Datenbanksysteme I

Wintersemester 2024/25

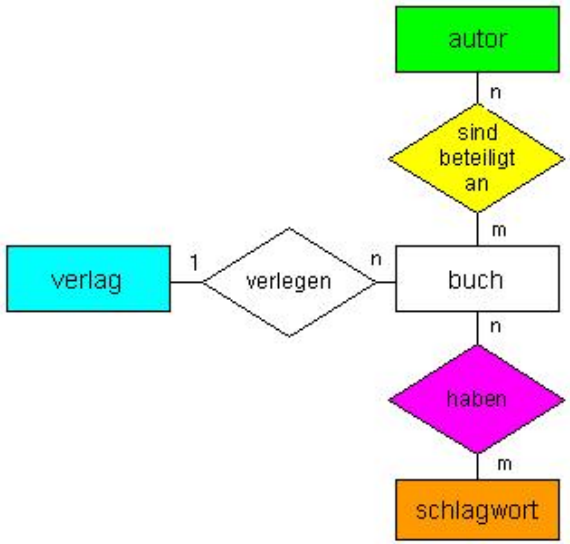
—

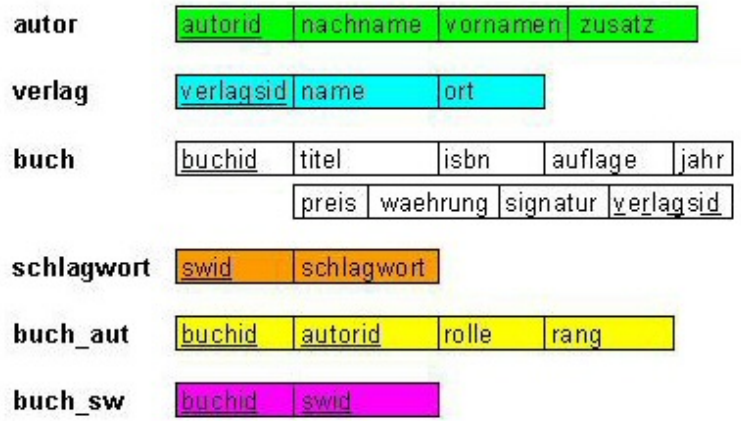
Übungsblatt 5

Abteilung Datenbanken
Institut für Informatik
Universität Leipzig

Aufgabe 1: SQL-Anfragen

Die nachfolgenden Anfragen basieren auf dem DB-Schema *Bibliothek* (VL-Skript 5-7) des im LOTS integrierten SQL-Trainers. Formulieren Sie die Anfragen in SQL und führen Sie diese im SQL-Trainer aus. Die Zahlen in Klammern geben an, wie viele Ergebnistupel zu erwarten sind. Hinweis: SQL-Anfragen können Sie im LOTS unter dem Menüpunkt SQL-Training ausführen (`https://lots.uni-leipzig.de/sql-training/`).





Buch_Aut.rolle kann sein: Herausgebers (H), Verfasser (V), Übersetzer (U), Mitarbeiter(M)
Buch_Aut.rang: Position des Autors in der Autorenliste (z.B. 1 für Erstautor)
Autor.zusatz: Namenszusatz wie „von“ oder „van“
Buch.signatur entspricht der Signatur in der IfI-Bibliothek (Stand 1998)

Aufgabe 1: SQL-Anfragen

a) Gesucht sind die Bücher (Titel, ISBN), welche das Wort *Datenbanken* im Titel enthalten. Ordnen Sie die Ausgabe alphabetisch nach dem Titel der Bücher. (64)

```
SELECT titel, isbn
FROM buch
WHERE titel LIKE '%Datenbanken%'
ORDER BY titel
```

Siehe auch `https://www.postgresql.org/docs/12/functions-matching.html`

b) Welche Bücher sind nach 1997 erschienen? Geben Sie Titel sowie Alter der Bücher aus und ordnen Sie die Ausgabe nach Alter und nachrangig nach Titel der Bücher. (72)

```
SELECT titel, EXTRACT(YEAR FROM CURRENT_DATE) - jahr AS alt
FROM buch
WHERE jahr > 1997
ORDER BY alt DESC, titel
```

Siehe auch `https://www.postgresql.org/docs/12/functions-datetime.html`

Aufgabe 1: SQL-Anfragen

c) Wie viele Autoren haben denselben Nachnamen wie ein Verlag? (61)

```
SELECT COUNT(*)
FROM verlag v, autor a
WHERE a.nachname = v.name
```

Wie viele Verlage haben denselben Namen wie ein Autor? (17)

```
SELECT COUNT(DISTINCT verlagsid)
FROM verlag v, autor a
WHERE a.nachname = v.name
```

d) Welche Bücher sind im Springer-Verlag seit 1990 erschienen? Geben Sie Titel und Jahr der Bücher sowie den Ort des Verlages aus. (890)

```
SELECT b.titel, b.jahr, v.ort
FROM buch b NATURAL JOIN verlag v
WHERE b.jahr >= 1990 AND v.name='Springer'
```

Aufgabe 1: SQL-Anfragen

e) Welche Autoren (Vorname, Nachname) haben mindestens ein Buch zum Schlagwort Datenbank verfasst? (5)

```
SELECT DISTINCT a.vornamen, a.nachname
FROM autor a
NATURAL JOIN buch_aut ba
NATURAL JOIN buch_sw bs
NATURAL JOIN schlagwort sw
WHERE sw.schlagwort = 'Datenbank' AND ba.rolle = 'v'
```

f) Wie viel Prozent der Autoren sind mit unvollständigem (d. h. mit einem mit '.' abgekürzten) oder gar keinem (d. h. NULL-wertigen) Vornamen in der Datenbank gespeichert?

```
SELECT 100 * (CAST(COUNT(*) AS double precision)/
(SELECT COUNT(*) FROM autor)) as share
FROM autor
WHERE vornamen LIKE '%.%' OR vornamen IS NULL
```

Ergebnis: 24.26%

Aufgabe 1: SQL-Anfragen

g) Geben Sie die ältesten Bücher (BuchId, Titel, Jahr) der Datenbank aus. (1)

```
SELECT buchid, titel, jahr
FROM buch
WHERE jahr=(SELECT MIN(jahr) FROM buch)
```

h) Geben Sie für jeden Buchautor seine ID und seinen Namen sowie die Anzahl der von ihm verfassten Bücher aus. Ordnen Sie die Ergebnismenge absteigend nach der Anzahl der Bücher des Autors, bei gleicher Anzahl alphabetisch nach dem Namen. (3633)

```
SELECT a.autorid, a.nachname, count(*) AS ANZAHL
FROM autor a NATURAL JOIN buch_aut ba
WHERE ba.rolle='v'
GROUP BY a.autorid, a.nachname
ORDER BY ANZAHL DESC, a.nachname
```

Aufgabe 1: SQL-Anfragen

i) Welche Verlage haben in allen Jahren von 1995 bis einschließlich 2001 wenigstens eines ihrer Bücher in die Datenbank eingebracht, vorausgesetzt, dass aus dem betreffenden Jahr überhaupt Bücher in der Datenbank sind? (1)

```
SELECT v.name
FROM verlag v
WHERE NOT EXISTS (

(SELECT DISTINCT jahr
FROM buch
WHERE jahr BETWEEN 1995 AND 2001)

EXCEPT

(SELECT DISTINCT jahr
FROM buch b
WHERE b.verlagsid=v.verlagsid)
)
```

Aufgabe 1: SQL-Anfragen

j) Erstellen Sie eine Liste aller Schlagworte und zählen Sie, wie oft jedes Schlagwort insgesamt über alle Büchern verwendet wurde. Sortieren Sie die Liste nach Anzahl und nachrangig alphabetisch nach dem Schlagwort. (844)

```
SELECT schlagwort, COALESCE(anzahl, 0) as anzahl
FROM schlagwort sw
LEFT OUTER JOIN (
    SELECT swid, COUNT(*) AS anzahl
    FROM buch_sw
    GROUP BY swid
) AS swa ON sw.swid=swa.swid
ORDER BY 2 DESC, 1
```

Siehe auch https://www.postgresql.org/docs/12/functions-conditional.html#FUNCTIONS-COALESCE-NVL-IFNULL

Aufgabe 1: SQL-Anfragen

k) Von wie vielen Autoren werden jeweils alle ihre Bücher für den Preis von 79,90 angeboten? (46)
Hinweise: Autoren ohne Bücher sollen nicht mitgezählt werden. Autorenrolle ist hier nicht zu berücksichtigen.

```
SELECT COUNT(*) FROM (
    (SELECT autorid
    FROM buch_aut NATURAL JOIN buch
    WHERE preis = 79.90)
    EXCEPT
    (SELECT autorid
    FROM buch_aut NATURAL JOIN buch
    WHERE preis <> 79.90 or preis IS NULL)
) AS c
```

Aufgabe 1: SQL-Anfragen

l) Wie heißen die Bücher mit den meisten Schlagwörtern? (5)

```
SELECT titel, buchid
FROM buch NATURAL JOIN buch_sw
GROUP BY buchid, titel
HAVING COUNT(*) = (
    SELECT MAX(swc)
    FROM (
        SELECT COUNT(swid) AS swc
        FROM buch_sw
        GROUP BY buchid
    ) AS cnt
)
```

Aufgabe 2: Relationenalgebra → SQL

Relationen:

R(a,b,c)

S(b,c,d)

a) $\pi_{a,b}(\sigma_{c=10}(R))$

```
SELECT a,b FROM R
WHERE c = 10
```

b) $R \bowtie_{R.a=S.d} S$

```
SELECT * FROM R, S
WHERE R.a = S.d
```

c) $R \ltimes S$

```
SELECT R.*
FROM R NATURAL JOIN S
```

d) $R \rtimes S$

```
SELECT *
FROM R LEFT OUTER JOIN S
```

Aufgabe 2: Relationenalgebra → SQL

Relationen:
R(a,b,c)
S(b,c,d)

e) R - S

```
SELECT *
FROM R EXCEPT S
```

oder: $R - S = \{x \mid x \in R \wedge x \notin S\}$

```
SELECT * FROM R
WHERE (R.a, R.b, R.c) NOT IN (SELECT * FROM S)
```

oder:

```
SELECT R.*
FROM R LEFT OUTER JOIN S ON (
          R.a = S.b AND R.b = S.c AND R.c = S.d)
WHERE S.b IS NULL AND S.c IS NULL AND S.d IS NULL
```

Aufgabe 2: Relationenalgebra → SQL

f) R ∩ S

$$R \cap S = \{x \mid x \in R \wedge x \in S\}$$

Relationen:

R(a,b,c)

S(b,c,d)

```
SELECT *
FROM R EXCEPT (
    SELECT *
    FROM R EXCEPT S)
```

oder:

```
SELECT DISTINCT *
FROM R INNER JOIN S
ON (R.a = S.b AND R.b = S.c AND R.c = S.d)
```

oder:

```
SELECT *
FROM R INTERSECT S
```

Aufgabe 2: Relationenalgebra → SQL

Relationen:

R(a,b,c)

S(b,c,d)

g) $R \div \pi_{b,c}(S)$

Division in SQL

Beispiel

R		
a	b	c
1	1	1
1	2	3
2	1	1
3	2	3

÷

S	
b	c
1	1
2	3

=

÷
a
1

Variante 1: Kreuzprodukt

Variante 2: Korrelierte Subquery

Variante 1: Kreuzprodukt

$$R' := A(R) - A(S)$$

$$R \div S := \pi_{R'}(R) - \pi_{R'}((\pi_{R'}(R) \times S) - R)$$

```
SELECT DISTINCT R.a
FROM R
WHERE R.a NOT IN (
    SELECT R2.a
    FROM R AS R2, S
    WHERE (R2.a, S.b, S.c) NOT IN (
        SELECT R3.a, R3.b, R3.c
        FROM R AS R3
    )
)
```

R		
a	b	c
1	1	1
1	2	3
2	1	1
3	2	3

÷

S	
b	c
1	1
2	3

=

÷
a
1

R
a
1
2
3

–

R'
a
2
3

→

÷
a
1

↑ Π_a

Π_a(R2)
a
1
2
3

X

S	
b	c
1	1
2	3

→

Π_a(R2)xS		
a	b	c
1	1	1
1	2	3
2	1	1
2	2	3
3	1	1
3	2	3

–

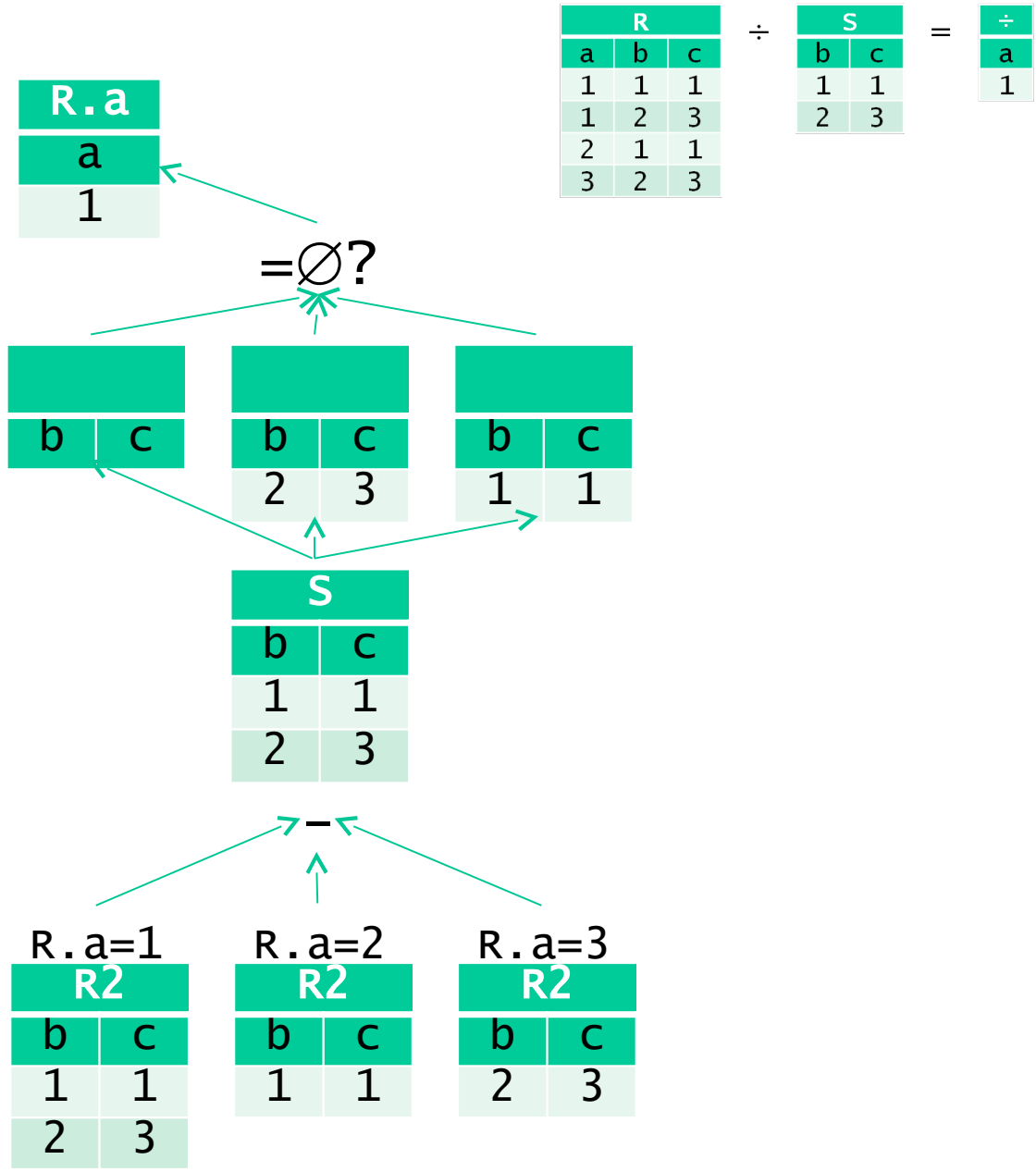
R3		
a	b	c
1	1	1
1	2	3
2	1	1
3	2	3

→

Π_a(R2)xS-R3		
a	b	c
2	2	3
3	1	1

Variante 2: Korrelierte Subquery

```
SELECT DISTINCT R.a
FROM R
WHERE NOT EXISTS (
   SELECT b, c
   FROM S
   WHERE (b,c) NOT IN (
      SELECT R2.b, R2.c
      FROM R AS R2
      WHERE R.a = R2.a
   )
)
```



Aufgabe 2) Relationenalgebra → SQL

Relationen:

R(a,b,c)

S(b,c,d)

f) $R \div \pi_{b,c}(S)$

```
SELECT      DISTINCT a
FROM        R
WHERE       b IN
            (SELECT b FROM S) AND
            c IN
            (SELECT c FROM S)
GROUP BY    a
HAVING      COUNT(*) =
            (SELECT COUNT(*) FROM S)
```