

## **Dossier - Exploitation d'une base de données**

**BUT INFORMATIQUE - S2**

**SAE 2.04**

### **Exploitation d'une Base de Données**

Camélia ANTOINE - Sophie DONG - Léo-Paul CHAUVET

Université Sorbonne Paris Nord - IUT de Villetaneuse

Enseignant : Bouchaib Lemaire

Date rendu : 22 Mai 2023

# **SOMMAIRE**

## **I - Modélisation de données**

- 1 - Cahier des charges
- 2 - Modèles de données
- 3 - Règles de la base de données
- 4 - Script de création de la base de données
- 5 - Requêtes

## **II - Visualisations de données**

- 1 - Fonctions sur des étudiants
- 2 - Fonctions sur des groupes et la promotion
- 3 - Fonctions sur les enseignants
- 4 - Vue

## **III - Restrictions d'accès aux données**

- 1 - Règles d'accès aux données
- 2 - Procédures pour mettre en oeuvre ces règles
- 3 - Vue pour mettre en oeuvre ces règles

## **IV - Synthèse**

# I - Modélisation de données

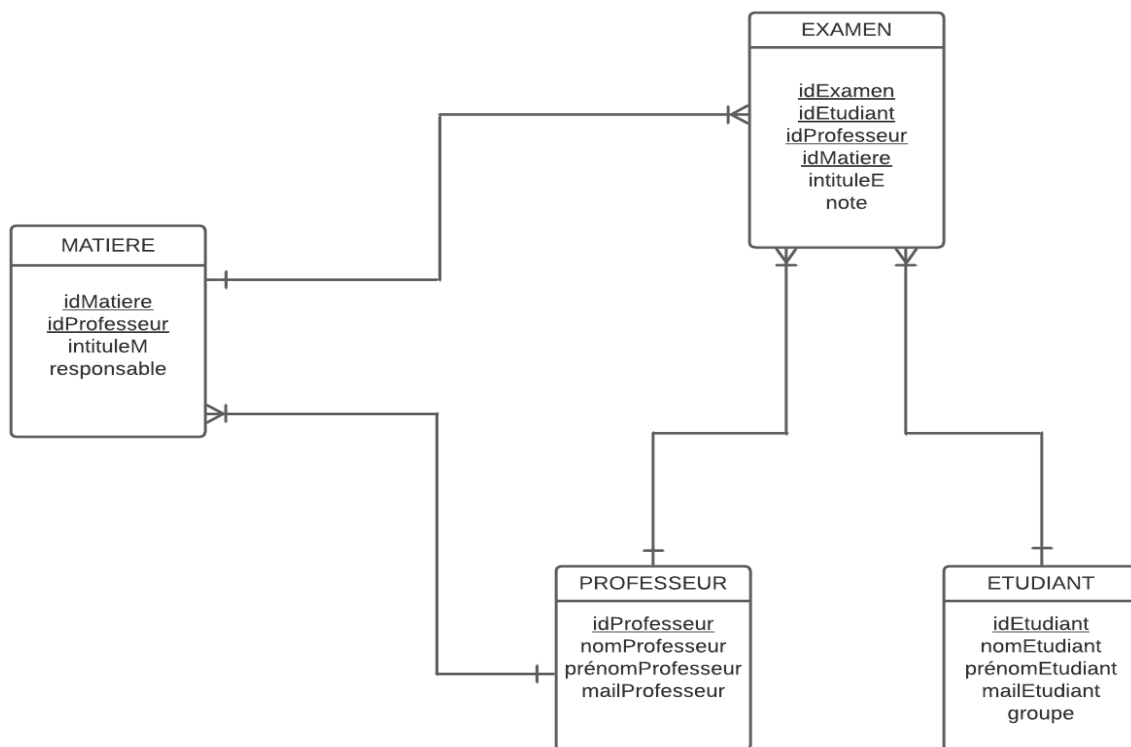
## 1 - Cahier des charges

Base de données composée de 4 tables :

**Légende** : \* correspond à une clé étrangère

- Table Professeur(idProfesseur (INTEGER), nomProfesseur (VARCHAR), prénomProfesseur (VARCHAR), mailProfesseur (VARCHAR))
- Table Etudiant(idEtudiant (INTEGER), nomEtudiant (VARCHAR), prénomEtudiant (VARCHAR), mailEtudiant (VARCHAR), groupe (VARCHAR))
- Table Matiere(idProfesseur\* (INTEGER) de la table PROFESSEURS, idMatière (VARCHAR), intituléM (VARCHAR), responsable (BOOLEAN))
- Table Examen(idProfesseur\* (INTEGER) de la table PROFESSEURS, idEtudiant\* (INTEGER) de la table ETUDIANTS, idExamen (INTEGER), idMatière (VARCHAR), note (FLOAT), intituléE (VARCHAR))

## 2 - Modèles de données (schéma canonique)



### **3 - Règles de la base de données**

La base de données possède trois groupes d'utilisateurs :

- Le groupe étudiant possède les droits suivants :
  - Table Professeur : Il n'a aucun droit et ne voit pas la table.
  - Table Étudiant : Il peut se voir, mais ne peut rien modifier.
  - Table Matière : Il peut voir la table, mais ne peut rien modifier.
  - Table Examen : Il peut voir la table, mais ne peut rien modifier.
- Le groupe professeurs, qui possède les droits suivants :
  - Table Professeur : Il peut voir la table, mais ne peut rien modifier.
  - Table Étudiant : Il peut voir la table, mais ne peut rien modifier.
  - Table Matière : Il peut voir la table, mais ne peut rien modifier.
  - Table Examen : Il a tous les droits.
- Le groupe Admin\_IUT, qui possède les droits suivants :
  - Table Professeur : Il a tous les droits.
  - Table Étudiant : Il a tous les droits.
  - Table Matière : Il a tous les droits.
  - Table Examen : Il a tous les droits.

### **4 - Script de création de la base de données**

```
CREATE TABLE Professeur(  
    idProfesseur INTEGER PRIMARY KEY,  
    nomProfesseur VARCHAR(100) NOT NULL,  
    prenomProfesseur VARCHAR(100) NOT NULL,  
    mailProfesseur VARCHAR(100) NOT NULL  
);  
  
CREATE TABLE Etudiant(  
    idEtudiant INTEGER PRIMARY KEY,  
    nomEtudiant VARCHAR(100) NOT NULL,  
    prenomEtudiant VARCHAR(100) NOT NULL,  
    mailEtudiant VARCHAR(100) NOT NULL,
```

```

    groupe VARCHAR(50) NOT NULL
);

CREATE TABLE Matiere(
    idProfesseur INTEGER REFERENCES PROFESSEUR ON DELETE CASCADE,
    idMatiere VARCHAR(100),
    intituleM VARCHAR(100) NOT NULL,
    responsable BOOLEAN,
    PRIMARY KEY(IdProfesseur, idMatiere)
);

CREATE TABLE Examen(
    idExamen VARCHAR(100),
    idProfesseur INTEGER REFERENCES PROFESSEUR ON DELETE CASCADE,
    idEtudiant INTEGER REFERENCES ETUDIANT ON DELETE CASCADE,
    idMatiere VARCHAR(100),
    intituleE VARCHAR(100) NOT NULL,
    note FLOAT NOT NULL,
    PRIMARY KEY (idExamen, idProfesseur, idEtudiant, idMatiere)
);

```

```

INSERT INTO Professeur(idProfesseur, nomProfesseur,
prenomProfesseur, mailProfesseur) VALUES
(12100050, 'Lemaire', 'Bouchaib',
'bouchaib.lemaire@univ.paris13.fr'),
(12100051, 'Hébert', 'David', 'hebert.iut@gmail.com'),
(12100052, 'Butelle', 'Franck',
'franck.butelle@lipn.univ.paris13.fr');

INSERT INTO Etudiant(idEtudiant, nomEtudiant, prenomEtudiant,
mailEtudiant, groupe) VALUES
(12200554, 'Dong', 'Sophie', 'sophie.dong@edu.univ.paris13.fr',
'Tlaloc'),
(12200001, 'Ate', 'Tom', 'tom.ate@edu.univ.paris13.fr', 'Tlaloc'),
(12200002, 'Boise', 'Fram', 'fram.boise@edu.univ.paris13.fr',
'Tlaloc'),
(12201175, 'Antoine', 'Camélia',
'camelia.antoine@edu.univ.paris13.fr', 'Indra'),
(12101312, 'Chauvet', 'Léo.Paul',
'paul.chauvet@edu.univ.paris13.fr', 'Shango');

```

```

INSERT INTO Matiere(idProfesseur, idMatiere, intituleM,
responsable) VALUES
(12100050, 'R2.01', 'Développement orienté objets', true),
(12100050, 'R2.02', 'Développement d applications avec IHM',
true),
(12100052, 'R2.03', 'Qualité de développement', false),
(12100052, 'R2.04.A', 'Communication et fonctionnement bas
niveaux', true),
(12100052, 'R2.04.B', 'Communication et fonctionnement bas
niveaux', true),
(12100052, 'R2.05', 'Introduction aux services réseaux', false),
(12100050, 'R2.06', 'Exploitation d une base de données', true),
(12100051, 'R2.07', 'Graphes', true),
(12100051, 'R2.08', 'Outils Numériques pour les Statistiques
descriptives', false),
(12100051, 'R2.09', 'Méthodes numériques', false),
(12100052, 'R2.10', 'Gestion de projet & des organisations',
false),
(12100050, 'R2.11', 'Droit des contrats et du numérique', false),
(12100052, 'R2.12', 'Anglais d entreprise', false),
(12100051, 'R2.13', 'Communication avec le milieu professionnel',
false),
(12100050, 'R2.14', 'PPP : Métiers de l informatique', false),
(12100050, 'S2.01', 'Développement d une application', true),
(12100051, 'S2.02', 'Exploration algorithmique d un problème',
false),
(12100052, 'S2.03', 'Installation de services réseau', true),
(12100050, 'S2.04', 'Exploitation d une base de données', true),
(12100051, 'S2.05', 'Gestion d un projet', false),
(12100051, 'S2.06', 'Organisation d un travail d équipe', false),
(12100050, 'P2.01', 'Portfolio', false);

INSERT INTO Examen(idExamen, idProfesseur, idEtudiant, idMatiere,
intituleE, note) VALUES
('1', 12100050, 12200554, 'R2.01', 'Interrogation', 20),
('2', 12100050, 12200554, 'R2.02', 'Examen', 16),
('3', 12100051, 12200554, 'R2.07', 'Interrogation', 14),
('4', 12100051, 12200554, 'R2.08', 'Examen', 17),
('5', 12100050, 12200554, 'R2.06', 'Examen', 10),
('6', 12100050, 12200554, 'R2.11', 'Examen', 16),
('7', 12100051, 12200554, 'R2.09', 'Examen', 13),

```

```

('8', 12100051, 12200554, 'R2.13', 'Examen', 19),
('9', 12100052, 12200554, 'R2.03', 'Examen', 15),
('10', 12100052, 12200554, 'R2.04.A', 'Examen', 14),
('11', 12100050, 12200554, 'R2.14', 'Examen', 19),
('12', 12100052, 12200554, 'R2.04.B', 'Examen', 16),
('13', 12100052, 12200554, 'R2.05', 'Examen', 15),
('14', 12100050, 12200554, 'S2.01', 'SAE', 19),
('15', 12100051, 12200554, 'S2.02', 'SAE', 20),
('16', 12100051, 12200554, 'S2.05', 'SAE', 17),
('17', 12100050, 12200554, 'S2.04', 'SAE', 9),
('18', 12100052, 12200554, 'S2.03', 'SAE', 17),
('19', 12100050, 12200554, 'P2.01', 'SAE', 20),

('1', 12100050, 12200001, 'R2.01', 'Interrogation', 8),
('2', 12100050, 12200001, 'R2.02', 'Examen', 18),
('3', 12100051, 12200001, 'R2.07', 'Interrogation', 4),
('4', 12100051, 12200001, 'R2.08', 'Examen', 7),
('5', 12100050, 12200001, 'R2.06', 'Examen', 1),
('6', 12100050, 12200001, 'R2.11', 'Examen', 6),
('7', 12100051, 12200001, 'R2.09', 'Examen', 7),
('8', 12100051, 12200001, 'R2.13', 'Examen', 1),
('9', 12100052, 12200001, 'R2.03', 'Examen', 15),
('10', 12100052, 12200001, 'R2.04.A', 'Examen', 4),
('11', 12100050, 12200001, 'R2.14', 'Examen', 1),
('12', 12100052, 12200001, 'R2.04.B', 'Examen', 8),
('13', 12100052, 12200001, 'R2.05', 'Examen', 15),
('14', 12100050, 12200001, 'S2.01', 'SAE', 10),
('15', 12100051, 12200001, 'S2.02', 'SAE', 0),
('16', 12100051, 12200001, 'S2.05', 'SAE', 17),
('17', 12100050, 12200001, 'S2.04', 'SAE', 5),
('18', 12100052, 12200001, 'S2.03', 'SAE', 12),
('19', 12100050, 12200001, 'P2.01', 'SAE', 10),

('1', 12100050, 12200002, 'R2.01', 'Interrogation', 15),
('2', 12100050, 12200002, 'R2.02', 'Examen', 7),
('3', 12100051, 12200002, 'R2.07', 'Interrogation', 4),
('4', 12100051, 12200002, 'R2.08', 'Examen', 7),
('5', 12100050, 12200002, 'R2.06', 'Examen', 1),
('6', 12100050, 12200002, 'R2.11', 'Examen', 19),
('7', 12100051, 12200002, 'R2.09', 'Examen', 7),
('8', 12100051, 12200002, 'R2.13', 'Examen', 1),

```

```

('9', 12100052, 12200002, 'R2.03', 'Examen', 2),
('10', 12100052, 12200002, 'R2.04.A', 'Examen', 4),
('11', 12100050, 12200002, 'R2.14', 'Examen', 1),
('12', 12100052, 12200002, 'R2.04.B', 'Examen', 8),
('13', 12100052, 12200002, 'R2.05', 'Examen', 11),
('14', 12100050, 12200002, 'S2.01', 'SAE', 10),
('15', 12100051, 12200002, 'S2.02', 'SAE', 0),
('16', 12100051, 12200002, 'S2.05', 'SAE', 5),
('17', 12100050, 12200002, 'S2.04', 'SAE', 5),
('18', 12100052, 12200002, 'S2.03', 'SAE', 10),
('19', 12100050, 12200002, 'P2.01', 'SAE', 9),

('1', 12100050, 12201175, 'R2.01', 'Interrogation', 18),
('2', 12100050, 12201175, 'R2.02', 'Examen', 16),
('3', 12100051, 12201175, 'R2.07', 'Interrogation', 15),
('4', 12100051, 12201175, 'R2.08', 'Examen', 11),
('5', 12100050, 12201175, 'R2.06', 'Examen', 15),
('6', 12100050, 12201175, 'R2.11', 'Examen', 17),
('7', 12100051, 12201175, 'R2.09', 'Examen', 18),
('8', 12100051, 12201175, 'R2.13', 'Examen', 13),
('9', 12100052, 12201175, 'R2.03', 'Examen', 15),
('10', 12100052, 12201175, 'R2.04.A', 'Examen', 17),
('11', 12100050, 12201175, 'R2.14', 'Examen', 15),
('12', 12100052, 12201175, 'R2.04.B', 'Examen', 15),
('13', 12100052, 12201175, 'R2.05', 'Examen', 14),
('14', 12100050, 12201175, 'S2.01', 'SAE', 17),
('15', 12100051, 12201175, 'S2.02', 'SAE', 20),
('16', 12100051, 12201175, 'S2.05', 'SAE', 18),
('17', 12100050, 12201175, 'S2.04', 'SAE', 18),
('18', 12100052, 12201175, 'S2.03', 'SAE', 17),

('1', 12100050, 12101312, 'R2.01', 'Interrogation', 18),
('2', 12100050, 12101312, 'R2.02', 'Examen', 16),
('3', 12100051, 12101312, 'R2.07', 'Interrogation', 15),
('4', 12100051, 12101312, 'R2.08', 'Examen', 7),
('5', 12100050, 12101312, 'R2.06', 'Examen', 15),
('6', 12100050, 12101312, 'R2.11', 'Examen', 17);

```



## 5 - Requête

Requête n°1 :

```
SELECT * FROM Matiere;
```

idprofesseur	idmatiere	intitulem	responsable
12100050	R2.01	Développement orienté objets	t
12100050	R2.02	Développement d applications ave...	t
12100052	R2.03	Qualité de développement	f
12100052	R2.04.A	Communication et fonctionnement ...	t
12100052	R2.04.B	Communication et fonctionnement ...	t
12100052	R2.05	Introduction aux services réseaux	f
12100050	R2.06	Exploitation d une base de données	t
12100051	R2.07	Graphes	t
12100051	R2.08	Outils Numériques pour les Statisti...	f
12100051	R2.09	Méthodes numériques	f
12100052	R2.10	Gestion de projet & des organisati...	f
12100050	R2.11	Droit des contrats et du numérique	f
12100052	R2.12	Anglais d entreprise	f
12100051	R2.13	Communication avec le milieu prof...	f
12100050	R2.14	PPP : Métiers de l informatique	f
12100050	S2.01	Développement d une application	t
12100051	S2.02	Exploration algorithmique d un pro...	f
12100052	S2.03	Installation de services réseau	t
12100050	S2.04	Exploitation d une base de données	t
12100051	S2.05	Gestion d un projet	f
12100051	S2.06	Organisation d un travail d équipe	f
12100050	P2.01	Portfolio	f

Cette requête renvoie au tableau des matières où 't' (true) signifie que le professeur est responsable.

Requête n°2 :

```
SELECT nomEtudiant, prenomEtudiant, AVG(note) AS moyenne
```

```
FROM Etudiant JOIN Examen USING (idEtudiant)
GROUP BY nomEtudiant, prenomEtudiant ;
```

i	nometudiant	prenometudiant	moyenne
	Antoine	Camélia	16.055555555555557
	Boise	Fram	6.631578947368421
	Ate	Tom	7.842105263157895
	Dong	Sophie	16.105263157894736
	Chauvet	Léo.Paul	14.666666666666666

Cette requête renvoie un tableau comportant les noms et prénoms des étudiants ainsi que leur moyenne générale respective.

Requête n°3 :

```
SELECT idMatiere, nomEtudiant, prenomEtudiant, AVG(note) AS
moyenneM
FROM Etudiant JOIN Examen USING (idEtudiant)
WHERE Examen.idMatiere = 'R2.08'
GROUP BY idMatiere, nomEtudiant, prenomEtudiant;
```

i	idmatiere	nometudiant	prenometudiant	moyennem
	R2.08	Antoine	Camélia	11
	R2.08	Ate	Tom	7
	R2.08	Boise	Fram	7
	R2.08	Chauvet	Léo.Paul	7
	R2.08	Dong	Sophie	17

Cette requête renvoie les moyennes de chaque élève dans la matière R2.08.

Requête n°4 :

```
SELECT nomProfesseur, prenomProfesseur, idMatiere, intituleM
FROM Professeur JOIN Matiere USING (idProfesseur)
WHERE Professeur.idProfesseur = '12100050' AND Matiere.responsable
= true;
```

i	nomprofesseur	prenomprofesseur	idmatiere	intitulem
	Lemaire	Bouchaib	R2.01	Développement orienté objets
	Lemaire	Bouchaib	R2.02	Développement d applications avec IHM
	Lemaire	Bouchaib	R2.06	Exploitation d une base de données
	Lemaire	Bouchaib	S2.01	Développement d une application
	Lemaire	Bouchaib	S2.04	Exploitation d une base de données

Cette requête renvoie toutes les matières et SAE dont monsieur Lemaire est responsable.

Requête n°5 :

```
SELECT idExamen, nomEtudiant, prenomEtudiant, intituleE, note
FROM Etudiant JOIN Examen USING (idEtudiant)
WHERE Examen.intituleE = 'SAE' AND Examen.note>17
ORDER BY note ASC;
```

i	idexamen	nometudiant	prenometudiant	intitulee	note
17		Antoine	Camélia	SAE	18
16		Antoine	Camélia	SAE	18
14		Dong	Sophie	SAE	19
19		Dong	Sophie	SAE	20
15		Dong	Sophie	SAE	20
15		Antoine	Camélia	SAE	20

Cette requête renvoie un tableau comportant les noms et prénoms des étudiants ayant eu plus de 17 à une SAE ainsi que l'id de cet examen. Elle nous permet de visualiser qui parmi les élèves a le mieux réussi ses SAE, et donc de savoir qui sont les têtes de classe.

Requête n°6 :

```
SELECT * FROM (SELECT ETUDIANT.groupe,
(SUM(EXAMEN.note)/COUNT(EXAMEN.idExamen)) AS moyennes
FROM ETUDIANT NATURAL JOIN EXAMEN
GROUP BY ETUDIANT.groupe) AS moyennes
ORDER BY moyennes;
```

groupe	moyennes
Tlaloc	10.192982456140351
Shango	14.666666666666666
Indra	16.055555555555557

Cette requête renvoie au classement des groupes du meilleur au moins bon (grâce à la moyenne de groupe).

## II - Visualisations de données

### 1 - Fonctions sur des étudiants

Relevé des notes d'un étudiant :

```
CREATE OR REPLACE FUNCTION Releve_Notes(in idEtudiant int, out
nomEtudiant varchar, out prenomEtudiant varchar, out groupe
varchar, out idMatiere varchar,out intituleE varchar,out Note
float)
RETURNS SETOF record AS
$$
SELECT Etudiant.nomEtudiant, prenomEtudiant, groupe, idMatiere,
intituleE, note
FROM Examen JOIN Etudiant ON Examen.IdEtudiant =
Etudiant.IdEtudiant
WHERE Etudiant.IdEtudiant=$1
ORDER BY idExamen;
$$language SQL;

SELECT * FROM Releve_Notes(12201175);
```

nometudiant	prenometudiant	groupe	idmatiere	intitulee	note
Antoine	Camélia	Indra	R2.01	Interrogation	18
Antoine	Camélia	Indra	R2.04.A	Examen	17
Antoine	Camélia	Indra	R2.14	Examen	15
Antoine	Camélia	Indra	R2.04.B	Examen	15
Antoine	Camélia	Indra	R2.05	Examen	14
Antoine	Camélia	Indra	S2.01	SAE	17
Antoine	Camélia	Indra	S2.02	SAE	20
Antoine	Camélia	Indra	S2.05	SAE	18
Antoine	Camélia	Indra	S2.04	SAE	18
Antoine	Camélia	Indra	S2.03	SAE	17

Meilleure note obtenue par un étudiant :

```
CREATE OR REPLACE FUNCTION Meilleure_Note(in idEtudiant int, out
nomEtudiant varchar, out prenomEtudiant varchar, out note float)
RETURNS SETOF record AS
$$
SELECT Etudiant.nomEtudiant, prenomEtudiant, MAX(note)
FROM Examen JOIN Etudiant ON Examen.idEtudiant=Etudiant.idEtudiant
WHERE Etudiant.idEtudiant=$1
GROUP BY Etudiant.idEtudiant
$$language SQL;

SELECT * FROM Meilleure_note(12201175);
```

nometudiant	prenometudiant	note
Antoine	Camélia	20

Relevé des moyennes d'un étudiant dans chaque matière :

```
CREATE OR REPLACE FUNCTION Releve_Moyenne(in idEtudiant int, out
idMatiere varchar,out moyenne float)
RETURNS SETOF record AS
$$
SELECT idMatiere, AVG(note)
FROM Examen
GROUP BY Examen.idMatiere
ORDER BY Examen.idMatiere
$$language SQL;

SELECT * FROM Releve_Moyenne(12201175);
```

idmatiere	moyenne
P2.01	13
R2.01	15.8
R2.02	14.6
R2.03	11.75
R2.04.A	9.75
R2.04.B	11.75
R2.05	13.75
R2.06	8.4
R2.07	10.4
R2.08	9.8
R2.09	11.25
R2.11	15
R2.13	8.5
R2.14	9
S2.01	14
S2.02	10
S2.03	14
S2.04	9.25
S2.05	14.25

Moyenne général d'un étudiant :

```
CREATE OR REPLACE FUNCTION Moyenne_G(in idEtudiant int, out
moyenne float)
RETURNS float AS
$$
SELECT AVG(note)
FROM Examen WHERE Examen.idEtudiant=$1
$$language SQL;

SELECT * FROM Moyenne_G(12101312);
```

moyenne
14.666666666666666

## 2 - Fonctions sur des groupes et la promotion

Moyenne général d'un groupe :

```
CREATE OR REPLACE FUNCTION Moyenne_Groupe(inout groupe varchar,
out moyenne float)
RETURNS SETOF record AS
$$
SELECT groupe, AVG(note)
FROM Etudiant JOIN Examen ON Examen.idEtudiant=Etudiant.idEtudiant
WHERE Etudiant.groupe=$1
GROUP BY groupe
$$language SQL;

SELECT * FROM Moyenne_Groupe('Tlaloc');
```

groupe	moyenne
Tlaloc	10.192982456140351

Les étudiants du groupe dans l'ordre alphabétique :

```
CREATE OR REPLACE FUNCTION Etudiants_Du_Groupe(in groupe varchar, out
nomEtudiant varchar, out prenomEtudiant varchar)
RETURNS SETOF record AS
$$
SELECT nomEtudiant, prenomEtudiant
FROM Etudiant
WHERE Etudiant.groupe=$1
GROUP BY nomEtudiant, prenomEtudiant
ORDER BY nomEtudiant
$$language SQL;

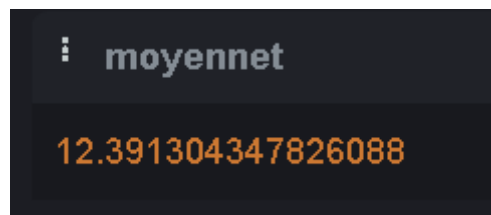
SELECT * FROM Etudiants_Du_Groupe('Tlaloc');
```

nometudiant	prenometudiant
Ale	Tom
Boise	Fram
Dong	Sophie

Moyenne de la promotion dans un type d'examen :

```
CREATE OR REPLACE FUNCTION Moyenne_Promo_Type_Examen(in intituleE
varchar, out moyennet float)
RETURNS SETOF float AS
$$
SELECT AVG(note)
FROM Examen
WHERE Examen.intituleE=$1
$$language SQL;

SELECT moyennet FROM Moyenne_Promo_Type_Examen('SAE');
```



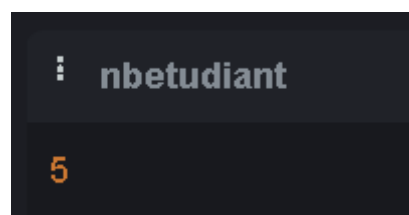
A screenshot of a database query result. It shows a column header 'moyennet' and a single row with the value '12.391304347826088'.

moyennet
12.391304347826088

Nombre d'étudiants dans la promotion :

```
CREATE OR REPLACE FUNCTION NbEtudiant_Promo(out nbEtudiant int)
RETURNS int AS
$$
SELECT COUNT(DISTINCT(idEtudiant))
FROM Etudiant
$$ LANGUAGE sql;

SELECT * FROM NbEtudiant_Promo();
```



A screenshot of a database query result. It shows a column header 'nbetudiant' and a single row with the value '5'.

nbetudiant
5



### 3 - Fonctions sur les enseignants

Matières dont un enseignant est responsable :

```
CREATE OR REPLACE FUNCTION Responsable_Matiere(in idProfesseur
int, out matieres VARCHAR)
RETURNS SETOF varchar AS
$$
SELECT idmatiere
FROM matiere
WHERE matiere.idProfesseur=$1 AND responsable=true ;
$$ LANGUAGE sql;

SELECT * FROM Responsable_Matiere(12100052);
```

**i matiere**

R2.04.A

R2.04.B

S2.03

Nombre de copies corrigées par un enseignant :

```
CREATE OR REPLACE FUNCTION Nb_Copie(in idProfesseur int, out
Nb_Copie int)
RETURNS int AS
$$
SELECT COUNT(idProfesseur)
FROM Examen
WHERE Examen.idProfesseur=$1;
$$ LANGUAGE sql;

SELECT * FROM Nb_Copie(12100051);
```

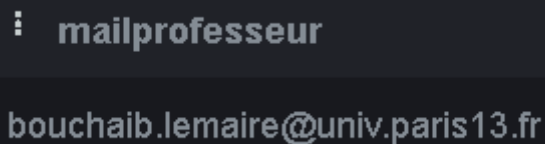
**i nb\_copie**

26

Mail d'un enseignant :

```
CREATE OR REPLACE FUNCTION Mail_Enseignant(in nomprofesseur
VARCHAR, in prenomprofesseur VARCHAR, out mailprofesseur VARCHAR)
RETURNS varchar AS
$$
SELECT mailprofesseur
FROM professeur
WHERE          professeur.nomprofesseur=$1          AND
professeur.prenomprofesseur=$2;
$$ LANGUAGE sql;

SELECT * FROM Mail_Enseignant('Lemaire', 'Bouchaib');
```



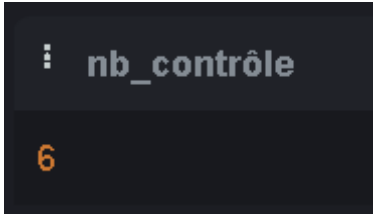
The screenshot shows a query result with two columns. The first column is labeled 'mailprofesseur' and the second column contains the email address 'bouchaib.lemaire@univ.paris13.fr'.

mailprofesseur
bouchaib.lemaire@univ.paris13.fr

Nombre de contrôle qu'un enseignant a organisé :

```
CREATE OR REPLACE FUNCTION Controle_Organise(in idProfesseur int,
out nb_contrôle int)
RETURNS int AS
$$
SELECT      COUNT(DISTINCT(idExamen))      FROM      Examen      WHERE
Examen.idProfesseur=$1;
$$ LANGUAGE sql;

SELECT * FROM Controle_Organise(12100051);
```



The screenshot shows a query result with one column labeled 'nb\_contrôle' containing the value '6'.

nb_contrôle
6

## 4 - Vue

```
CREATE VIEW moyPerExam AS
SELECT idExamen, idMatiere, intituleE, SUM(note)/COUNT(idEtudiant)
AS moy
FROM Examen
GROUP BY idExamen, idMatiere, intituleE
ORDER BY idExamen;

SELECT * FROM moyPerExam;
```

idexamen	idmatiere	intitulee	moy
1	R2.01	Interrogation	15.8
10	R2.04.A	Examen	9.75
11	R2.14	Examen	9
12	R2.04.B	Examen	11.75
13	R2.05	Examen	13.75
14	S2.01	SAE	14
15	S2.02	SAE	10
16	S2.05	SAE	14.25
17	S2.04	SAE	9.25
18	S2.03	SAE	14
19	P2.01	SAE	13
2	R2.02	Examen	14.6
3	R2.07	Interrogation	10.4
4	R2.08	Examen	9.8
5	R2.06	Examen	8.4
6	R2.11	Examen	15
7	R2.09	Examen	11.25
8	R2.13	Examen	8.5
9	R2.03	Examen	11.75

### III - Restrictions d'accès aux données

#### 1 - Règles d'accès aux données

```
CREATE user "12100050" WITH PASSWORD '0001';
CREATE user "12100051" WITH PASSWORD '0002';
CREATE user "12100052" WITH PASSWORD '0003';

CREATE user "12201175" WITH PASSWORD '1000';
CREATE user "12200554" WITH PASSWORD '2000';
CREATE user "12101312" WITH PASSWORD '3000';
CREATE user "12200001" WITH PASSWORD '4000';
CREATE user "12200002" WITH PASSWORD '5000';

CREATE user Admin_IUT WITH PASSWORD '1234';

CREATE GROUP role_professeur WITH USER "12100050", "12100051",
"12100052";
CREATE GROUP role_etudiant WITH USER "12201175", "12200554",
"12101312", "12200001", "12200002";

GRANT SELECT ON Matiere TO role_etudiant;
GRANT SELECT ON Examen TO role_etudiant;

GRANT SELECT ON Professeur TO role_professeur;
GRANT SELECT ON Etudiant TO role_professeur;
GRANT SELECT ON Matiere TO role_professeur;
GRANT SELECT, UPDATE, INSERT, DELETE ON Examen TO role_professeur;

GRANT all privileges ON Professeur TO Admin_IUT;
GRANT all privileges ON Etudiant TO Admin_IUT;
GRANT all privileges ON Matiere TO Admin_IUT;
GRANT all privileges ON Examen TO Admin_IUT;
```

Ce script nous permet de créer les différents utilisateurs et de les ajouter dans deux groupes différents : role\_professeur et role\_etudiant. Chacun de ces groupes a des droits différents sur les tables. Le rôle de superadmin appelé Admin\_IUT, a quant à lui tous les droits sur chaque table.

## **2 - Procédures pour mettre en oeuvre ces règles**

Fonction n°1 :

```
CREATE or REPLACE FUNCTION Changer_Note(in note float, in
idEtudiant int, in idMatiere varchar, in intituleE varchar)
RETURNS float AS
$$
UPDATE Examen SET note=$1
WHERE $2 = Examen.idEtudiant and
Examen.idProfesseur ::varchar(15) = session_user AND
Examen.idMatiere = $3 AND
Examen.intituleE = $4 RETURNING note;
$$language sql SECURITY DEFINER;
```

```
SELECT * FROM Changer_Note(16,12200554,'R2.01','Interrogation');
```

Cette fonction permet à un enseignant de modifier la note d'un élève grâce à l'id matière et l'intitulé de l'examen.

Fonction n°2 :

```
CREATE or REPLACE FUNCTION MesResultats(out intituleM varchar, out
idMatiere varchar, out intituleE varchar, out note float)
RETURNS SETOF record AS $$
SELECT intituleM, idMatiere, intituleE, note
FROM Matiere NATURAL JOIN Examen
WHERE Examen.idEtudiant ::varchar(15) = session_user;
$$language SQL SECURITY DEFINER;
```

```
SELECT * FROM MesResultats();
```

Cette fonction permet à un élève d'accéder à son propre relevé de notes.

Fonction n°3 :

```
CREATE OR REPLACE FUNCTION mat(out idMatiere varchar, out
intituleM varchar )
RETURNS SETOF record AS
$$
SELECT idMatiere,intituleM
FROM Matiere
WHERE idProfesseur ::varchar(15) = session_user;
$$language SQL SECURITY DEFINER;
```

```
SELECT * FROM mat();
```

Cette fonction donne l'id de la matière et son intitulé si le professeur est lié à celle-ci indépendamment du fait qu'il en soit le responsable ou non.

### **3 - Vue pour mettre en oeuvre ces règles**

```
CREATE VIEW Test AS
SELECT * FROM Etudiant
WHERE idEtudiant ::varchar(15) = session_user;
```

```
SELECT * FROM Test;
```

Cette vue permet à l'étudiant qui est l'utilisateur actuel de voir toutes ses données.

## IV - Synthèse

Dans un premier temps, nous avons créé une database en s'appuyant sur le cahier des charges et le schéma. Elle est composée des tables Professeur, Etudiant, Matiere, Examen. Pour les tables Etudiant et Professeur, nous avons décidé de mettre peu de personnes afin que cela soit plus lisible. De plus, nous avons effectué plusieurs requêtes afin de permettre une bonne visualisation de cette database.

Dans un second temps, afin de visualiser les données, nous avons créé plusieurs fonctions ciblées. Il y a des fonctions sur les étudiants, les groupes, la promotion et les enseignants. Pour finir, une vue a été créée. Toutes ces fonctions pourraient être utiles dans le cadre de l'IUT.

Dans un dernier temps, les règles de la base de données décidées dans la première partie ont été mises en place. Les utilisateurs ont été créés ainsi que leur groupe et leurs privilèges. Enfin, des fonctions et une vue mettant en œuvre ces règles ont été créées.