DENKSPORT MIT SQL

DOAG, 20.NOVEMBER 2019

Dr. Andrea Kennel

DR. ANDREA KENNEL

Dozentin Datenbanken

Fachcoach Projektmanagement

Fachhochschule Nordwestschweiz

Brugg/Windisch

andrea.kennel@fhnw.ch
andrea@infokennel.ch
www.infokennel.ch

Tabellen

STUD (STUDIERENDE)

ID	Name	Vorname	Strasse Nr.	PLZ	Ort	E-Mail	Geburtstag
500	Anna	Gut	Hofweg 6	3000	Bern		27.09.1997
501	Otto	Hug	Dorfstrasse 20	5200	Brugg		15.03.1985
502	Kai	Iseli	Lindenhof 5	5200	Brugg		05.12.1989
503	Lara	Meier	Markplatz 7	8000	Zürich		05.02.1996

SENO (SEMESTERNOTE)

ID	MOFU_ID	Pers_ID	Note
120	001	500	5.5
121	001	502	3.0
122	001	501	5.0
743	002	503	4.5
744	002	502	5.5

MOFU (MODULDURCHFÜHRUNG)

ID	Modul	Semester	Doz_ID	Klasse
001	dbc	2017 HS	10745	3iCbb
002	dbc	2017 HS	10745	3iCengl
039	dbc	2018 HS	15236	3iCb
040	dbc	2018 HS	10745	3iCbb
041	dbc	2018 HS	10745	3iCengl
057	eis	2018 HS	10745	4Ibb
084	webpr	2018 HS	07851	5Iv

DOZI (DOZIERENDE)

ID	Vorname	Name	Kürzel	...
10745	Andrea	Kennel	KEA	
15236	Silvia	Ackermann	ACS	
07851	Dierk	König	KOD	

Aufgaben

Erstellen Sie eine Liste, die alle Orte aufsteigend und pro Ort die Anzahl Studierenden ausgibt. (2 Punkte) Hinweis: Benutzen Sie zur Berechnung die Funktion COUNT(*).

Listen Sie alle Studierenden auf, die noch kein Modul absolviert (= keine Note) haben. (4 Punkte)

Listen Sie alle Dozierenden und Studierenden auf, die sich nicht kennen, das heisst bei denen der/die Studierende in keiner Vorlesung war. (6 Punkte)

Lösungen “Anzahl Studis je Ort”

```
SELECT plz, ort, count(*)
FROM stud
GROUP BY plz, ort
ORDER BY ort asc;

SELECT count(id) anz, ort
FROM stud
ORDER BY ort;

SELECT mofu.*, count(seno.pers_id)
FROM mofu, seno
ORDER By mofu.loc;

SELECT ort, count(ort)
FROM stud
GROUP BY ort
ORDER BY ort asc;
```

Lösungen “Anzahl Studis je Ort”

```
SELECT plz, ort, count(*)
FROM stud
GROUP BY plz, ort
ORDER BY ort asc;
```

```
SELECT count(id) anz, ort
FROM stud
ORDER BY ort;
```

```
SELECT mofu.*, count(seno.pers_id)
FROM mofu, seno
ORDER By mofu.loc;
```

```
SELECT ort, count(ort)
FROM stud
GROUP BY ort
ORDER BY ort asc;
```

Lösungen “Studis ohne Modul”

```
SELECT stud.id, stud.name
FROM stud
WHERE stud.id NOT IN
 (SELECT stud_id FROM seno);

SELECT *
FROM stud INNER JOIN seno
  ON (stud.id != seno.stud_id );

SELECT stud.*
FROM stud LEFT OUTER JOIN seno
  ON (stud.id = seno.stud_id)
WHERE seno.stud_id IS NULL;

SELECT stud.*
FROM stud, seno
WHERE stud.id = seno.stud_id (+)
  AND seno.id IS NULL;
```

```
SELECT *
FROM stud
WHERE (
  count(
    SELECT * FROM seno
    WHERE stud.id = seno.stud_id
  ) = 0
);

SELECT *
FROM stud
WHERE (
    SELECT count(*) FROM seno
    WHERE stud.id = seno.stud_id  ) = 0;

SELECT stud.id, stud.name
FROM stud
MiNUS
SELECT seno.stud_id, NULL
FROM seno;
```

Lösungen “Studis ohne Modul”

```
SELECT stud.id, stud.name
FROM stud
WHERE stud.id NOT IN
 (SELECT stud_id FROM seno);
```

```
SELECT *
FROM stud INNER JOIN seno
  ON (stud.id != seno.stud_id );
```

```
SELECT stud.*
FROM stud LEFT OUTER JOIN seno
  ON (stud.id = seno.stud_id)
WHERE seno.stud_id IS NULL;
```

```
SELECT stud.*
FROM stud, seno
WHERE stud.id = seno.stud_id (+)
  AND seno.id IS NULL;
```

```
SELECT *
FROM stud
WHERE (
  count(
    SELECT * FROM seno
    WHERE stud.id = seno.stud_id
  ) = 0
);
```

```
SELECT *
FROM stud
WHERE (
    SELECT count(*) FROM seno
    WHERE stud.id = seno.stud_id  ) = 0;
```

```
SELECT stud.id, stud.name
FROM stud
MiNUS
SELECT seno.stud_id, NULL
FROM seno;
```

Lösungen “Studi kennt Dozi nicht”

```
SELECT stud.name, stud.vorname,
  dozi.name, dozi.vorname
FROM stud INNER JOIN seno
  ON (stud.id != seno.stud_id)
  INNER JOIN mofu
  ON (seno.mofu_id != mofu.id)
  INNER JOIN dozi
  ON (mofu.dozi_id != dozi.id);
```

```
SELECT stud.name, stud.vorname,
  dozi.name, dozi.vorname
FROM stud, dozi
MINUS
SELECT stud.name, stud.vorname,
  dozi.name, dozi.vorname
FROM stud, seno, mofu, dozi
WHERE stud.id = seno.stud_id
  AND seno.mofu_id = mofu.id
  AND mofu.dozi_id = dozi.id;
```

Lösungen “Studi kennt Dozi nicht”

```
SELECT stud.name, stud.vorname,
  dozi.name, dozi.vorname
FROM stud INNER JOIN seno
  ON (stud.id != seno.stud_id)
  INNER JOIN mofu
  ON (seno.mofu_id != mofu.id)
  INNER JOIN dozi
  ON (mofu.dozi_id != dozi.id);
```

```
SELECT stud.name, stud.vorname,
  dozi.name, dozi.vorname
FROM stud, dozi
MINUS
SELECT stud.name, stud.vorname,
  dozi.name, dozi.vorname
FROM stud, seno, mofu, dozi
WHERE stud.id = seno.stud_id
  AND seno.mofu_id = mofu.id
  AND mofu.dozi_id = dozi.id;
```

Aufgabe zum Thema Dimension

BIER_ID	BIER_NAME	SORTE	BRAUEREI	KATEGORIE	ORTSCHAFT	KANTON
Kobra	Glattgold	Pils	Hardwald	Mikro	Wallisellen	Zürich
101	Balthasar	Bock	Hardwald	Mikro	Wallisellen	Zürich
102	Glatthopfen	Ale	Monsterbräu	Nano	Dübendorf	Zürich
103	Kobra	Stout	Monsterbräu	Nano	Wallisellen	Zürich
104	Hopfenperle	Lager	Feldschlösschen	Konzern	Rheinfelden	Aargau
105	Pacific Porter	Porter	Sudwerk	Mikro	Pfäffikon	Zürich
106	Western Rider	Ale	Sudwerk	Mikro	Pfäffikon	Zürich

Beiliegend ein Datenbeispiel, das bezüglich Ortschaft inkonsistent ist, da eine Brauerei nur in einer Ortschaft sein darf. Schreiben Sie ein Select, das solche Inkonsistenzen findet.

Lösungen

```
SELECT
  SET1.Brauerei, SET1.Ortschaft,
  SET2.Brauerei, SET2.Ortschaft
FROM Bier SET1 LEFT OUTER JOIN
     Bier SET2 ON (
  SET1.Brauerei = SET2.Brauerei)
WHERE NOT SET1.Ortschaft =
SET2.Ortschaft;

SELECT Brauerei, count(Ortschaft)
FROM Bier
GROUP BY Brauerei;

SELECT *
FROM Bier a
  JOIN ON Bier b
WHERE a.Brauerei = b.Brauerei
  AND a.Ortschaft NOT b.Ortschaft;
```

```
SELECT ID, Bier_Name,
    count(Ortschaft)
FROM Bier
WHERE count(Ortschaft) = 1
ORDER BY ID, Bier_Name;

SELECT b1.Brauerei, b1.Ortschaft,
   b2.Ortschaft
FROM Bier b1 FULL OUTER JOIN Bier b2
  ON b1.ID = b2.ID
WHERE b1.Ortschaft <> b2.Ortschaft
GROUP BY b1.Ortschaft;

SELECT Brauerei
FROM
(SELECT distinct Brauerei, Ortschaft
FROM Bier)
GROUP BY Brauerei
HAVING count(*) > 1;
```

Lösungen

```
SELECT
  SET1.Brauerei, SET1.Ortschaft,
  SET2.Brauerei, SET2.Ortschaft
FROM Bier SET1 LEFT OUTER JOIN
     Bier SET2 ON (
  SET1.Brauerei = SET2.Brauerei)
WHERE NOT SET1.Ortschaft =
SET2.Ortschaft;

SELECT Brauerei, count(Ortschaft)
FROM Bier
GROUP BY Brauerei;

SELECT *
FROM Bier a
  JOIN ON Bier b
WHERE a.Brauerei = b.Brauerei
  AND a.Ortschaft NOT b.Ortschaft;
```

```
SELECT ID, Bier_Name,
    count(Ortschaft)
FROM Bier
WHERE count(Ortschaft) = 1
ORDER BY ID, Bier_Name;

SELECT b1.Brauerei, b1.Ortschaft,
   b2.Ortschaft
FROM Bier b1 FULL OUTER JOIN Bier b2
  ON b1.ID = b2.ID
WHERE b1.Ortschaft <> b2.Ortschaft
GROUP BY b1.Ortschaft;

SELECT Brauerei
FROM
(SELECT distinct Brauerei, Ortschaft
FROM Bier)
GROUP BY Brauerei
HAVING count(*) > 1;
```

Lösungen

```
SELECT DISTINCT Brauerei, Ortschaft,
  count(Ortschaft)
FROM Bier
GROUP BY Brauerei, Ortschaft
HAVING count(Ortschaft) > 1;

SELECT count(DISTINCT Ortschaft) Anz,
  Brauerei
FROM Bier
GROUP BY Brauerei
HAVING Anz > 1;

SELECT count(DISTINCT Ortschaft) Anz,
  Brauerei
FROM Bier
GROUP BY Brauerei
HAVING count(DISTINCT Ortschaft) > 1;
```

Lösungen

```
SELECT DISTINCT Brauerei, Ortschaft,
  count(Ortschaft)
FROM Bier
GROUP BY Brauerei, Ortschaft
HAVING count(Ortschaft) > 1;

SELECT count(DISTINCT Ortschaft) Anz,
  Brauerei
FROM Bier
GROUP BY Brauerei
HAVING Anz > 1;

SELECT count(DISTINCT Ortschaft) Anz,
  Brauerei
FROM Bier
GROUP BY Brauerei
HAVING count(DISTINCT Ortschaft) > 1;
```

Diskussion

?

!

andrea.kennel@fhnw.ch
andrea@infokennel.ch
www.infokennel.ch