Nom : HATEM
Prénom :Kenza
Groupe : Zeus

22 mai 2023

S2.04 Exploitation d'une Base de Données

Enseignant Responsable : Mr David BUSCALDI

IUT Villetaneuse

Université Sorbonne Paris Nord

Ce projet a pour objectif :

1. L'étude d'un modèle de données pour mettre en place une base de données de gestion des notes des étudiants en BUT

2. L'étude et la mise en œuvre de la gestion des données dérivées : relevé de notes, bilans.

3. L'étude et la mise en œuvre des restrictions d'accès à ces données : étudiant, enseignant, responsable de matière.

Il sera organisé comme ce qui suit :

I. Modélisation des données

1. Le cahier des charges

2. La base de données

a. Script de création des tables
b. Exemple d'illustration
c. Script de peuplement des tables

II. Visualisation des données

1. Procédures :

a. Notes_etudiant
b. Moyenne_module_etudiant
c. Moyenne_module
d. Notes_groupe
e. Moyenne_groupe
f. Note_modulaire

2. Vues :

a. Etudiants_Zeus
b. Etudiants_Shango
c. Etudiants_Tlaloc
d. Etudiants_Whaitiri
e. Moyennes_tous_Etudiants
f. Etudiants_Avec_Moyenne_Sup_10
g. Etudiants_Redoubles
h. Moyenne_modules
i. Moyenne_groupes

III. Restriction d'accès aux données

1. Les droits d'accès aux tables

2. Le script de la restriction des données

a. Le rôle etudiant
b. Le rôle professeur
c. Le rôle adiministration

3. Illustration des rôles créés

4. Procédures pour mettre en œuvre ces restrictions

a. Changer_la_note
b. Mes_notes
c. Ma_note

I. Modélisation des Données :

1. Le cahier des charges :

La SAE 204 contribue à 40% de la compétence 4 : Gérer des données de l'information.

Le système de gestion de données utilisé durant ce projet est POSTGRESQL.

Le temps consacré pour cette SAE et de 4 séances de 2heures, ce qui fait 8 heures au total.

Dans ce projet :

A partir d'un besoin exprimé on va mettre en place et structurer une base de données de gestion des notes des étudiants en BUT.

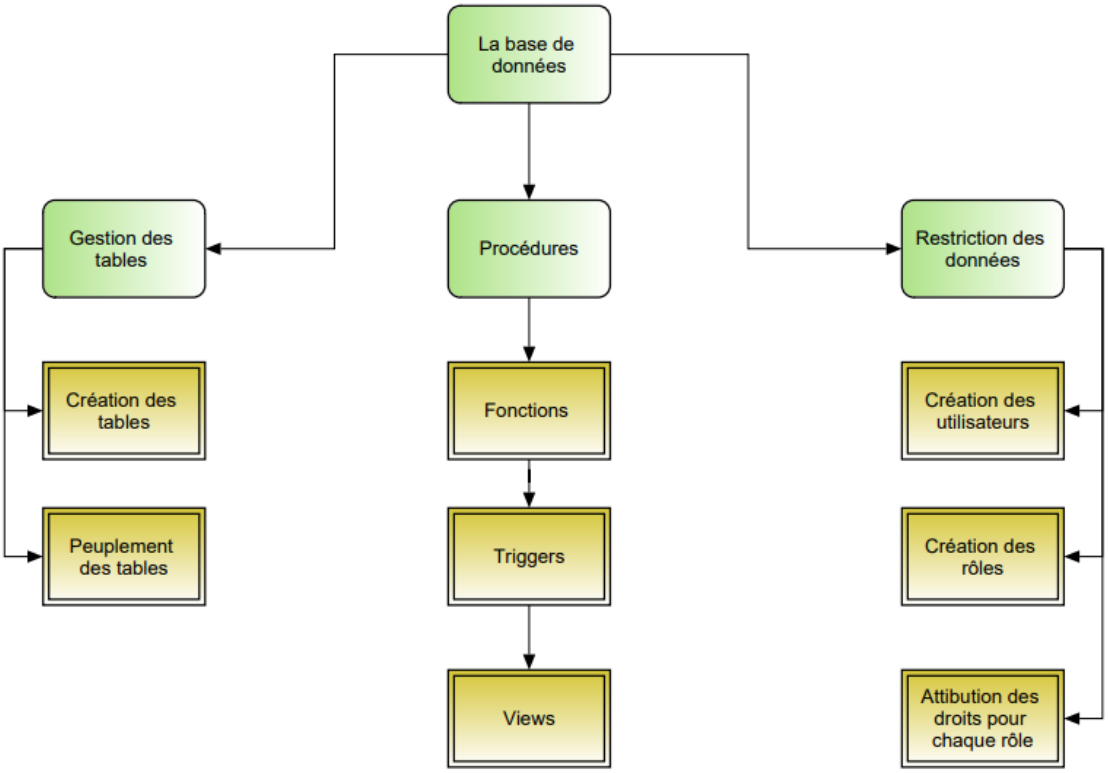
Réaliser plusieurs procédures qui vont permettre de visualiser les données en utilisant SQL et PSPGSQL.

Réaliser plusieurs essais sur la base de données.

Respecter le droit d'accès aux données par les différents utilisateurs.

Fournir un guide d'utilisation simple mais pertinent pour l'entreprise pour décrire les besoins du projet en fournissant un script pour la totalité du projet.

Un diagramme résumant les différentes étapes du projet :



2- La base de données :

La table Professeur :

a) idP : entier(integer) qui contient l'identifiant du professeur.
b) Nom : chaine de caractère (varchar) qui contient le nom du professeur.
c) Prenom : chaine de caractère (varchar) qui contient le prénom du professeur.
d) Mail : chaine de caractère (varchar) qui contient le mail du professeur.

La table Professeur a comme clé primaire : idP.

La table Etudiant :

a) idE : entier(integer) qui contient l'identifiant de l'étudiant.
b) idD : entier (integer) qui fait référence à la table département et qui contient l'identifiant du département de l'étudiant.
c) Nom : chaine de caractère (varchar) qui contient le nom de l'étudiant.
d) Prenom : chaine de caractère (varchar) qui contient le prénom de l'étudiant.
e) Mail : chaine de caractère (varchar) qui contient le mail de l'étudiant.
f) Groupe : chaine de caractère (varchar) qui contient le groupe de l'étudiant.

La table Etudiant a comme clé primaire : idE.

La table Module :

a) idM : entier(integer) qui contient l'identifiant du module.
b) idP : entier(integer) fait référence à la table Professeur et qui contient le professeur responsable du module.
c) Nom : une chaine de caractère (varchar) et qui contient le nom de la matière.
d) idD : un entier (integer) qui fait référence à la table Département et qui contient l'identifiant du département.

La table Module a comme clé primaire idM.

La table Controle :

a) idC : entier (integer) qui contient l'identifiant du contrôle.
b) idP : entier(integer) qui fait référence à la table Professeur et contient l'identifiant du professeur qui a fait le contrôle.
c) idM : entier (integer) qui fait référence à la table Module et contient l'identifiant du module concerné par le contrôle.
d) idE : entier(integer) qui fait référence à la table Etudiant et contient l'identifiant de l'étudiant qui a passé le contrôle.
e) Nom : chaine de caractère(varchar) et contient le nom du contrôle.
f) Note : float qui contient la note de l'étudiant.

La table Controle a comme clé primaire (idC , idP , IdM , IdE).

La table Departement :

a) idD : entier (integer) qui contient l'identifiant du département.

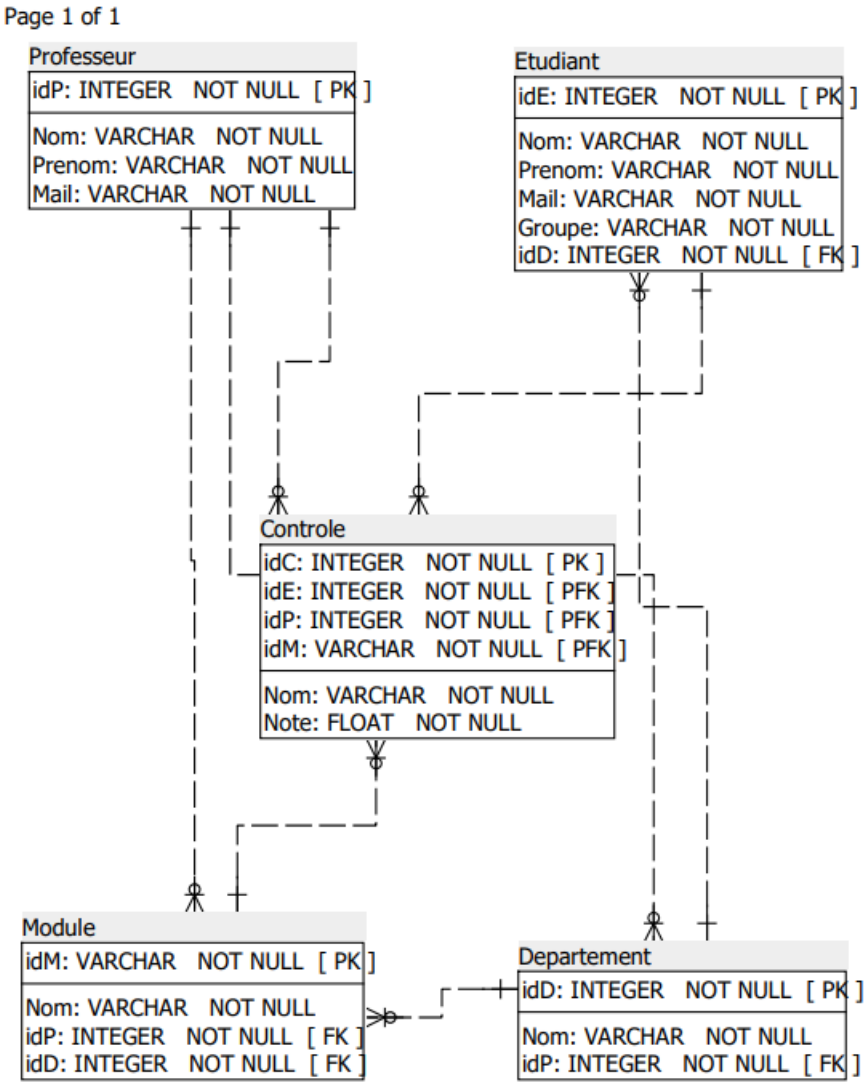
b) idP : entier (integer) qui fait référence à la table Professeur qui contient l'identifiant du professeur responsable du département.
c) Nom : chaine de caractère(varchar) et contient le nom du département.

La table Departement a comme clé primaire idD.

1- Script de création des tables :

```
CREATE TABLE Professeur(
IdP integer primary key,
Nom varchar ,
Prenom varchar  ,
Mail varchar
);

CREATE TABLE Etudiant(
IdE integer primary key,
Nom varchar(100),
Prenom varchar(100),
Groupe varchar(100),
Mail varchar(100)
);

CREATE TABLE Module(
IdM varchar primary key,
Nom varchar(50),
idP integer references Professeur on delete set null ,
idD integer references Departement on delete set null
);

CREATE TABLE Controle(
IdP integer references Professeur(IdP) on delete set null ,
IdE integer references Etudiant(IdE) on delete set null,
IdC integer,
IdM varchar(100) references Module(idM) on delete set null,
Nom varchar(100),
Note float ,
Primary key (idP, idE , idM , idC)) ;

CREATE TABLE Departement(
IdD integer primary key,
idP integer references Professeur(idP) ,
Nom varchar(100)
);
```

2- Exemple d'illustration :



3- Script de peuplement des tables :

```
INSERT INTO Etudiant(idE , Nom , Prenom ,Groupe , Mail) VALUES
(1220001 , 'Mazraoui' , 'Karim' ,'zeus' , 'karim.mazraoui@univ-paris13.fr')
,
(1220002 , 'Dupont' , 'Marine' , 'zeus' ,'marine.dupont@univ-paris13.fr') ,
(1220003 , 'Saka' , 'Pierre' ,'shango' , 'pierre.saka@univ-paris13.fr') ,
(1220004 , 'Barbara' , 'Selma' ,'tlaloc', 'selma.barbara@univ-paris13.fr')
,
(1220005 , 'Makassou' , 'Fatima' ,'whaitiri', 'fatima.makassou@univ-
paris13.fr') ;

INSERT INTO Professeur VALUES
(1440001 , 'Lacroix' , 'Mathieux' , 'mathieux.lacroix@univ-paris13.fr') ,
(1440002 , 'Ellouze' , 'Slim' , 'slim.ellouze@univ-paris13.fr') ,
```

```
(1440003 , 'Zetlaoui' , 'Eleonore' , 'eleonore.zetlaoui@univ-paris13.fr') ,
(1440004 , 'Martinez' , 'Antony' , 'antony.martinez@univ-paris13.fr') ,
(1440005 , 'Bacher' , 'Axel' , 'axel.bacher@univ-paris13.fr') ,
(1440006 , 'Azzag' , 'Hanane' , 'hanane.azzag@univ-paris13.fr'),
(1440007 , 'Hébert' , 'David' , 'david.hebert@univ-paris13.fr'),
(1440008 , 'Buscaldi' , 'David' , 'david.buscaldi@univ-paris13.fr'),
(1440009 , 'Bjerrum' , 'Marie' , 'marie.bjerrum@univ-paris13.fr'),
(1440010 , 'Couleau' , 'Christelle' , 'christelle.couleau@univ-
paris13.fr');

INSERT INTO Departement(idD , idP , Nom) VALUES
(01 , 1440007 , 'Mécanique') ,
(02 , 1440007 , 'Informatique') ,
(03 , 1440009 , 'Automatique') ;

INSERT INTO Module(IdP, IdM, Nom, idD)
VALUES(1440001, 'R1-01', 'Initiation au développement',02),
(1440002, 'R1-02', 'Developpement interface web',02),
(1440008, 'R1-05', 'Introduction aux bases de données et SQL',02),
(1440007, 'R1-06', 'Mathématique discrète',02),
(1440009, 'R1-07', 'Outils mathématique fondamentaux',02),
(1440003, 'R1-08', 'Gestion de projet',02),
(1440004, 'R1-10', 'Anglais technique',02),
(1440006, 'R1-12', 'Projet professionel et personnel',02),
(1440005, 'R1-13', 'Service Réseaux',02);

INSERT INTO Controle(IdP, IdE, IdC, IdM, Nom, Note) VALUES
(1440001, 1220001, '01', 'R1-01', 'Mini-Controle', 20),
(1440001, 1220002, '01', 'R1-01', 'Mini-Controle', 07),
(1440001, 1220003, '01', 'R1-01', 'Mini-Controle', 14),
(1440001, 1220004, '01', 'R1-01', 'Mini-Controle', 12),
(1440001, 1220005, '01', 'R1-01', 'Mini-Controle', 18),
(1440001, 1220001, '02', 'R1-01', 'Examen final', 20),
(1440001, 1220002, '02', 'R1-01', 'Examen final', 07),
(1440001, 1220003, '02', 'R1-01', 'Examen final', 11),
(1440001, 1220004, '02', 'R1-01', 'Examen final', 12),
(1440001, 1220005, '02', 'R1-01', 'Examen final', 18),
(1440002, 1220001, '03', 'R1-02', 'Controle', 19),
(1440002, 1220002, '03', 'R1-02', 'Controle', 15),
(1440002, 1220003, '03', 'R1-02', 'Controle', 14),
(1440002, 1220004, '03', 'R1-02', 'Controle', 12),
(1440002, 1220005, '03', 'R1-02', 'Controle', 18),
(1440002, 1220001, '04', 'R1-02', 'Examen final', 20),
(1440002, 1220002, '04', 'R1-02', 'Examen final', 07),
(1440002, 1220003, '04', 'R1-02', 'Examen final', 14),
(1440002, 1220004, '04', 'R1-02', 'Examen final', 09),
(1440002, 1220005, '04', 'R1-02', 'Examen final', 18),
(1440003, 1220001, '03', 'R1-08', 'Controle', 18),
(1440003, 1220002, '03', 'R1-08', 'Controle', 17),
```

```
(1440003, 1220003, '03', 'R1-08', 'Controle', 14),
(1440003, 1220004, '03', 'R1-08', 'Controle', 12),
(1440003, 1220005, '03', 'R1-08', 'Controle', 08),
(1440004, 1220001, '04', 'R1-10', 'Examen final', 16),
(1440004, 1220002, '04', 'R1-10', 'Examen final', 07),
(1440004, 1220003, '04', 'R1-10', 'Examen final', 15),
(1440004, 1220004, '04', 'R1-10', 'Examen final', 13),
(1440004, 1220005, '04', 'R1-10', 'Examen final', 11),
(1440005, 1220001, '05', 'R1-13', 'Examen final', 20),
(1440005, 1220002, '05', 'R1-13', 'Examen final', 07),
(1440005, 1220003, '05', 'R1-13', 'Examen final', 14),
(1440005, 1220004, '05', 'R1-13', 'Examen final', 12),
(1440005, 1220005, '05', 'R1-13', 'Examen final', 18),
(1440006, 1220001, '06', 'R1-12', 'Examen final', 20),
(1440006, 1220002, '06', 'R1-12', 'Examen final', 06),
(1440006, 1220003, '06', 'R1-12', 'Examen final', 14),
(1440006, 1220004, '06', 'R1-12', 'Examen final', 12),
(1440006, 1220005, '06', 'R1-12', 'Examen final', 00),
(1440007, 1220001, '07', 'R1-06', 'Examen final', 19),
(1440007, 1220002, '07', 'R1-06', 'Examen final', 07),
(1440007, 1220003, '07', 'R1-06', 'Examen final', 14),
(1440007, 1220004, '07', 'R1-06', 'Examen final', 05),
(1440007, 1220005, '07', 'R1-06', 'Examen final', 18),
(1440008, 1220001, '08', 'R1-05', 'Examen final', 20),
(1440008, 1220002, '08', 'R1-05', 'Examen final', 07),
(1440008, 1220003, '08', 'R1-05', 'Examen final', 01),
(1440008, 1220004, '08', 'R1-05', 'Examen final', 12),
(1440008, 1220005, '08', 'R1-05', 'Examen final', 18),
(1440009, 1220001, '09', 'R1-07', 'Examen final', 19),
(1440009, 1220002, '09', 'R1-07', 'Examen final', 04),
(1440009, 1220003, '09', 'R1-07', 'Examen final', 15),
(1440009, 1220004, '09', 'R1-07', 'Examen final', 12),
(1440009, 1220005, '09', 'R1-07', 'Examen final', 16);
```

II. Visualisation des Données :

1- Procédures :

a. La Fonction **Notes_etudiant** qui affiche toutes les notes d'un seul étudiant dont son identifiant passé en paramètre :

```
CREATE OR REPLACE FUNCTION Notes_etudiant(in idE int , out Nom varchar ,
 out Prenom varchar , out module varchar , out controle varchar ,out note
float)
 returns setof record as $$
 select Etudiant.Nom , Etudiant.Prenom , Controle.idM , Controle.Nom, note
 from Etudiant join Controle ON Etudiant.idE = Controle.idE
 Where Etudiant.idE=$1 ;
 $$language SQL ;
 /*exemple*/
```

```
 sae204=> select * from Notes_etudiant(1220001) ;
   nom    | prenom | module |   controle    | note
----------+--------+--------+---------------+------
 Mazraoui | Karim  | R1-01  | Mini-Controle |   20
 Mazraoui | Karim  | R1-01  | Examen final  |   20
 Mazraoui | Karim  | R1-02  | Controle      |   19
 Mazraoui | Karim  | R1-02  | Examen final  |   20
 Mazraoui | Karim  | R1-08  | Controle      |   18
 Mazraoui | Karim  | R1-10  | Examen final  |   16
 Mazraoui | Karim  | R1-13  | Examen final  |   20
 Mazraoui | Karim  | R1-12  | Examen final  |   20
 Mazraoui | Karim  | R1-06  | Examen final  |   19
 Mazraoui | Karim  | R1-05  | Examen final  |   20
 Mazraoui | Karim  | R1-07  | Examen final  |   19
(11 rows)
```

b. La fonction **Moyenne_module_etudiant** affiche la moyenne de chaque module d'un seul étudiant dont son identifiant passé en paramètre :

```
CREATE OR REPLACE FUNCTION Moyenne_module_etudiant( in idE int ,
out Nom varchar , out Prenom varchar ,out Module varchar , out Moyenne
float)
returns setof record as $$
select Etudiant.Nom , Etudiant.Prenom , Module.Nom , avg(note)
from Etudiant join Controle On Etudiant.idE=Controle.idE
join Module On Module.idM=Controle.idM
Where Etudiant.idE=$1
group by Module.Nom , Etudiant.Nom , Etudiant.Prenom;
$$language SQL ;
/*exemple*/
sae204=> select * from Moyenne_module_etudiant(1220001) ;
nom    | prenom |                  module                   | moyenne
----------+--------+-------------------------------------------+---------
Mazraoui | Karim  | Anglais technique                         |      16
Mazraoui | Karim  | Developpement interface web               |    19.5
 Mazraoui | Karim  | Gestion de projet                         |      18
 Mazraoui | Karim  | Initiation au développement               |      20
 Mazraoui | Karim  | Introduction aux bases de données et SQL  |      20
 Mazraoui | Karim  | Mathématique discrète                     |      19
 Mazraoui | Karim  | Outils mathématique fondamentaux          |      19
 Mazraoui | Karim  | Projet professionel et personnel          |      20
 Mazraoui | Karim  | Service Réseaux                           |      20
(9 rows)

sae204=> select * from Moyenne_module_etudiant(1220005) ;
   nom    | prenom |                  module                   | moyenne
----------+--------+-------------------------------------------+---------
 Makassou | Fatima | Anglais technique                         |      11
 Makassou | Fatima | Developpement interface web               |      18
```

```
 Makassou | Fatima | Gestion de projet                        |       8
 Makassou | Fatima | Initiation au développement              |      18
 Makassou | Fatima | Introduction aux bases de données et SQL |      18
 Makassou | Fatima | Mathématique discrète                    |      18
 Makassou | Fatima | Outils mathématique fondamentaux         |      16
 Makassou | Fatima | Projet professionel et personnel         |       0
 Makassou | Fatima | Service Réseaux                          |      18
(9 rows)
```

c. La fonction **Moyenne_etudiant** renvoie la moyenne général de l'étudiant :

```
Create or replace function Moyenne_etudiant(in idE int , out Nom varchar ,
out Prenom varchar , out moyenne float) returns setof record as $$
select Etudiant.Nom , Etudiant.Prenom , avg(note)
from Etudiant join Controle ON Etudiant.idE=Controle.idE
Where Etudiant.idE=$1
group by Etudiant.idE ;
$$language SQL ;
/*exemple*/
sae204=> select * from Moyenne_etudiant(1220001) ;
   nom    | prenom |      moyenne
----------+--------+--------------------
 Mazraoui | Karim  | 19.181818181818183
(1 row)

sae204=> select * from Moyenne_etudiant(1220005) ;
   nom    | prenom |      moyenne
----------+--------+--------------------
 Makassou | Fatima | 14.636363636363637
(1 row)
```

d. La fonction **Moyenne_module** renvoie la moyenne d'un seul module passé en paramètre :

```
CREATE OR REPLACE FUNCTION Moyenne_module(in idM varchar , out Module
varchar ,
out Moyenne float)returns setof record as $$
select Module.Nom , avg(note)
from Module join Controle On Module.idM=Controle.idM
Where Module.idM =$1
Group by Module.idM ;
$$language SQL  ;
/*exemple*/
sae204=> select * from Moyenne_module('R1-01') ;
          module           | moyenne
---------------------------+---------
 Initiation au développement |    13.9
(1 row)
```

```
sae204=> select * from Moyenne_module('R1-02') ;
          module            | moyenne
----------------------------+---------
 Developpement interface web |    14.6
(1 row)
```

e. La fonction **Notes_groupe** renvoie les notes de tous les étudiants d'un seul groupe passé en paramètre :

```
 CREATE OR REPLACE FUNCTION Notes_groupe(in groupe varchar , out nom
varchar,
 out prenom varchar , out groupe varchar, out module varchar ,
 out controle varchar, out note float )returns setof record as $$
 select Etudiant.Nom , Etudiant.prenom , Etudiant.groupe , Controle.idM ,
 Controle.Nom , Controle.note
 from Etudiant join Controle ON Etudiant.idE = Controle.IdE
 WHERE Etudiant.Groupe =$1 ;
 $$ language SQL ;
 /*exemple*/
sae204=> select * from Notes_groupe('zeus') ;
   nom    | prenom | groupe | module |   controle    | note
----------+--------+--------+--------+---------------+------
 Mazraoui | Karim  | zeus   | R1-01  | Mini-Controle |   20
 Dupont   | Marine | zeus   | R1-01  | Mini-Controle |    7
 Mazraoui | Karim  | zeus   | R1-01  | Examen final  |   20
 Dupont   | Marine | zeus   | R1-01  | Examen final  |    7
 Mazraoui | Karim  | zeus   | R1-02  | Controle      |   19
 Dupont   | Marine | zeus   | R1-02  | Controle      |   15
 Mazraoui | Karim  | zeus   | R1-02  | Examen final  |   20
 Dupont   | Marine | zeus   | R1-02  | Examen final  |    7
 Mazraoui | Karim  | zeus   | R1-08  | Controle      |   18
 Dupont   | Marine | zeus   | R1-08  | Controle      |   17
 Mazraoui | Karim  | zeus   | R1-10  | Examen final  |   16
 Dupont   | Marine | zeus   | R1-10  | Examen final  |    7
 Mazraoui | Karim  | zeus   | R1-13  | Examen final  |   20
 Dupont   | Marine | zeus   | R1-13  | Examen final  |    7
 Mazraoui | Karim  | zeus   | R1-12  | Examen final  |   20
 Dupont   | Marine | zeus   | R1-12  | Examen final  |    6
 Mazraoui | Karim  | zeus   | R1-06  | Examen final  |   19
 Dupont   | Marine | zeus   | R1-06  | Examen final  |    7
 Mazraoui | Karim  | zeus   | R1-05  | Examen final  |   20
 Dupont   | Marine | zeus   | R1-05  | Examen final  |    7
 Mazraoui | Karim  | zeus   | R1-07  | Examen final  |   19
 Dupont   | Marine | zeus   | R1-07  | Examen final  |    4
(22 rows)
```

f. La fonction **Moyenne_groupe** renvoie la moyenne d'un groupe passé en paramètre :

```
CREATE OR REPLACE FUNCTION Moyenne_groupe( inout Groupe varchar , out
moyenne float)
returns setof record as $$
select Etudiant.Groupe , avg(note)
from Etudiant JOIN Controle ON Etudiant.idE=Controle.idE
Where Etudiant.groupe=$1
Group By Etudiant.groupe ;
$$ language sql ;
/*exemple*/
sae204=> select * from Moyenne_groupe('zeus') ;
 groupe |      moyenne
--------+--------------------
 zeus   | 13.727272727272727
(1 row)

sae204=> select * from Moyenne_groupe('whaitiri') ;
  groupe  |      moyenne
----------+--------------------
 whaitiri | 14.636363636363637
(1 row)
```

g. La fonction **Note_modulaire** renvoie la moyenne d'un module passé en paramètre pour un seul étudiant :

```
CREATE OR REPLACE FUNCTION Note_modulaire(in idE integer ,inout Module
varchar , out Note float)
returns setof record as $$
select Controle.idM , avg(note) from Controle
Where Controle.idE=$1 and Controle.idM=$2
group by Controle.idM ;
$$language SQL security definer  ;
/*exemple *//*calcule la moyenne d'un module pour un seul étudiant*/
sae204=> select * from Note_modulaire(1220001,'R1-01') ;
 module | note
--------+------
 R1-01  |   20
(1 row)
sae204=> select * from Note_modulaire(1220003,'R1-01') ;
 module | note
--------+------
 R1-01  | 12.5
(1 row)
```

2- Les Vues :

a. La vue **Etudiants_Zeus** affiche tous les étudiants de zeus :

```
Create view Etudiants_Zeus
As
Select Nom  , Prenom From Etudiant
Where Groupe='zeus' ;

/* exemple*/
sae204=# select * from Etudiants_Zeus ;
   nom    | prenom
----------+--------
 Mazraoui | Karim
 Dupont   | Marine
(2 rows)
```

b. La vue **Etudiants_Shango** affiche tous les étudiants du groupe shango :

```
Create view Etudiants_Shango
As
Select Nom  , Prenom From Etudiant
Where Groupe='shango' ;
/* xemepple*/
sae204=# select * from Etudiants_Shango ;
 nom  | prenom
------+--------
 Saka | Pierre
(1 row)
```

c. La vue **Etudiants_Tlaloc** affiche tous les étudiants du groupe tlaloc :

```
Create view Etudiants_Tlaloc
As
Select Nom  , Prenom From Etudiant
Where Groupe='tlaloc' ;
/*exemple*/
sae204=# select * from Etudiants_Tlaloc ;
   nom   | prenom
---------+--------
 Barbara | Selma
(1 row)
```

d. La vue **Etudiants_Whaitiri** affiche tous les étudiants du groupe whaitiri :

```
Create view Etudiants_Whaitiri
As
Select Nom  , Prenom From Etudiant
Where Groupe='whaitiri' ;
/*exmpel*/
sae204=# select * from Etudiants_Whaitiri ;
   nom    | prenom
----------+--------
 Makassou | Fatima
```

```
(1 row)
```

e. La vue **Moyennes_tous_Etudiants** affiche la moyenne générale de tous les étudiants de la promo :

```
Create view Moyennes_tous_Etudiants
As
Select Etudiant.nom , Etudiant.prenom , avg(note) As Moyenne
From Etudiant join Controle On Etudiant.idE = Controle.idE
Group BY Etudiant.idE
Order by Moyenne desc  ;
/*afficher la vue*/
sae204=# select * from Moyennes_tous_Etudiants ;
   nom    | prenom |      moyenne
----------+--------+--------------------
 Mazraoui | Karim  |  19.09090909090909
 Makassou | Fatima | 14.636363636363637
 Saka     | Pierre | 12.727272727272727
 Barbara  | Selma  | 11.363636363636363
 Dupont   | Marine |  8.272727272727273
(5 rows)
```

f. La vue **Etudiants_Avec_Moyenne_Sup_10** affiche les étudiants qui ont eu une moyenne supérieur à 10 :

```
CREATE VIEW Etudiants_Avec_Moyenne_Sup_10 AS
SELECT Etudiant.Nom, Etudiant.Prenom, AVG(Controle.note) AS Moyenne
FROM Etudiant
JOIN Controle ON Etudiant.idE = Controle.idE
GROUP BY Etudiant.idE
HAVING AVG(Controle.note) > 10;
/*exemple*/
sae204=# Select * from Etudiants_Avec_Moyenne_Sup_10 ;
   nom    | prenom |      moyenne
----------+--------+--------------------
 Mazraoui | Karim  |  19.09090909090909
 Makassou | Fatima | 14.636363636363637
 Saka     | Pierre | 12.727272727272727
 Barbara  | Selma  | 11.363636363636363
(4 rows)
```

g. La vue **Etudiants_Redoubles** affiche tous les étudiants qui ont eu une moyenne générale inférieur à 10 :

```
CREATE VIEW Etudiants_Redoubles AS
SELECT Etudiant.Nom, Etudiant.Prenom, AVG(Controle.note) AS Moyenne
FROM Etudiant
JOIN Controle ON Etudiant.idE = Controle.idE
GROUP BY Etudiant.idE
HAVING AVG(Controle.note) < 10;
```

```
/* exemple*/
sae204=# select * from Etudiants_Redoubles ;
  nom   | prenom |      moyenne
--------+--------+-------------------
 Dupont | Marine | 8.272727272727273
(1 row)
```

h. La vue **Moyenne_modules** affiche la moyenne générale de toute la promo pour chaque module :

```
CREATE View Moyenne_modules
AS
select Module.Nom , avg(note)
from Module join Controle On Module.idM=Controle.idM
group by Module.idM ;

/*exemple*/
sae204=# select * from Moyenne_modules ;
                   nom                    | avg
------------------------------------------+------
 Mathématique discrète                    | 12.6
 Initiation au développement              | 13.8
 Projet professionel et personnel         | 10.4
 Outils mathématique fondamentaux         | 13.2
 Service Réseaux                          | 14.2
 Developpement interface web              | 14.6
 Gestion de projet                        | 13.8
 Anglais technique                        | 12.8
 Introduction aux bases de données et SQL | 11.6
(9 rows)
```

i. La vue **Moyennes_groupes** affiche la moyenne générale de chaque groupe :

```
CREATE VIEW Moyennes_groupes
as
select Etudiant.groupe , avg(note)
from Etudiant join Controle On Etudiant.IdE=Controle.idE
Group by Etudiant.groupe ;

/*exemple*/
sae204=# SELECT * from Moyennes_groupes ;
  groupe  |        avg
----------+--------------------
 shango   | 12.727272727272727
 whaitiri | 14.636363636363637
 tlaloc   | 11.363636363636363
 zeus     | 13.681818181818182
(4 rows)
```

III. Restriction d'accès aux Données :

1- Les droits d'accès aux tables :

Il y a trois groupes d'utilisateurs : L'étudiant , le professeur , l'administaration.

- Le groupe etudiant qui contient tous les étudiants a aucun droit sur les tables (Professeur, Etudiant, Departement, Controle) ; a le droit de voir la table Module. +a le droit d'exécuter deux fonctions Mes_notes , Ma_note qu'on va détailler plus tard.
- Le groupe professeur a le droit de lectures des tables (Professeur, Etudiant , Departement, Module) ; a le droit de lecture ou d'insérer sur les tables (Contrôle). + a le droit d'exécuter la fonction Changer_note qu'on va détailler éventuellement plus tard.
- L'administaration (superadmin) a tous les droits sur toutes les tables.

2- Le script de la restriction des données :

a. Le rôle etudiant : Elle consiste à créer pour chaque étudiant un utilisateur en se référant à son identifiant, et les attribuer tous au rôle etudiant.

```
create group etudiant ;
create user "1220001" with password '1234' ;
create user "1220002" with password '1234' ;
create user "1220003" with password '1234' ;
create user "1220004" with password '1234' ;
create user "1220005" with password '1234' ;
grant etudiant to "1220001" ,"1220002" ,"1220003","1220004","1220005";
Grant SELECT ON module TO etudiant ;
GRANT EXECUTE ON FUNCTION Mes_notes(OUT Module varchar, OUT Controle
varchar, OUT note float) TO etudiant;
GRANT EXECUTE ON FUNCTION Ma_note(in idM varchar, in idC integer)TO
etudiant;
```

b. **Le rôle professeur :** Elle consiste à créer pour chaque professeur un utilisateur en se référant également à son identifiant , et les attribuer au rôle professeur.

```
create group professeur ;
create user "1440001" with password '1234' ;
create user "1440002" with password '1234' ;
create user "1440003" with password '1234' ;
create user "1440004" with password '1234' ;
create user "1440005" with password '1234' ;
create user "1440006" with password '1234' ;
create user "1440007" with password '1234' ;
create user "1440008" with password '1234' ;
create user "1440009" with password '1234' ;
create user "1440010" with password '1234' ;
grant professeur to "1440001","1440002","1440003","1440004","1440005",
"1440006","1440007","1440008","1440009","1440010";
Grant select on professeur, etudiant , departement , module to professeur ;
Grant select, insert on controle to professeur ;
GRANT EXECUTE ON FUNCTION Changer_la_note(in idE integer , in idC integer
```

```
,in idM varchar , in nouvelle_note float) TO professeur;
```

c. **Le rôle administration :** Ce rôle à tous les droits sur la base de données , néanmoins il a pas droit d'exécuter les trois fonctions attribuées aux autres utilisateurs.

```
create group administration ;
create user superadmin with password '1234' ;
grant administration to superadmin ;
GRANT ALL PRIVILEGES ON DATABASE sae204 TO administration ;
```

3- Illustration des rôles créés :

```
     groupe     |                                  utilisateurs
----------------+-------------------------------------------------------------------------------------
 administration | {superadmin}
 etudiant       | {1220004,1220003,1220002,1220001,1220005}
 professeur     | {1440009,1440008,1440007,1440006,1440005,1440004,1440003,1440002,1440001,1440010}
(3 rows)

~
```

4- Procédures pour mettre en œuvre ces restrictions :

a- La fonction **Changer_la_note** permet juste au prof responsable du c^^ontre à modifier la note d'un étudiants :

```
CREATE OR REPLACE FUNCTION Changer_la_note(in idE integer , in idC integer
,in idM varchar , in nouvelle_note float) returns void as $$
BEGIN
   IF session_user::integer = (select idP from Controle Where Controle.idE=$1
AND Controle.idC=$2 AND Controle.idM =$3) Then
      Update Controle
      Set Note=$4
      Where Controle.idE=$1 and Controle.idC=$2 and Controle.idM=$3 ;
   ELSE
      RAISE EXCEPTION 'Vous devez être le prof responsable du contrôle pour
pouvoir modifier le controle';
   END IF;
END ;
$$ language plpgsql Security definer ;
```

Exemple d'application :

En se connectant à la base de données sae204 avec l'utilisateurs 1440001 qui est un professeur responsables du module R1-01 et des contrôles 01 et 02 , La note de l'étudiant à la base était 18 , en exécutant la fonction **Changer_la_note** on voit bien que la note a été changé en 19. Mais avec le même utilisateur en essayant de changer la note d'un étudiant sur un contrôle dont il est pas responsable(le contrôle 09), ça fonctionne pas et ça affiche le message « Vous devez être le prof responsable du contrôle pour pouvoir modifier la note ».

Je tiens à préciser que le rôle professeur à le droit d'exécuter quelques fonctions pour savoir si la note a bien été changée.

```
kenzahatem@hatemkenza:~$ psql -U 1440001 -h localhost -d sae204
Password for user 1440001:
psql (13.10 (Debian 13.10-0+deb11u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256
, compression: off)
Type "help" for help.

sae204=> select * from Note_etudiant(1220001 ,01 ,'R1-01') ;
 module |      nom      | note
--------+---------------+------
 R1-01  | Mini-Controle |   18
(1 row)

sae204=> select * from Changer_la_note(1220001,01
,'R1-01',19) ;
 changer_la_note
-----------------

(1 row)

sae204=> select * from Note_etudiant(1220001 ,01 ,'R1-01') ;
 module |      nom      | note
--------+---------------+------
 R1-01  | Mini-Controle |   19
(1 row)

sae204=> select * from Changer_la_note(1220001,09
,'R1-07',19) ;
ERROR:  Vous devez être le prof responsable du contrôle pour pouvoir modifie
r le controle
CONTEXT:  PL/pgSQL function changer_la_note(integer,integer,character varyin
g,double precision) line 8 at RAISE
```

b. La fonction **Mes_notes** est une fonction qui affiche toutes les notes de l'étudiant, cette fonction est accessible qu'aux étudiants :

```
Create or replace function Mes_notes(out Module varchar , out Controle varchar
,out note Float)
returns setof record as
$$
select Module.nom , Controle.Nom , Note From Module Join Controle
On Module.idM= Controle.idM
Where Controle.idE = session_user::integer ;
$$language SQL Security definer ;
```

Exemple d'application :

En se connectant à la base de données en tant que l'utilisateur 1220001 (etudiant) on pourra exécuter la fonction **Mes_notes()** pour afficher toutes les notes obtenues par l'étudiant.

```
kenzahatem@hatemkenza:~$ psql -U 1220001 -h localhost -d sae204
Password for user 1220001:
psql (13.10 (Debian 13.10-0+deb11u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256
, compression: off)
Type "help" for help.

sae204=> select * from Mes_notes() ;
                 module                  |   controle    | note
-----------------------------------------+---------------+------
 Initiation au développement             | Mini-Controle |   20
 Initiation au développement             | Examen final  |   20
 Developpement interface web             | Controle      |   19
 Developpement interface web             | Examen final  |   20
 Gestion de projet                       | Controle      |   18
 Anglais technique                       | Examen final  |   16
 Service Réseaux                         | Examen final  |   20
 Projet professionel et personnel        | Examen final  |   20
 Mathématique discrète                   | Examen final  |   19
 Introduction aux bases de données et SQL | Examen final  |   20
 Outils mathématique fondamentaux        | Examen final  |   19
(11 rows)
```

c. La fonction **Ma_note** affiche la note d'un contrôle dont l'identifiant et le module passé en paramètre, cette fonction est accessible qu'au rôle etudiant :

```
CREATE FUNCTION Ma_note(in idM varchar, in idC integer)
RETURNS TABLE (Module varchar, Controle varchar, Note float) AS
$$
SELECT Module.Nom, Controle.Nom, Note
FROM Controle
JOIN Module ON Module.idM = Controle.idM
WHERE Controle.idE = session_user::integer AND Controle.idC=$2 AND
Module.idM=$1;
$$ LANGUAGE SQL security definer;
```

Exemple d'application :

En se connectant en tant qu'utilisateur 1220002 ou pourra voir la note qu'on a obtenu dans le module R1-01 le premier contrôle par exemple.

```
kenzahatem@hatemkenza:~$ psql -U 1220002 -h localhost -d sae204
Password for user 1220002:
psql (13.10 (Debian 13.10-0+deb11u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256
, compression: off)
Type "help" for help.

sae204=> select * from Ma_note('R1-01',01) ;
           module            |   controle    | note
-----------------------------+---------------+------
 Initiation au développement | Mini-Controle |    7
(1 row)

sae204=> select * from Ma_note('R1-01',02) ;
           module            |   controle   | note
-----------------------------+--------------+------
 Initiation au développement | Examen final |    7
(1 row)
```

Merci !!