

## TP n° 1 : Quelques fonctions SQL et requêtes

### Exercice 1 :

Requêtes de Création de tables de la base de données de gestion des cours :

```
Language SQL ? Rows 10 ? Clear Command Find Tables

1 create table classroom
2     (building      varchar(15),
3      room_number   varchar(7),
4      capacity      numeric(4,0),
5      primary key (building, room_number)
6  );
7
8 create table department
9     (dept_name     varchar(20),
10    building        varchar(15),
11    budget          numeric(12,2),
12    primary key (dept_name)
13 );
```

```
1 create table course
2     (course_id     varchar(8),
3      title          varchar(50),
4      dept_name      varchar(20),
5      credits        numeric(2,0),
6      primary key (course_id),
7      foreign key (dept_name) references department
8  );
9 create table teacher
10    (ID             varchar(5),
11     name            varchar(20),
12     dept_name       varchar(20),
13     salary          numeric(8,2),
14     primary key (ID),
15     foreign key (dept_name) references department
16 );
```

```

1  create table section
2      (course_id      varchar(8),
3       sec_id         varchar(8),
4       semester       varchar(6)
5         check (semester in ('Fall', 'Winter', 'Spring', 'Summer')),
6       year           numeric(4,0),
7       building       varchar(15),
8       room_number    varchar(7),
9       time_slot_id   varchar(4),
10      primary key (course_id, sec_id, semester, year),
11      foreign key (course_id) references course,
12      foreign key (building, room_number) references classroom
13  );
14

```

```

1  create table teaches
2      (ID              varchar(5),
3       course_id       varchar(8),
4       sec_id          varchar(8),
5       semester        varchar(6),
6       year            numeric(4,0),
7       primary key (ID, course_id, sec_id, semester, year),
8       foreign key (course_id, sec_id, semester, year) references section,
9       foreign key (ID) references teacher
10  );
11

```

```

1  create table student
2      (ID              varchar(5),
3       name            varchar(20),
4       dept_name       varchar(20),
5       tot_cred        numeric(3,0),
6       primary key (ID),
7       foreign key (dept_name) references department
8  );
9

```

```

10  create table takes
11  (ID          varchar(5),
12   course_id   varchar(8),
13   sec_id      varchar(8),
14   semester    varchar(6),
15   year        numeric(4,0),
16   grade       varchar(2),
17   primary key (ID, course_id, sec_id, semester, year),
18   foreign key (course_id, sec_id, semester, year) references section,
19   foreign key (ID) references student
20  );
21

```

```

22  create table advisor
23  (s_ID        varchar(5),
24   i_ID        varchar(5),
25   primary key (s_ID),
26   foreign key (i_ID) references teacher (ID),
27   foreign key (s_ID) references student (ID)
28  );
29

```

```

21  create table time_slot
22  (time_slot_id varchar(4),
23   day          varchar(1),
24   start_hr     numeric(2),
25   start_min    numeric(2),
26   end_hr       numeric(2),
27   end_min      numeric(2),
28   primary key (time_slot_id, day, start_hr, start_min)
29  );
30

```

```

31  create table prereq
32  (course_id   varchar(8),
33   prereq_id   varchar(8),
34   primary key (course_id, prereq_id),
35   foreign key (course_id) references course,
36   foreign key (prereq_id) references course
37  );

```

**Résultat :**

| Recently Created Tables |                |
|-------------------------|----------------|
| ADVISOR                 | 6 minutes ago  |
| TIME_SLOT               | 6 minutes ago  |
| TEACHES                 | 6 minutes ago  |
| TEACHER                 | 6 minutes ago  |
| TAKES                   | 6 minutes ago  |
| STUDENT                 | 6 minutes ago  |
| SECTION                 | 6 minutes ago  |
| PREREQ                  | 6 minutes ago  |
| CLASSROOM               | 6 minutes ago  |
| COURSE                  | 6 minutes ago  |
| DEPARTMENT              | 6 minutes ago  |
| HTMLDB_PLAN_TABLE       | 17 minutes ago |

2) Insérer un nouveau cours dont l'identifiant est BIO-101, intitulé Intro. to Biology, assurée par le département Biology et son crédit est de 4

`INSERT INTO course VALUES ('BIO-101', 'Intro. to Biology', 'Biology', 4);`

Vérification :

| <div> <div>↶ ↷ 🔍 🛠️ A-Z ⚙️</div> <div> 1 <code>SELECT * FROM course WHERE course_id = 'BIO-101';</code><br/> 2 <input type="text"/> </div> </div> |                   |           |                   |
|---------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|-----------|-------------------|
| Results                                                                                                                                           | Explain           | Describe  | Saved SQL History |
| COURSE_ID                                                                                                                                         | TITLE             | DEPT_NAME | CREDITS           |
| BIO-101                                                                                                                                           | Intro. to Biology | Biology   | 4                 |

**Exercice 2 :**

1. Afficher la structure de la relation section et son contenu (cours proposés).

Pour voir les colonnes et leurs types dans la table **Section** :

`DESCRIBE section;`

```
1 DESCRIBE section;
2
3
```

| Results | Explain      | Describe | Saved SQL | History |   |   |   |   |   |
|---------|--------------|----------|-----------|---------|---|---|---|---|---|
| SECTION | COURSE_ID    | VARCHAR2 | 8         | -       | - | 1 | - | - | - |
|         | SEC_ID       | VARCHAR2 | 8         | -       | - | 2 | - | - | - |
|         | SEMESTER     | VARCHAR2 | 6         | -       | - | 3 | - | - | - |
|         | YEAR         | NUMBER   | -         | 4       | 0 | 4 | - | - | - |
|         | BUILDING     | VARCHAR2 | 15        | -       | - | - | ✓ | - | - |
|         | ROOM_NUMBER  | VARCHAR2 | 7         | -       | - | - | ✓ | - | - |
|         | TIME_SLOT_ID | VARCHAR2 | 4         | -       | - | - | ✓ | - | - |

Afficher le contenu de la table **Section** :

`SELECT * FROM section;`

<

2) Afficher tous les renseignements sur les cours que l'on peut programmer (relation course)

`SELECT * FROM course;`

```
1 SELECT * FROM course;
```

| 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       |

3) Afficher les titres des cours et les départements qui les proposent.

```
SELECT title , dept_name
FROM course ;
```

```
1 SELECT title , dept_name
2 FROM course ;
```

| TITLE                      | DEPT_NAME  |
|----------------------------|------------|
| Intro. to Biology          | Biology    |
| Genetics                   | Biology    |
| Computational Biology      | Biology    |
| Intro. to Computer Science | Comp. Sci. |

4) Afficher les noms des départements ainsi que leur budget

```
SELECT dept_name , budget
FROM department ;
```

|   |                           |
|---|---------------------------|
| 1 | SELECT dept_name , budget |
| 2 | FROM department ;         |

| Results    | Explain | Describe | Saved SQL | History |
|------------|---------|----------|-----------|---------|
| DEPT_NAME  |         | BUDGET   |           |         |
| Biology    |         | 90000    |           |         |
| Comp. Sci. |         | 100000   |           |         |
| Elec. Eng. |         | 85000    |           |         |
| Finance    |         | 120000   |           |         |

5) Afficher tous les noms des enseignants et leur département

```
SELECT name , dept_name
FROM teacher;
```

|   |                         |
|---|-------------------------|
| 1 | SELECT name , dept_name |
| 2 | FROM teacher;           |
| 3 |                         |

| Results    | Explain | Describe   | Saved SQL | History |
|------------|---------|------------|-----------|---------|
| NAME       |         | DEPT_NAME  |           |         |
| Srinivasan |         | Comp. Sci. |           |         |
| Wu         |         | Finance    |           |         |
| Mozart     |         | Music      |           |         |
| Einstein   |         | Physics    |           |         |

6) Afficher tous les noms des enseignants ayant un salaire supérieur strictement à 65.000 \$.

```
SELECT name
FROM teacher
WHERE salary > 65000;
```

|   |                       |
|---|-----------------------|
| 1 | SELECT name           |
| 2 | FROM teacher          |
| 3 | WHERE salary > 65000; |
| 4 |                       |

| Results  | Explain | Describe | Saved SQL | History |
|----------|---------|----------|-----------|---------|
| NAME     |         |          |           |         |
| Wu       |         |          |           |         |
| Einstein |         |          |           |         |
| Gold     |         |          |           |         |
| Katz     |         |          |           |         |

7) Afficher les noms des enseignants ayant un salaire compris entre 55.000 \$ et 85.000 \$.

```
SELECT name
FROM teacher
WHERE salary between 55000 and 85000;
```

|               |                                       |   |      |    |
|---------------|---------------------------------------|---|------|----|
| Language      | SQL                                   | ? | Rows | 10 |
| Clear Command | Find Tables                           |   |      |    |
| 1             | SELECT name                           |   |      |    |
| 2             | FROM teacher                          |   |      |    |
| 3             | WHERE salary between 55000 and 85000; |   |      |    |
| 4             |                                       |   |      |    |

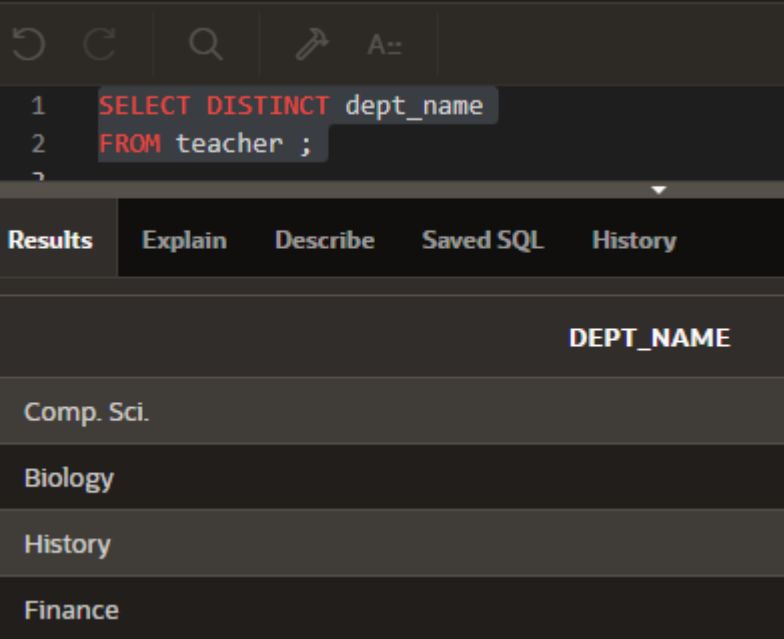
  

| Results    | Explain | Describe | Saved SQL | History |
|------------|---------|----------|-----------|---------|
| NAME       |         |          |           |         |
| Srinivasan |         |          |           |         |
| El Said    |         |          |           |         |
| Katz       |         |          |           |         |

8) Afficher les noms des départements, en utilisant la relation teacher et éliminer les doublons.



```
SELECT DISTINCT dept_name  
FROM teacher ;
```

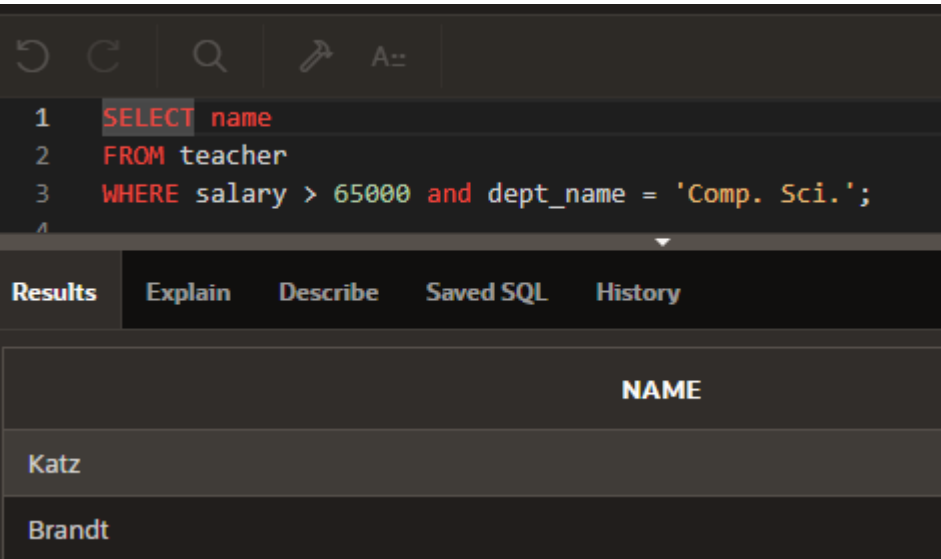


The screenshot shows a SQL IDE interface. The top toolbar contains icons for undo, redo, search, and a keyboard shortcut 'A++'. The SQL editor displays the query: `1 SELECT DISTINCT dept_name` and `2 FROM teacher ;`. Below the editor, a tabbed interface shows 'Results' selected. The results table has a single column header 'DEPT\_NAME' and four rows of data: 'Comp. Sci.', 'Biology', 'History', and 'Finance'.

| DEPT_NAME  |
|------------|
| Comp. Sci. |
| Biology    |
| History    |
| Finance    |

9) Afficher tous les noms des enseignants du département informatique ayant un salaire supérieur strictement à 65.000 \$.

```
SELECT name  
FROM teacher  
WHERE salary > 65000 and dept_name = 'Comp. Sci.';
```

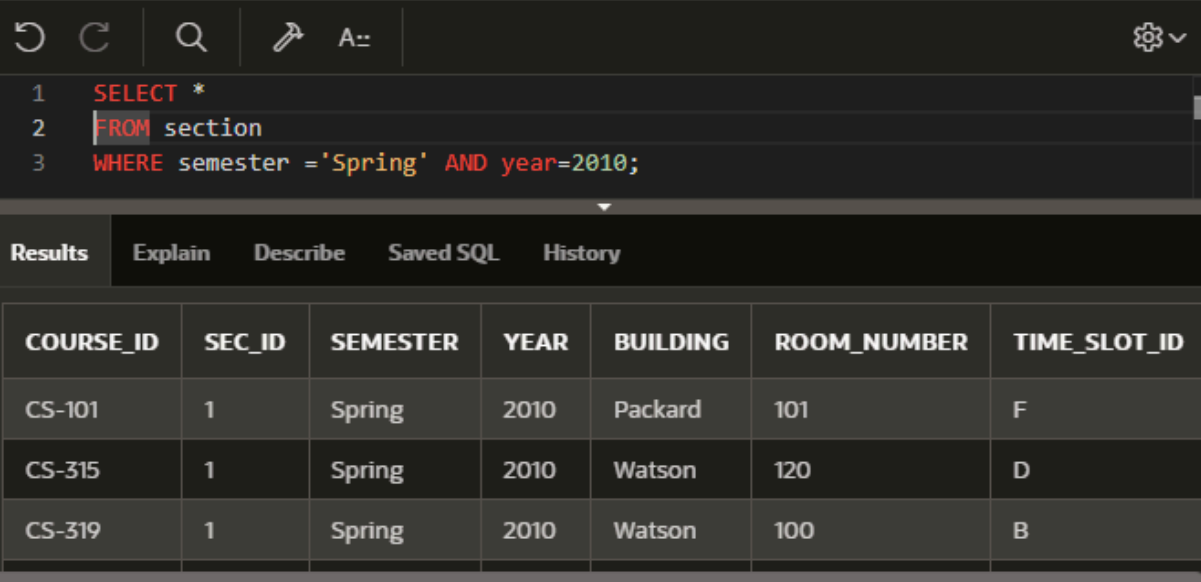


The screenshot shows a SQL IDE interface. The top toolbar contains icons for undo, redo, search, and a keyboard shortcut 'A++'. The SQL editor displays the query: `1 SELECT name`, `2 FROM teacher`, and `3 WHERE salary > 65000 and dept_name = 'Comp. Sci.';`. Below the editor, a tabbed interface shows 'Results' selected. The results table has a single column header 'NAME' and two rows of data: 'Katz' and 'Brandt'.

| NAME   |
|--------|
| Katz   |
| Brandt |

10) Afficher tous les renseignements sur les cours proposés au printemps 2010 (relation section).

```
SELECT *  
FROM section  
WHERE semester ='Spring' AND year=2010;
```



The screenshot shows a SQL query editor with a dark theme. The query is entered in the editor and is highlighted. Below the editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, and it displays a table with the following data:

| COURSE_ID | SEC_ID | SEMESTER | YEAR | BUILDING | ROOM_NUMBER | TIME_SLOT_ID |
|-----------|--------|----------|------|----------|-------------|--------------|
| CS-101    | 1      | Spring   | 2010 | Packard  | 101         | F            |
| CS-315    | 1      | Spring   | 2010 | Watson   | 120         | D            |
| CS-319    | 1      | Spring   | 2010 | Watson   | 100         | B            |

11) Afficher tous les titres des cours dispensés par le département informatique qui ont plus de trois crédits.

```
SELECT title  
FROM course  
WHERE dept_name ='Comp. Sci.'  
AND credits >3;
```

|   |                                |
|---|--------------------------------|
| 1 | SELECT title                   |
| 2 | FROM course                    |
| 3 | WHERE dept_name = 'Comp. Sci.' |
| 4 | AND credits >3;                |

|         |         |          |           |         |
|---------|---------|----------|-----------|---------|
| Results | Explain | Describe | Saved SQL | History |
|---------|---------|----------|-----------|---------|

| TITLE                      |
|----------------------------|
| Intro. to Computer Science |
| Game Design                |

12) Afficher tous les noms des enseignants ainsi que le nom de leur département et les noms des bâtiments qui les hébergent.

```
SELECT teacher.name , teacher.dept_name , department.building
FROM teacher , department
WHERE teacher.dept_name = department.dept_name ;
```

|   |                                                               |
|---|---------------------------------------------------------------|
| 1 | SELECT teacher.name , teacher.dept_name , department.building |
| 2 | FROM teacher , department                                     |
| 3 | WHERE teacher.dept_name = department.dept_name ;              |
| 4 |                                                               |

|         |         |          |           |         |
|---------|---------|----------|-----------|---------|
| Results | Explain | Describe | Saved SQL | History |
|---------|---------|----------|-----------|---------|

| NAME       | DEPT_NAME  | BUILDING |
|------------|------------|----------|
| Srinivasan | Comp. Sci. | Taylor   |
| Wu         | Finance    | Painter  |
| Mozart     | Music      | Packard  |

13) Afficher tous les étudiants ayant suivi au moins un cours en informatique

```
SELECT distinct student.name
FROM student , takes , course
WHERE student.id = takes.id and takes.course_id = course .
course_id and course.dept_name = 'Comp. Sci.';
```

The screenshot shows a SQL query editor with the following code:

```
1 SELECT distinct student.name
2 FROM student , takes , course
3 WHERE student.id = takes.id and takes.course_id = course .
4 course_id and course.dept_name = 'Comp. Sci.';
```

Below the editor is a results window with tabs: Results, Explain, Describe, Saved SQL, and History. The Results tab is active, showing a table with the following data:

| NAME     |
|----------|
| DOUBIKAS |
| Shankar  |
| Williams |

At the bottom of the results window, it says "6 rows returned in 0.04 seconds" and there is a "Download" button.

14) Afficher les noms des étudiants ayant suivi un cours dispensé par un enseignant nommé Einstein (éliminer les doublons).

```
SELECT DISTINCT student.name
FROM student, teacher, takes, teaches
WHERE student.id = takes.id
  AND takes.course_id = teaches.course_id
  AND takes.sec_id = teaches.sec_id
  AND takes.semester = teaches.semester
  AND takes.year = teaches.year
  AND teaches.id = teacher.id
  AND teacher.name = 'Einstein';
```

|   |                                         |
|---|-----------------------------------------|
| 1 | SELECT DISTINCT student.name            |
| 2 | FROM student, teacher, takes, teaches   |
| 3 | WHERE student.id = takes.id             |
| 4 | AND takes.course_id = teaches.course_id |
| 5 | AND takes.sec_id = teaches.sec_id       |
| 6 | AND takes.semester = teaches.semester   |
| 7 | AND takes.year = teaches.year           |
| 8 | AND teaches.id = teacher.id             |

| Results | Explain | Describe | Saved SQL | History |
|---------|---------|----------|-----------|---------|
| NAME    |         |          |           |         |
| Peltier |         |          |           |         |

15) Afficher tous les identifiants des cours et les enseignants qui les ont assurés.

```
SELECT teacher.name, teaches.course_id
FROM teacher, teaches
WHERE teacher.id = teaches.id;
```

|   |                                        |
|---|----------------------------------------|
| 1 | SELECT teacher.name, teaches.course_id |
| 2 | FROM teacher, teaches                  |
| 3 | WHERE teacher.id = teaches.id;         |
| 4 |                                        |

| Results    | Explain | Describe  | Saved SQL | History |
|------------|---------|-----------|-----------|---------|
| NAME       |         | COURSE_ID |           |         |
| Srinivasan |         | CS-101    |           |         |
| Srinivasan |         | CS-315    |           |         |
| Srinivasan |         | CS-347    |           |         |

16) Afficher le nombre d'inscrits pour chaque enseignement proposé au printemps 2010

```
SELECT takes.course_id, takes.sec_id, takes.semester, takes.year, COUNT(*)
FROM takes
WHERE takes.semester = 'Spring' AND takes.year = 2010
GROUP BY takes.course_id, takes.sec_id, takes.semester, takes.year;
```

|   |          |                           |               |                   |             |          |
|---|----------|---------------------------|---------------|-------------------|-------------|----------|
| 1 | SELECT   | takes.course_id,          | takes.sec_id, | takes.semester,   | takes.year, | COUNT(*) |
| 2 | FROM     | takes                     |               |                   |             |          |
| 3 | WHERE    | takes.semester = 'Spring' | AND           | takes.year = 2010 |             |          |
| 4 | GROUP BY | takes.course_id,          | takes.sec_id, | takes.semester,   | takes.year; |          |
| 5 |          |                           |               |                   |             |          |

| Results   | Explain | Describe | Saved SQL | History  |
|-----------|---------|----------|-----------|----------|
| COURSE_ID | SEC_ID  | SEMESTER | YEAR      | COUNT(*) |
| CS-315    | 1       | Spring   | 2010      | 2        |
| CS-101    | 1       | Spring   | 2010      | 1        |
| CS-319    | 2       | Spring   | 2010      | 1        |

17) Afficher les noms des départements et les salaires maximum de leurs enseignants.

SELECT dept\_name , max ( salary ) FROM teacher 2 GROUP BY dept\_name ;

18) Afficher le nombre d'inscrits pour chaque enseignement proposés

SELECT takes.course\_id, takes.sec\_id, takes.semester, takes.year, COUNT(\*)  
FROM takes  
GROUP BY takes.course\_id, takes.sec\_id, takes.semester, takes.year;

SQL Commands Schema WKSP\_DBATPS

Language SQL Rows 10 Save Run

Clear Command Find Tables

```

1 SELECT takes.course_id, takes.sec_id, takes.semester, takes.year, COUNT(*)
2 FROM takes
3 GROUP BY takes.course_id, takes.sec_id, takes.semester, takes.year;
4

```

Results Explain Describe Saved SQL History

| COURSE_ID | SEC_ID | SEMESTER | YEAR | COUNT(*) |
|-----------|--------|----------|------|----------|
| CS-190    | 2      | Spring   | 2009 | 2        |
| CS-319    | 1      | Spring   | 2010 | 1        |
| CS-347    | 1      | Fall     | 2009 | 2        |
| MU-199    | 1      | Spring   | 2010 | 1        |
| CS-101    | 1      | Fall     | 2009 | 6        |

19) . Afficher le nombre total de cours qui ont eu lieu dans chaque bâtiment, pendant l'automne 2009 et le printemps 2010.

```

SELECT building, COUNT(*)
FROM section
WHERE (semester, year) IN (('Fall', 2009), ('Spring', 2010))
GROUP BY building;

```

</

20) Afficher le nombre total de cours dispensées par chaque département et qui ont eu dans le même bâtiment qui l'abrite.

```
SELECT department.dept_name, COUNT(*)
FROM section, department, teacher, teaches
WHERE section.course_id = teaches.course_id
  AND section.sec_id = teaches.sec_id
  AND section.semester = teaches.semester
  AND section.year = teaches.year
  AND teaches.id = teacher.id
  AND teacher.dept_name = department.dept_name
  AND department.building = section.building
GROUP BY department.dept_name;
```



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with icons for undo, redo, search, and a command prompt. Below the toolbar, a SQL query is entered in a text area. The query is as follows:

```

1  SELECT department.dept_name, COUNT(*)
2  FROM section, department, teacher, teaches
3  WHERE section.course_id = teaches.course_id
4        AND section.sec_id = teaches.sec_id
5        AND section.semester = teaches.semester
6        AND section.year = teaches.year
7        AND teaches.id = teacher.id
8        AND teacher.dept_name = department.dept_name
9        AND department.building = section.building
10 GROUP BY department.dept_name;

```

Below the query editor, there is a tabbed interface with the following tabs: Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is currently selected, displaying the following table:

| DEPT_NAME | COUNT(*) |
|-----------|----------|
| Physics   | 1        |

At the bottom of the Results tab, it states: "5 rows returned in 0.14 seconds" and provides a "Download" link.

21) Afficher les titres des cours proposés et qui ont eu lieu et les enseignants qui les ont assurés.

```

SELECT course.title, teacher.name
FROM section, teacher, teaches, course
WHERE section.course_id = teaches.course_id
      AND section.sec_id = teaches.sec_id
      AND section.semester = teaches.semester
      AND section.year = teaches.year
      AND teaches.id = teacher.id
      AND section.course_id = course.course_id
ORDER BY course.title;

```

The screenshot shows a SQL IDE interface. The top toolbar includes icons for undo, redo, search, and a settings gear. The SQL editor contains the following query:

```

1  SELECT course.title, teacher.name
2  FROM section, teacher, teaches, course
3  WHERE section.course_id = teaches.course_id
4        AND section.sec_id = teaches.sec_id
5        AND section.semester = teaches.semester
6        AND section.year = teaches.year
7        AND teaches.id = teacher.id
8        AND section.course_id = course.course_id
9  ORDER BY course.title;

```

Below the editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, displaying a table with two columns: 'TITLE' and 'NAME'.

| TITLE                    | NAME       |
|--------------------------|------------|
| Database System Concepts | Srinivasan |
| Game Design              | Brandt     |

22) Afficher le nombre total de cours qui ont eu lieu pour chacune des périodes Summer, Fall et Spring.

`SELECT section . semester , count (*) FROM section 2 GROUP BY section . semester`

The screenshot shows a SQL IDE interface. The top toolbar includes a 'Language' dropdown set to 'SQL', a 'Rows' dropdown set to '10', and buttons for 'Save' and 'Run'. Below the toolbar are 'Clear Command' and 'Find Tables' buttons. The SQL editor contains the following query:

```

1  SELECT section . semester , count (*) FROM section
2  GROUP BY section . semester
3

```

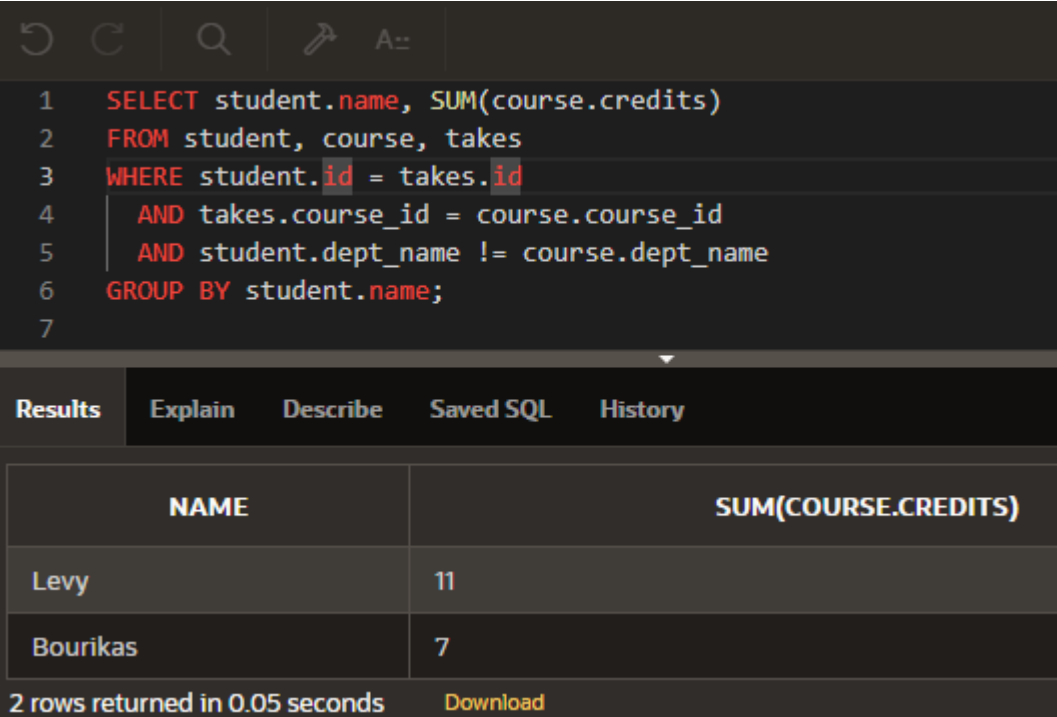
Below the editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, displaying a table with two columns: 'SEMESTER' and 'COUNT(\*)'.

| SEMESTER | COUNT(*) |
|----------|----------|
| Fall     | 3        |
| Summer   | 2        |
| Spring   | 10       |

At the bottom of the results pane, it says '3 rows returned in 0.01 seconds' with a 'Download' link.

23) Affiché pour chaque étudiant le nombre total de crédits qu'il a obtenu, en suivant des cours qui n'ont pas été proposés par son département.

```
SELECT student.name, SUM(course.credits)
FROM student, course, takes
WHERE student.id = takes.id
      AND takes.course_id = course.course_id
      AND student.dept_name != course.dept_name
GROUP BY student.name;
```



The screenshot shows a SQL query editor with a dark theme. The query is as follows:

```
1 SELECT student.name, SUM(course.credits)
2 FROM student, course, takes
3 WHERE student.id = takes.id
4       AND takes.course_id = course.course_id
5       AND student.dept_name != course.dept_name
6 GROUP BY student.name;
7
```

Below the query editor, there is a tabbed interface with 'Results' selected. The results are displayed in a table with two columns: 'NAME' and 'SUM(COURSE.CREDITS)'. The table contains two rows of data.

| NAME     | SUM(COURSE.CREDITS) |
|----------|---------------------|
| Levy     | 11                  |
| Bourikas | 7                   |

At the bottom of the results section, it says '2 rows returned in 0.05 seconds' and there is a 'Download' link.

24) Pour chaque département, afficher le nombre total de crédits des cours qui ont eu lieu dans ce département.

```
SELECT section . building , sum ( course . credits ) FROM section , course 2 WHERE
section . course_id = course . course_id GROUP BY section . building ;
```

LanguageSQLRows10

Clear CommandFind TablesSaveRun

↶↷🔍🔗A-Z⚙️

1SELECT section . building , sum ( course . credits ) FROM section , course

2WHERE section . course\_id = course . course\_id GROUP BY section . building ;

3

ResultsExplainDescribeSaved SQLHistory

| BUILDING | SUM(COURSE.CREDITS) |
|----------|---------------------|
| Watson   | 10                  |
| Packard  | 14                  |
| Taylor   | 17                  |
| Painter  | 11                  |

4 rows returned in 0.02 secondsDownload