Bases de Données Fall, 2024

TD 6

The cut-and-paste code from this pdf file will work directly on the computer with `postgreSQL`, in case you want to try these queries.

Here are the tables we used in class:

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

(a) course

id	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000.00
12121	Wu	Finance	90000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
32343	El Said	History	60000.00
33456	Gold	Physics	87000.00
45565	Katz	Comp. Sci.	75000.00
58583	Califieri	History	62000.00
76543	Singh	Finance	80000.00
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
98345	Kim	Elec. Eng.	80000.00

(b) teacher

id	*name*	*dept_name*	*tot_cred*
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120

(c) student

course_id	*sec_id*	*semester*	*year*	*building*	*rn*	*time_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

(d) section

id	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

(e) teaches

id	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	CS-101	1	Fall	2009	A
00128	CS-347	1	Fall	2009	A-
12345	CS-101	1	Fall	2009	C
12345	CS-190	2	Spring	2009	A
12345	CS-315	1	Spring	2010	A
12345	CS-347	1	Fall	2009	A
19991	HIS-351	1	Spring	2010	B
23121	FIN-201	1	Spring	2010	C+
44553	PHY-101	1	Fall	2009	B-
45678	CS-101	1	Fall	2009	F
45678	CS-101	1	Spring	2010	B+
45678	CS-319	1	Spring	2010	B
54321	CS-101	1	Fall	2009	A-
54321	CS-190	2	Spring	2009	B+
55739	MU-199	1	Spring	2010	A-
76543	CS-101	1	Fall	2009	A
76543	CS-319	2	Spring	2010	A
76653	EE-181	1	Spring	2009	C
98765	CS-101	1	Fall	2009	C-
98765	CS-315	1	Spring	2010	B
98988	BIO-101	1	Summer	2009	A
98988	BIO-301	1	Summer	2010	

(f) takes

dept_name	*building*	*budget*
Biology	Watson	90000.00
Comp. Sci.	Taylor	100000.00
Elec. Eng.	Taylor	85000.00
Finance	Painter	120000.00
History	Painter	50000.00
Music	Packard	80000.00
Physics	Watson	70000.00

(g) department

1. Find the names of all students who have taken a class in Fall 2009, using **NATURAL INNER JOIN** statement.

   Recherchez les noms de tous les étudiants qui ont suivi un cours à l'automne 2009, en utilisant l'instruction **NATURAL INNER JOIN**.

```
SELECT DISTINCT student.name
FROM takes NATURAL INNER JOIN student
WHERE takes.semester = 'Fall' AND takes.year = 2009;
```

name
Bourikas
Brown
Levy
Peltier
Shankar
Williams
Zhang

2. List all pairs of students who have taken at least two classes together. No products, only use **JOINS**.

   Listez toutes les tuples d'étudiants qui ont suivi au moins deux cours ensemble. N'utilisez pas les produits, utilisez uniquement **JOINS**.

```
SELECT A.name, B.name
FROM (student NATURAL INNER JOIN takes) as A
       CROSS JOIN
     (student NATURAL INNER JOIN takes) as B
WHERE A.course_id = B.course_id AND A.sec_id = B.sec_id
      AND A.semester = B.semester AND A.year = B.year
      AND A.name < B.name
GROUP BY A.id, B.id
HAVING COUNT(*) >= 2;
```

name	*name*
Shankar	Zhang
Shankar	Williams
Bourikas	Shankar

3. List all pairs of students who have taken most classes together. Do not use products, use only **JOINS**.

   Listez tous les tuples d'étudiants qui ont suivi la plupart des cours ensemble. N'utilisez pas de produits, utilisez uniquement **JOINS**.

```
SELECT A.name, B.name
FROM (student NATURAL INNER JOIN takes) as A
       CROSS JOIN
     (student NATURAL INNER JOIN takes) as B
WHERE A.course_id = B.course_id AND A.sec_id = B.sec_id
      AND A.semester = B.semester AND A.year = B.year
      AND A.id <> B.id AND A.name < B.name
GROUP BY A.name, A.id, B.name, B.id
HAVING count(*) >= ALL(
   SELECT count(*)
   FROM (student NATURAL INNER JOIN takes) as A
          CROSS JOIN
        (student NATURAL INNER JOIN takes) as B
   WHERE A.course_id = B.course_id AND A.sec_id = B.sec_id
        AND A.semester = B.semester AND A.year = B.year
        AND A.id <> B.id AND A.name < B.name
  GROUP BY A.id, B.id);
```

name	*name*
Shankar	Zhang
Shankar	Williams
Bourikas	Shankar

4. Give the total number of students taught by each teacher (the same student in two classes is counted twice), including teachers who have not taught any students. Sort by decreasing number of students. Do not use products, use only **JOINS**.

   Donnez le nombre total d'élèves enseignés par chaque enseignant (le même élève dans deux classes est compté deux fois), y compris les enseignants qui n'ont enseigné à aucun élève. Triez par nombre décroissant d'élèves. N'utilisez pas de produits, utilisez uniquement **JOINS**.

```
SELECT teacher.name, count(course_id)
FROM (takes INNER JOIN teaches USING (course_id, sec_id,
     semester, year))
       RIGHT OUTER JOIN teacher ON teaches.id = teacher.id
GROUP BY teacher.name, teacher.id
ORDER BY count(course_id) DESC;
```

name	*count*
Srinivasan	10
Brandt	3
Katz	2
Crick	2
Einstein	1
Kim	1
Mozart	1
El Said	1
Wu	1
Singh	0
Gold	0
Califieri	0

5. Give the number of 'A' grades given by **each** teacher. Do not use products, use only **JOINS**.

   Donnez le nombre de notes 'A' données par **chaque** enseignant. N'utilisez pas de produits, utilisez uniquement **JOINS**.

```
WITH mytakes (id, course_id, sec_id, semester, year, grade) AS
   (SELECT id, course_id, sec_id, semester, year, grade
    FROM takes
    WHERE grade = 'A')

SELECT teacher.name, count(course_id)
FROM (mytakes INNER JOIN teaches USING (course_id, sec_id,
     semester, year))
       RIGHT OUTER JOIN teacher ON teaches.id = teacher.id
GROUP BY teacher.name, teacher.id
ORDER BY count(course_id) DESC;
```

name	*count*
Srinivasan	4
Brandt	2
Crick	1
Califieri	0
Einstein	0
Wu	0
Kim	0
Mozart	0
El Said	0
Katz	0
Singh	0
Gold	0

6. Give all pairs of teachers and students where the student has taken the course of the teacher, together with how many times that student has been in a course of that teacher. Use only **JOINS**.

   Indiquez toutes les tuples d'enseignants et d'élèves pour lesquelles l'élève a suivi le cours de l'enseignant, ainsi que le nombre de fois où cet élève a suivi un cours de cet enseignant. Utilisez uniquement **JOINS**.

```
SELECT teacher.name, student.name, count(*)
FROM (teacher NATURAL JOIN teaches) INNER JOIN
      (takes NATURAL JOIN student) USING
      (course_id, sec_id, semester, year)
GROUP BY teacher.name, student.name;
```

name	*name*	*count*
Brandt	Brown	1
Brandt	Shankar	1
Brandt	Williams	1
Crick	Tanaka	2
Einstein	Peltier	1
El Said	Brandt	1
Katz	Levy	2
Kim	Aoi	1
Mozart	Sanchez	1
Srinivasan	Bourikas	2
Srinivasan	Brown	1
Srinivasan	Levy	1
Srinivasan	Shankar	3
Srinivasan	Williams	1
Srinivasan	Zhang	2
Wu	Chavez	1

7. Give the pair of teachers and students where the student has taken at least two classes from that teacher. Use only **JOINS**.

   Donnez toutes les tuples de professeurs et d'élèves où l'élève a suivi au moins deux cours de ce professeur. Utilisez uniquement **JOINS**.

```
SELECT mytable.tn, mytable.sn
FROM (SELECT teacher.name as tn, student.name as sn, count(*) as tc
        FROM (teacher NATURAL JOIN teaches) INNER JOIN (takes NATURAL JOIN
    student) USING (course_id, sec_id,       semester, year)
        GROUP BY teacher.name, student.name) AS mytable
WHERE tc >= 2;
```

tn	*sn*
Crick	Tanaka
Katz	Levy
Srinivasan	Bourikas
Srinivasan	Shankar
Srinivasan	Zhang