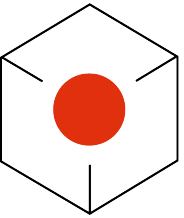
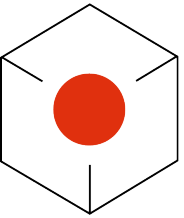


info.kennel

Do you know SQL ?

Dr. Andrea Kennel, Lecturer and Consultant
InfoPunkt Kennel GmbH
Dübendorf-Switzerland
June 2019



Tables

STUD (STUDENTS)

ID	Firstname	Name	Address	PCode	City	E-Mail	Birthdate
500	Anna	Gut	Hofweg 6	3000	Bern		27.09.97
501	Otto	Hug	Dorfstrasse 20	5200	Brugg		15.03.85
502	Kai	Iseli	Lindenhof 5	5200	Brugg		05.12.89
503	Lara	Meier	Markplatz 7	8000	Zürich		05.02.96

SEGR (SEMESTERGRADE)

ID	LESE_ID	Pers_ID	Grade
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

LESE (LECUTREINSEMESTER)

ID	Modul	Semester	Lect_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

LECT (LECTURER)

ID	Firstname	Name	Short	…
10745	Andrea	Kennel	KEA	
15236	Silvia	Ackermann	ACS	
07851	Dierk	König	KOD	

Exercises

- Create a list that shows all cities in ascending order and the number of people per city. (2 points) Note: Use the COUNT(*) function for calculation
- List all students who have not yet completed any module, (no grade). (4 points)
- List all pair of lecturers and students, that do not know each other, that means the student never was in a module of this lecturer. (6 points)

Solutions “Number of people per city”

```
SELECT pcode, city, count(*)
FROM stud
GROUP BY pcode, city
ORDER BY city asc;

SELECT count(id) number_of, city
FROM stud
ORDER BY city;

SELECT lese.*, count(segr.pers_id)
FROM lese, segr
ORDER By lese.loc;
```

Solutions “Students without modul”

```
SELECT stud.id, stud.name
FROM stud
WHERE stud.id NOT IN
 (SELECT pers_id FROM segr);
```

```
SELECT *
FROM stud INNER JOIN segr
  ON (stud.id != segr.pers_id);
```

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

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

```
SELECT *
FROM stud
WHERE (
  count(
    SELECT * FROM segr
    WHERE stud.id = segr.perso_id
  ) = 0
);
```

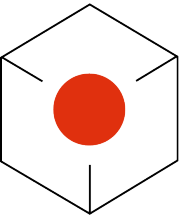
```
SELECT stud.id, stud.name
FROM stud
MiNUS
SELECT segr.pers_id, NULL
FROM segr;
```

Solutions “not know”

```
SELECT stud.name, stud.firstname,
  lect.name, lect.firstname
FROM stud INNER JOIN segr
  ON (stud.id != segr.pers_id)
  INNER JOIN lese
  ON (segr.lese_id != lese.id)
  INNER JOIN lect
  ON (lese.lect_id != lect.id);
```

```
SELECT stud.name, stud.firstname,
  lect.name, lect.firstname
FROM stud, lect
MINUS
SELECT stud.name, stud.firstname,
  lect.name, lect.firstname
FROM stud, segr, lese, lect
WHERE stud.id = segr.pers_id
  AND segr.lese_id = lese.id
  AND lese.lect_id = lect.id;
```

```
SELECT stud.name, stud.firstname,
  lect.name, lect.firstname
FROM stud, lect
WHERE (stud.id, lect.id) NOT IN
(SELECT segr.pers_id, lese.lect_id
 FROM segr, lese
 WHERE segr.lese_id = lese.id);
```



Query on Dimension

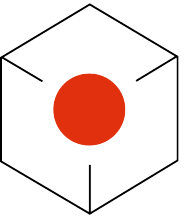
BEER_ID	BEER_NAME	KIND	BREWERY	SIZE	CITY	CANTON
100	Glattgold	Pils	Hardwald	Mikro	Wallisellen	Zürich
101	Balthasar	Bock	Hardwald	Mikro	Wallisellen	Zürich
102	Glatthopfen	Ale	Monsterbräu	Nano	Dübendorf	Zürich
103	Kobra	Stout	Monsterbräu	Nano	Wallisellen	Zürich
104	Hopfenperle	Lager	Feldschlösschen	Large	Rheinfelden	Aargau
105	Pacific Porter	Porter	Sudwerk	Mikro	Pfäffikon	Zürich
106	Western Rider	Ale	Sudwerk	Mikro	Pfäffikon	Zürich

Enclosed is an example of data that is inconsistent with the city, since a brewery can only be in one location (city). Write a select that finds such inconsistencies.

Solution

```
SELECT
  set1.brewery, set1.city,
  set2.brewery, set2.city
FROM beer set1 LEFT OUTER JOIN
     beer set2 ON (
   set1.brewery = set2.brewery)
WHERE NOT set1.city = set2.city;

SELECT brewery, count(city)
FROM beer
GROUP BY brewery;

SELECT *
FROM beer AS a
  JOIN ON beer AS b
WHERE a.brewery = b.brewery
  AND a.city NOT b.city;
```

```
SELECT beer_id, beer_name,
    count(city)
FROM beer
WHERE count(city) = 1
ORDER BY beer_id, beer_name;

SELECT b1.brewery, b1.city,
   b2.city
FROM beer b1 FULL OUTER JOIN beer b2
  ON b1.beer_id = b2.beer_id
WHERE b1.city <> b2.city
GROUP BY b1.city;

SELECT brewery, count(*)
FROM beer
GROUP BY brewery;
```



Solution

```
SELECT count(DISTINCT city) number_of,
  brewery
FROM beer
GROUP BY brewery
HAVING number_of > 1;

SELECT DISTINCT brewery, city,
  count(city)
FROM beer
GROUP BY brewery
HAVING count(city) > 1;

SELECT count(DISTINCT city) number_of,
 brewery
FROM beer
GROUP BY brewery
HAVING count(DISTINCT city) > 1;
```

Discussion

?

!

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