)
  ```

  ein kartesisches Produkt von zwei Kopien der Relation erzeugen (die Klammern sind eigentlich nicht nötig).
- Statt eines kartesischen Produktes können Sie auch gern gleich einen Join verwenden (die Klammern sind wieder überflüssig):

  ```
  (rho A Inhalt) join A.Pid=B.Pid (rho B Inhalt)
  ```

Hausaufgabe 12.2: Selbstverbund in RA (3)

Lösung:

- Man kann nach dem bekannten Muster zwei Teilanfragen (für jede der beiden Bedingungen) schreiben, und die Schnittmenge der Ergebnisse berechnen:

```
pi Name, Hersteller(sigma Vit = 'Lutein'(
        Praeparat join Inhalt))
intersect
pi Name, Hersteller(
    sigma Anmerkung LIKE '%Beta-Carotin%'(
        Praeparat join Inhalt))
```

$$\pi_{\text{Name,Hersteller}}\left(\sigma_{\text{Vit='Lutein'}}\left(\text{Praeparat} \bowtie \text{Inhalt}\right)\right) \cap$$
$$\pi_{\text{Name,Hersteller}}\left(\sigma_{\text{Anmerkung LIKE '\%Beta-Carotin\%'}}\left(\text{Praeparat} \bowtie \text{Inhalt}\right)\right)$$

Hausaufgabe 12.2: Selbstverbund in RA (4)

Lösung, Forts.:

- Alternativ kann man den Schnitt schon auf den `Pids` aus `Inhalt` berechnen, und erst danach mit `Praeparat` joinen:

```
pi Name, Hersteller(Praeparat join
    (pi Pid(sigma Vit = 'Lutein'(Inhalt))
     intersect
     pi Pid(
         sigma Anmerkung LIKE '%Beta-Carotin%'(
             Inhalt))))
```

Das wäre aus Performance-Gründen wohl besser (Performance ist in dieser Vorlesung nicht sehr wichtig). Eine kürze Lösung wäre natürlich auch besser (und das ist dieser Vorlesung schon wichtig). Obwohl man hier den doppelten Join einspart, ist aber eine zusätzliche Projektion auf die `Pid` vor dem Schnitt nötig (die anderen Spalten würden ja verschiedene Werte enthalten).

Hausaufgabe 12.2: Selbstverbund in RA (5)

Lösung, Forts.:

- Auch eine Lösung recht ähnlich zu SQL wäre möglich (mit einem kartesischen Produkt von zwei Kopien der `Inhalt`-Relation):

```
pi P.Name, P.Hersteller(
  sigma A.Vit = 'Lutein' and
        B.Anmerkung LIKE '%Beta-Carotin%' and
        A.Pid = B.Pid and
        P.Pid = A.Pid
    (rho P Praeparat x rho A Inhalt
                   x rho B Inhalt))
```

Auch hier könnte man den Join mit `Praeparat` erst später machen, nachdem die Bedingung für die beiden Kopien von `Inhalt` ausgewertet ist. Statt einem kartesischen Produkt könnte man auch einen Join mit Bedingung verwenden.

Hausaufgabe 12.2: Selbstverbund in RA (6)

Anmerkung:

- Wenn die linke und die rechte Tabelle die gleichen Spalten haben, entartet der natürliche Verbund zur Schnittmengenbildung $\cap$.

  Bei RelaX gilt das ohne Berücksichtigung der Relationen-Präfixe der Spalten-Namen (die dürfen unterschiedlich sein). Nach dem Skript wären die Präfixe dagegen Teil des Spaltennamens.

- Wenn linke und rechte Tabelle disjunkte Spaltenmengen haben, entartet der natürliche Verbund zum kartesischen Produkt $\times$.

  Es gibt in diesem Fall ja keine Spalte(n) für die Verbundbedingung.

- Beides ist kein Syntaxfehler, aber ein Stilproblem.

  Man sollte immer die Operation schreiben, die möglichst deutlich ausdrückt, was tatsächlich passiert.

Hausaufgabe 12.3: Selbstverbund in SQL (1)

- Schreiben Sie die Anfrage von Aufgabe 2 auch in SQL.
- Drucken Sie also Name und Hersteller von allen Präparaten, die sowohl Lutein enthalten, als auch einen Inhaltsstoff mit einer Bemerkung, die den Substring „Beta-Carotin“ enthält.
- Das erwartete Ergebnis ist:

Name	Hersteller
Kardiodrink	MensSana

Hausaufgabe 12.3: Selbstverbund in SQL (2)

Lösung:

- Man braucht drei Tupelvariablen:

```
SELECT P.Name, P.Hersteller
FROM   Praeparat P, Inhalt A, Inhalt B
WHERE  P.Pid = A.Pid
AND    A.PID = B.Pid
AND    A.Vit = 'Lutein'
AND    B.Anmerkung LIKE '%Beta-Carotin%'
```

Hausaufgabe 12.4: Mengenoperationen in RA (1)

- Schreiben Sie folgende Anfrage in relationaler Algebra (in der Syntax von RelaX):
  Geben Sie für alle Präparate Name, Hersteller und den Gehalt an Zink aus.
- Falls das Präparat kein Zink enthält (d.h. kein passender Eintrag in `Inhalt`), soll der Wert 0 ausgegeben werden.
- Das erwartete Ergebnis steht auf der nächsten Folie.

  Der genaue Name der letzten Spalte ist nicht vorgeschrieben. Die Präfixe der Spaltennamen sind in dieser Darstellung weggelassen. RelaX zeigt aber auch die Relationennamen.

Hausaufgabe 12.4: Mengenoperationen in RA (2)

Name	Hersteller	Menge
'A-Z Vital'	'gesundleben'	5
'Centrum'	'Pfizer'	5
'A-Z Komplett'	'Abtei'	10
'A-Z Complete'	'Doppelherz aktiv'	5
'Kardiodrink'	'MensSana'	10
'Mineraldrink'	'MensSana'	5
'Magnesium Verla 300'	'Verla-Pharm ...'	0

Hausaufgabe 12.4: Mengenoperationen in RA (3)

Hinweise:

- Nützliche Operationen könnten z.B. die Vereinigung `union` und die Mengendifferenz `-` sein.
- Bei `union` müssen auch die Spaltennamen übereinstimmen (nicht aber der Relationen-Präfix).
- Sie können z.B. `Menge<-0` in der Projektion `pi` verwenden, um eine Spalte `Menge` mit dem konstanten Wert 0 zu erstellen.

  In RelaX ist der Pfeil eigentlich andersherum, aber es versteht auch diese Variante, die mit der Vorlesung übereinstimmt.

Hausaufgabe 12.4: Mengenoperationen in RA (4)

Hinweise, Forts.:

- Im Prinzip hätte RelaX auch die von SQL bekannte Funktion `coalesce`, aber Sie brauchen das hier vermutlich nicht.

  Falls Sie diese Funktion doch nutzen wollen, beachten Sie, dass die Spalte beim Aufruf solcher Funktionen explizit benannt werden muss mittels „<-“.

Hausaufgabe 12.4: Mengenoperationen in RA (5)

Lösung:

- Die Anfrage ist in der Hauptsache eine Vereinigung bestehend aus einem Teil für Präparate mit Zink und einem Teil für Präparate ohne Zink.
- Anschließend folgt noch ein Join mit `Praeparat`, um `Name` und `Hersteller` zu bekommen:

```
pi Name, Hersteller, Menge (Praeparat join
    (pi Pid, Menge (sigma Vit = 'Zink' Inhalt)
     union
     pi Pid, Menge<-0
        (pi Pid (Praeparat) -
            pi Pid (sigma Vit = 'Zink' Inhalt))))
```

Hausaufgabe 12.4: Mengenoperationen in RA (6)

Lösung, Forts.:

- Mit einem Outer Join lässt sich die Anfrage kürzer formulieren:

```
pi Name, Hersteller, Menge<-coalesce(Menge,0)
    (Praeparat left join
        (sigma Vit = 'Zink' Inhalt))
```

Der „`left join`“ ist wieder ein natürlicher Verbund über der gemeinsamen Spalte `Pid`. Wenn man möchte, kann man die rechte Tabelle noch auf `Pid, Menge` projizieren, aber das ist überflüssig.

Hausaufgabe 12.5: Outer Join in SQL (1)

- Schreiben Sie die gleiche Anfrage in SQL, d.h. geben Sie für alle Präparate den Gehalt an Zink aus, inklusive dem Wert 0.

Hausaufgabe 12.5: Outer Join in SQL (2)

Lösung:

- Lösung mit der Vereinigung zweier Teilanfragen (Präparate mit Zink und solche ohne):

```
SELECT P.Name, P.Hersteller, I.Menge
FROM   Praeparat P, Inhalt I
WHERE  P.Pid = I.Pid AND I.Vit = 'Zink'
UNION  ALL
SELECT P.Name, P.Hersteller, 0 AS Menge
FROM   Praeparat P
WHERE  P.Pid NOT IN (SELECT I.Pid
                     FROM   Inhalt I
                     WHERE  I.Vit = 'Zink')
```

Die Tupelvariablen in der zweiten Hälfte könnten auch anders heißen.

Sie haben nichts mit denen in der ersten Hälfte zu tun.

Hausaufgabe 12.5: Outer Join in SQL (3)

Lösung, Forts.:

- Lösung mit einem Outer Join (deutlich kürzer):

```
SELECT P.Name, P.Hersteller,
       COALESCE(I.Menge, 0) AS Menge
FROM   Praeparat P LEFT JOIN Inhalt I
       ON P.Pid = I.Pid AND I.Vit = 'Zink'
```

Man beachte, dass die Einschränkung der Tupel für `I` auf Zink vor oder spätestens während des Outer Join passieren muss. In einem `WHERE` nach dem Outer Join wäre es zu spät. Alle Präparate werden ja wohl mindestens einen Inhaltsstoff haben. Dann würde der Outer Join effektiv einfach ein normaler Join sein. Bei einem `LEFT OUTER JOIN` müssen Selektionen auf der rechten Tabelle (Einschränkung möglicher Joinpartner) vor dem Outer Join ausgeführt werden.

Inhalt

1 Organisatorisches

2 Präsenzaufg. 11

3 ER-Modell

4 Hausaufgabe 12

5 Präsenzaufg. 12

Präsenzaufgabe: ER-Diagramm

- Zeichnen Sie ein ER-Diagramm in der Notation der Vorlesung (Barker-Notation), das von der `EMP-DEPT`-DB inspiriert ist:
  - Angestellte haben eine eindeutige Nummer, außerdem ist ein Name zu speichern, ein Beruf, ein Gehalt und optional eine Provision.
  - Abteilungen haben eine eindeutige Nummer, einen Namen und einen Ort.
  - Jeder Angestellte arbeitet in höchstens einer Abteilung.

    Neue Angestellte sind eventuell noch keiner Abteilung zugeordnet.
  - Ein Abteilung kann beliebig viele Angestellte haben.

    Es ist auch möglich, dass eine geplante Abteilung noch leer ist.
- Sie dürfen deutsche oder englische Bezeichnungen wählen.
- Sie müssen Relationsships in beiden Richtungen benennen.