Exercice 1 : Création des tables

1. Création de la table classroom

```
create table classroom
        (building           varchar(15),  -- Nom du bâtiment
        room_number             varchar(7),  -- Numéro de la salle
        capacity            numeric(4,0),  -- Capacité de la salle
        primary key (building, room_number)  -- Clé primaire composée du bâtiment et du numéro de la salle
        );
```

2. Création de la table department

```
create table department
        (dept_name      varchar(20),  -- Nom du département
        building            varchar(15),  -- Bâtiment où se trouve le département
        budget              numeric(12,2),  -- Budget du département
        primary key (dept_name)  -- Clé primaire sur le nom du département
        );
```

3. Création de la table course

```
create table course
        (course_id          varchar(8),  -- Identifiant du cours
        title               varchar(50),  -- Titre du cours
        dept_name           varchar(20),  -- Nom du département offrant le cours
        credits             numeric(2,0),  -- Nombre de crédits associés au cours
        primary key (course_id),  -- Clé primaire sur l'identifiant du cours
        foreign key (dept_name) references department  -- Clé étrangère vers le département
        );
```

4. Création de la table teacher

```
create table teacher
        (ID         varchar(5),  -- Identifiant de l'enseignant
        name                varchar(20),  -- Nom de l'enseignant
        dept_name           varchar(20),  -- Département auquel l'enseignant appartient
        salary              numeric(8,2),  -- Salaire de l'enseignant
        primary key (ID),  -- Clé primaire sur l'identifiant de l'enseignant
        foreign key (dept_name) references department  -- Clé étrangère vers le département
        );
```

5. Création de la table section

```
create table section
        (course_id          varchar(8),  -- Identifiant du cours
        sec_id              varchar(8),  -- Identifiant de la section
        semester            varchar(6)  -- Semestre (Fall, Winter, Spring, Summer)
        check (semester in ('Fall', 'Winter', 'Spring', 'Summer')),  -- Vérification des valeurs du semestre
        year                numeric(4,0),  -- Année académique
        building            varchar(15),  -- Bâtiment où la section est donnée
        room_number             varchar(7),  -- Numéro de la salle pour la section
        time_slot_id        varchar(4),  -- Identifiant du créneau horaire de la section
        primary key (course_id, sec_id, semester, year),  -- Clé primaire composée de plusieurs colonnes
```

```
        foreign key (course_id) references course,  -- Clé étrangère vers le cours
        foreign key (building, room_number) references classroom  -- Clé étrangère vers la salle
        );
```

6. Création de la table teaches

```
create table teaches
        (ID         varchar(5),  -- Identifiant de l'enseignant
        course_id       varchar(8),  -- Identifiant du cours
        sec_id          varchar(8),  -- Identifiant de la section
        semester        varchar(6),  -- Semestre dans lequel l'enseignement a lieu
        year            numeric(4,0),  -- Année académique
        primary key (ID, course_id, sec_id, semester, year),  -- Clé primaire composée de plusieurs colonnes
        foreign key (course_id, sec_id, semester, year) references section,  -- Clé étrangère vers la section
        foreign key (ID) references teacher  -- Clé étrangère vers l'enseignant
        );
```

7. Création de la table student

```
create table student
        (ID         varchar(5),  -- Identifiant de l'étudiant
        name            varchar(20),  -- Nom de l'étudiant
        dept_name       varchar(20),  -- Département auquel appartient l'étudiant
        tot_cred        numeric(3,0),  -- Total des crédits accumulés par l'étudiant
        primary key (ID),  -- Clé primaire sur l'identifiant de l'étudiant
        foreign key (dept_name) references department  -- Clé étrangère vers le département
        );
```

8. Création de la table takes

```
create table takes
        (ID         varchar(5),  -- Identifiant de l'étudiant
        course_id       varchar(8),  -- Identifiant du cours
        sec_id          varchar(8),  -- Identifiant de la section
        semester        varchar(6),  -- Semestre dans lequel le cours est pris
        year            numeric(4,0),  -- Année académique
        grade           varchar(2),  -- Note de l'étudiant pour ce cours
        primary key (ID, course_id, sec_id, semester, year),  -- Clé primaire composée de plusieurs colonnes
        foreign key (course_id, sec_id, semester, year) references section,  -- Clé étrangère vers la section
        foreign key (ID) references student  -- Clé étrangère vers l'étudiant
        );
```

9. Création de la table advisor

```
create table advisor
        (s_ID           varchar(5),  -- Identifiant de l'étudiant
        i_ID            varchar(5),  -- Identifiant de l'enseignant conseiller
        primary key (s_ID),  -- Clé primaire sur l'identifiant de l'étudiant
        foreign key (i_ID) references teacher (ID),  -- Clé étrangère vers l'enseignant
        foreign key (s_ID) references student (ID)  -- Clé étrangère vers l'étudiant
        );
```

10. Création de la table time_slot

```
create table time_slot
        (time_slot_id     varchar(4),  -- Identifiant du créneau horaire
        day               varchar(1),  -- Jour de la semaine (par exemple 'M' pour lundi)
        start_hr          numeric(2),  -- Heure de début
        start_min         numeric(2),  -- Minute de début
        end_hr            numeric(2),  -- Heure de fin
        end_min           numeric(2),  -- Minute de fin
        primary key (time_slot_id, day, start_hr, start_min)  -- Clé primaire composée de plusieurs
colonnes
        );
```

11. Création de la table prereq

```
create table prereq
        (course_id        varchar(8),  -- Identifiant du cours
        prereq_id         varchar(8),  -- Identifiant du cours prérequis
        primary key (course_id, prereq_id),  -- Clé primaire composée de plusieurs colonnes
        foreign key (course_id) references course,  -- Clé étrangère vers le cours
```

2/ Pour le peuplement : j'ai utilisé le script universityDB-data.sql.

Exercice 2 : Requêtes SQL

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

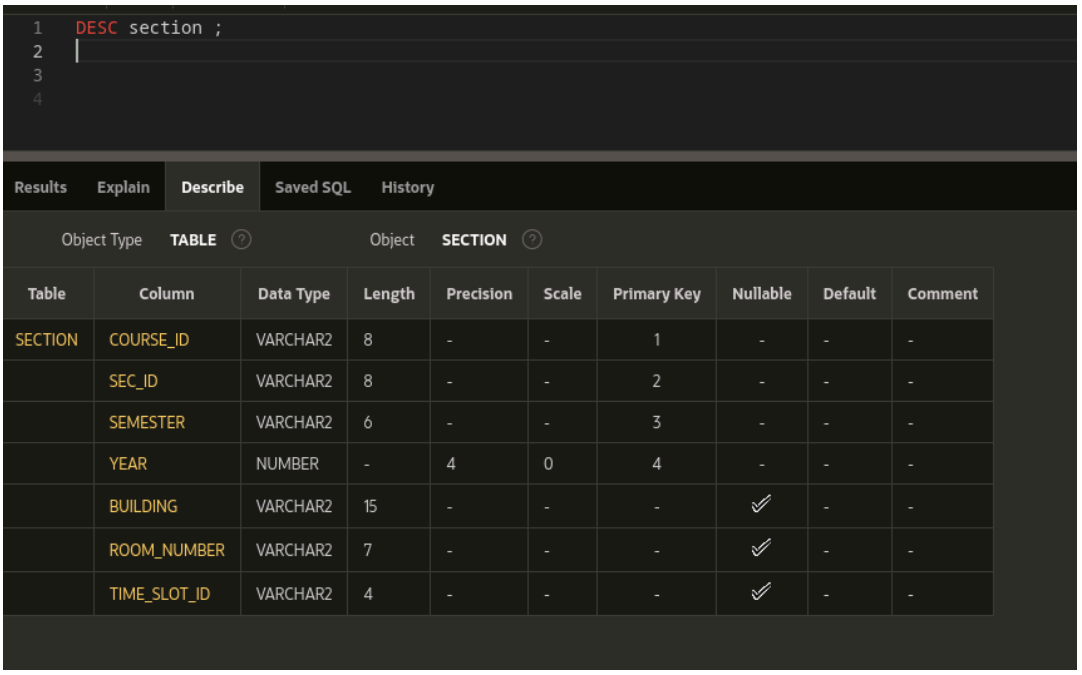


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SECTION	COURSE_ID	VARCHAR2	8	-	-	1	-	-	-
	SEC_ID	VARCHAR2	8	-	-	2	-	-	-
	SEMESTER	VARCHAR2	6	-	-	3	-	-	-
	YEAR	NUMBER	-	4	0	4	-	-	-
	BUILDING	VARCHAR2	15	-	-	-	✓	-	-
	ROOM_NUMBER	VARCHAR2	7	-	-	-	✓	-	-
	TIME_SLOT_ID	VARCHAR2	4	-	-	-	✓	-	-

```
SELECT * FROM section ;
```

Results | Explain | Describe | Saved SQL | History

COURSE_ID	SEC_ID	SEMESTER	YEAR	BUILDING	ROOM_NUMBER	TIME_SLOT_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

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

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

```
SELECT *
FROM course ;
```

Results | Explain | Describe | Saved SQL | History

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

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.02 seconds Download

3/ Afficher les titres des cours et les départements qui les proposent :

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

Results | Explain | Describe | Saved SQL | History

TITLE	DEPT_NAME
Intro. to Biology	Biology
Genetics	Biology
Computational Biology	Biology
Intro. to Computer Science	Comp. Sci.
Game Design	Comp. Sci.
Robotics	Comp. Sci.
Image Processing	Comp. Sci.
Database System Concepts	Comp. Sci.
Intro. to Digital Systems	Elec. Eng.
Investment Banking	Finance

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds Download

4/ Afficher les noms des départements ainsi que leur budget :

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

Results | Explain | Describe | Saved SQL | History

DEPT_NAME	BUDGET
Biology	90000
Comp. Sci.	100000
Elec. Eng.	85000
Finance	120000
History	50000
Music	80000
Physics	70000

7 rows returned in 0.02 seconds Download

5/ Afficher tous les noms des enseignants et leur département :

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

Results | Explain | Describe | Saved SQL | History

NAME	DEPT_NAME
Srinivasan	Comp. Sci.
Wu	Finance
Mozart	Music
Einstein	Physics
El Said	History
Gold	Physics
Katz	Comp. Sci.
Califieri	History
Singh	Finance
Crick	Biology

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.02 seconds Download

6/ Afficher tous les noms des enseignants ayant un salaire supérieur strictement à 65 000 $:

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

Results | Explain | Describe | Saved SQL | History

NAME
Wu
Einstein
Gold
Katz
Singh
Crick
Brandt
Kim

8 rows returned in 0.01 seconds Download

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;
```

Results | Explain | Describe | Saved SQL | History

NAME
Srinivasan
El Said
Katz
Califieri
Singh
Crick
Kim

7 rows returned in 0.00 seconds Download

8/ Afficher les noms des départements en utilisant la relation teacher et éliminer les doublons :

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

Results | Explain | Describe | Saved SQL | History

DEPT_NAME
Comp. Sci.
Biology
History
Finance
Elec. Eng.
Music
Physics

7 rows returned in 0.01 seconds Download

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.' ;
```

Results | Explain | Describe | Saved SQL | History

NAME
Katz
Brandt

2 rows returned in 0.01 seconds Download

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

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

Results | Explain | Describe | Saved SQL | History

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
CS-319	2	Spring	2010	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

7 rows returned in 0.02 seconds Download

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;
```

Results | Explain | Describe | Saved SQL | History

TITLE
Intro. to Computer Science
Game Design

2 rows returned in 0.01 seconds Download

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 ;
```

Results | Explain | Describe | Saved SQL | History

NAME	DEPT_NAME	BUILDING
Srinivasan	Comp. Sci.	Taylor
Wu	Finance	Painter
Mozart	Music	Packard
Einstein	Physics	Watson
El Said	History	Painter
Gold	Physics	Watson
Katz	Comp. Sci.	Taylor
Califieri	History	Painter
Singh	Finance	Painter
Crick	Biology	Watson
More than 10 rows available. Increase rows selector to view more rows.		

10 rows returned in 0.02 seconds Download

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.';
```

Results | Explain | Describe | Saved SQL | History

NAME
Brown
Zhang
Levy
Bourikas
Shankar
Williams

6 rows returned in 0.07 seconds Download

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' ;
```

Results | Explain | Describe | Saved SQL | History

NAME
Peltier

1 rows returned in 0.04 seconds Download

15/ Afficher tous les identifiants des cours et les enseignants qui les ont assurés :

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

Results | Explain | Describe | Saved SQL | History

NAME	COURSE_ID
Srinivasan	CS-101
Srinivasan	CS-315
Srinivasan	CS-347
Wu	FIN-201
Mozart	MU-199
Einstein	PHY-101
El Said	HIS-351
Katz	CS-101
Katz	CS-319
Crick	BIO-101

More than 10 rows available. Increase rows selector to view more rows.

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

Results | Explain | Describe | Saved SQL | History

COURSE_ID	SEC_ID	SEMESTER	YEAR	NUM_STUDENTS
CS-315	1	Spring	2010	2
CS-101	1	Spring	2010	1
CS-319	2	Spring	2010	1
CS-319	1	Spring	2010	1
FIN-201	1	Spring	2010	1
HIS-351	1	Spring	2010	1
MU-199	1	Spring	2010	1

7 rows returned in 0.01 seconds Download

17/ Afficher les noms des départements et les salaires maximum de leurs enseignants :

```
SELECT dept_name , max(salary) FROM teacher
GROUP BY dept_name ;
```

Results | Explain | Describe | Saved SQL | History

DEPT_NAME	MAX(SALARY)
Comp. Sci.	92000
Biology	72000
History	62000
Finance	90000
Elec. Eng.	80000
Music	40000
Physics	95000

7 rows returned in 0.01 seconds Download

18/ Afficher le nombre d'inscrits pour chaque enseignement proposé

```
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 ;
```

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
CS-101	1	Spring	2010	1
FIN-201	1	Spring	2010	1
BIO-101	1	Summer	2009	1
CS-315	1	Spring	2010	2
HIS-351	1	Spring	2010	1

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.02 seconds Download

19/ ffficher le nombre total de cours qui ont eu lieu dans chaque bˆatiment, pendant l'automne 2009et le printemps 2010.

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

Results | Explain | Describe | Saved SQL | History

BUILDING	COUNT(*)
Watson	3
Packard	4
Taylor	2
Painter	1

4 rows returned in 0.00 seconds Download

20/Afficher le nombre total de cours dispensés par chaque département et qui ont eu lieu dans le même bâtiment qui l'abrite :

```
SELECT department.dept_name,
       COUNT(*) AS num_sections
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;
```

Results | Explain | Describe | Saved SQL | History

DEPT_NAME	NUM_SECTIONS
Comp. Sci.	4
History	1
Elec. Eng.	1
Music	1
Physics	1

5 rows returned in 0.11 seconds Download

21/ Afficher les titres des cours proposés et qui ont eu lieu, ainsi que 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;
```

Results | Explain | Describe | Saved SQL | History

TITLE	NAME
Database System Concepts	Srinivasan
Game Design	Brandt
Game Design	Brandt
Genetics	Crick
Image Processing	Katz
Image Processing	Brandt
Intro. to Biology	Crick
Intro. to Computer Science	Srinivasan
Intro. to Computer Science	Katz
Intro. to Digital Systems	Kim

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.03 seconds Download

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

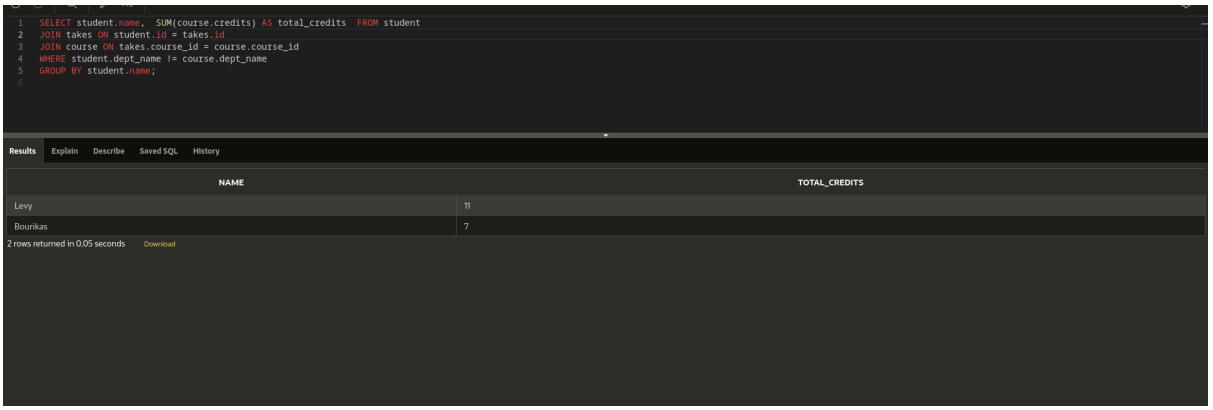
```
SELECT section.semester, COUNT(*) AS num_sections
FROM section GROUP BY section.semester;
```

Results | Explain | Describe | Saved SQL | History

SEMESTER	NUM_SECTIONS
Fall	3
Summer	2
Spring	10

3 rows returned in 0.01 seconds Download

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



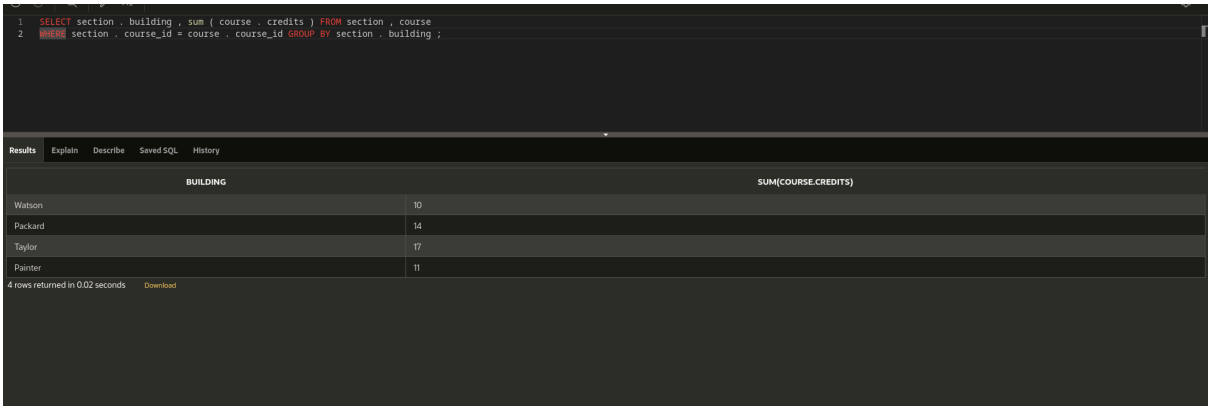
```
SELECT student.name,  SUM(course.credits) AS total_credits  FROM student
JOIN takes ON student.id = takes.id
JOIN course ON takes.course_id = course.course_id
WHERE student.dept_name != course.dept_name
GROUP BY student.name;
```

Results | Explain | Describe | Saved SQL | History

NAME	TOTAL_CREDITS
Levy	11
Bourikas	7

2 rows returned in 0.05 seconds Download

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
WHERE section . course_id = course . course_id GROUP BY section . building ;
```

Results | Explain | Describe | Saved SQL | History

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

4 rows returned in 0.02 seconds Download