SAE 2.04

EXPLOITATION D'UNE BASE DE DONNÉES

Base de données des étudiants en BUT

Arumugadas Ravichandran
Li Loic

SAE 2.04 : Exploitation d'une base de données

Objectifs du projet :

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

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

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

LI LOIC
ARUMUGADAS RAVICHANDRAN

SAE 2.04 : Exploitation d'une base de données

I - Modélisation de données

1) Cahier des charges

Contexte et Objectif :

L'objectif de cette SAE est de réaliser et mettre en place une base de données permettant la gestion des notes des étudiants en BUT pour l'IUT de Villetaneuse.

Exigences fonctionnelles :

- ★ La base de données doit permettre l'enregistrement des notes que les étudiants ont eu dans chaque matière.

- ★ Fournir des procédures, ou vues afin d'accéder à ces données -> ce qui permet la communication entre la base de données et l'utilisateur facilement (exemple : relevé les notes d'un groupe)

- ★ Effectuer un modèle de données avec un logiciel d'AGL afin de visualiser et d'avoir une idée de la base de données. Créer la base de données avec un logiciel et mettre en innovation toutes les fonctionnalités utiles pour la base de données. (Postgresql)

- ★ Écrire le script de la base de données.

LI LOIC
ARUMUGADAS RAVICHANDRAN

Exigences techniques :

Le modèle de données doit contenir les tables en relation à la base de données :

- ★ Table enseignante : Cette table doit contenir les données associées aux enseignants en comprenant son id, le nom et le prénom de l'enseignant.

- ★ Table étudiante : Cette table doit stocker les informations relatives aux étudiants en comprenant son id, le nom et le prénom de l'étudiant, le NIP et le groupe. D'ailleurs, l'attribut 'NIP' qui représente le numéro d'identifiant de l'étudiant, doit commencer par '122' et doit contenir 8 chiffres en total.

- ★ Table matière : Cette table doit contenir les informations relatives aux matières telle que son identifiant, le nom de la matière et l'enseignant qui s'occupe de la matière (responsable de la matière).

- ★ Table contrôle : Cette table doit contenir les informations associées aux contrôles telle que son identifiant, l'identifiant de la matière en question (clé secondaire), l'intitulé du contrôle et la date du contrôle.

- ★ Table note : Cette table doit contenir les informations relatives aux notes des étudiants telle que l'identifiant de l'étudiant, l'identifiant du contrôle et l'identifiant de la matière ainsi que la note obtenue.

Contraintes/Restrictions d'accès :

La base de données doit mettre en place des restrictions d'accès aux données selon le rôle de l'utilisateur. Les étudiants ne pourront accéder qu'à leurs propres notes tandis que les enseignants et les responsables de matière pourront accéder aux notes des étudiants inscrits dans leurs matières (afin de modifier une note ou la supprimer par exemple). Ainsi, la base de données doit permettre la gestion des données tout en permettant des restrictions d'accès selon le rôle de l'utilisateur. Des mesures de sécurité doivent être mises en place pour assurer la sécurité des données.

Il s'agit donc de :

- ★ Créer des rôles afin de définir des règles d'accès des données.

- ★ Effectuer un ensemble de procédures ou vues pour mettre en place un ensemble de règles d'accès à la base de données.

- ★ Mettre en place des contraintes, des restrictions et des règles d'accès aux données afin de sécuriser l'accès à la base de données.

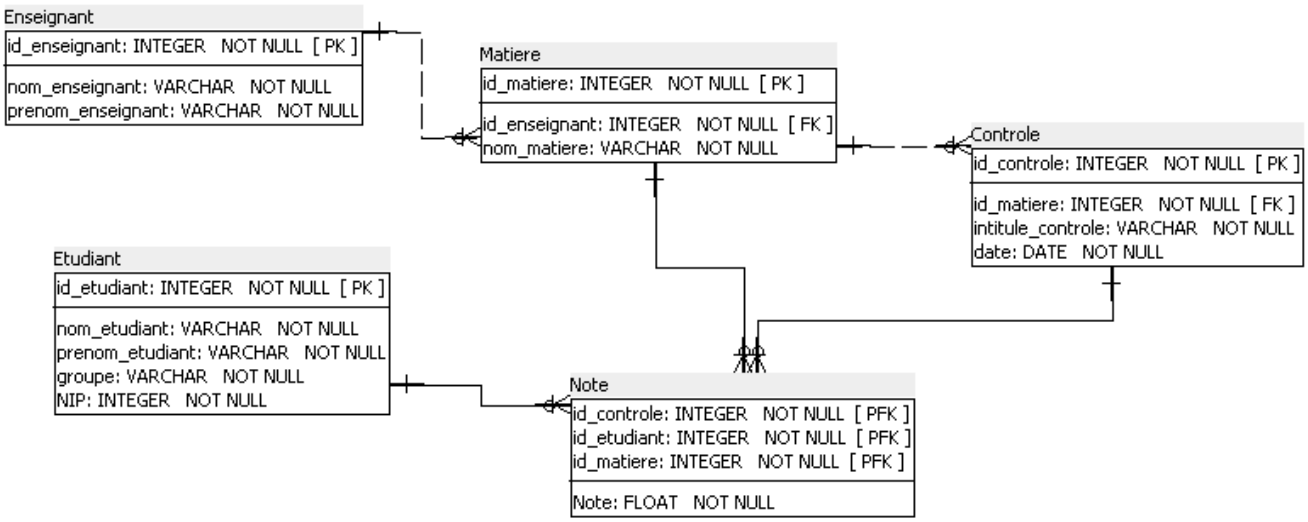
Exemple : un étudiant ne peut consulter que ses propres notes, etc…

Calendrier :

Le projet commence le 8 mars et se termine le 22 mai 2023.

2) Modélisation de la base de données :

Voici une modélisation de la base de données :



- Enseignant :

Dans cette table, nous allons retrouver 3 colonnes :

- id_enseignant qui est de type integer (la clé primaire)
- nom_enseignant qui est de type varchar et qui désigne le nom de l'enseignant
- prenom_enseignant qui est de type varchar et qui désigne le prénom de l'enseignant

- Etudiant :

Cette table contient :

  - id_etudiant qui est la clé primaire de la table (Integer)
  - nom_etudiant : le nom de l'étudiant (Varchar)
  - prenom_etudiant : prénom de l'étudiant (Varchar)
  - nip est un identifiant unique posséder par chaque étudiant, le nip doit se situer entre 12200000 et 12299999 (Integer)
  - groupe : le groupe auquel appartient l'étudiant (Varchar)

- Matiere :

Cette table contient :

  - id_matiere, la clé primaire de la table (Integer)
  - id_enseignant, une clé étrangère qui permet de désigner l'enseignant qui enseigne cette matière (Integer)
  - nom_matiere, le nom de la matière (Varchar)

- Controle :

Cette table contient :

  - id_contrôle, la clé primaire de la table (Integer)
  - id_matiere, une clé étrangère de la table matière pour désigner la matière concernée par le contrôle (Integer)
  - Contrôle : nom/titre du contrôle (Varchar)
  - date_contrôle, la date du contrôle (Date)

- Note :

Cette table contient :

  - id_etudiant, une clé étrangère qui désigne l'étudiant qui a obtenu la note (Integer)
  - id_contrôle, une clé étrangère qui désigne le contrôle passé par l'étudiant (Integer)
  - id_matiere, une clé étrangère qui désigne la matière (Integer)
  - Note qui désigne la note que l'étudiant a obtenue (Float)

3) Définir des règles de gestion

La gestion de cette base de données nécessite des restrictions d'accès selon le type d'utilisateur. Par exemple, un étudiant ne peut pas modifier ou ajouter une note mais peut tout simplement visualiser ces propres notes. Du même point de vue, un enseignant doit être en mesure de saisir des notes, modifier des notes ou bien supprimer des notes dans la matière auquel il enseigne. Il s'agit donc de mettre en place des rôles selon le type d'utilisateur afin de maintenir une bonne gestion des notes.

On retrouve 3 rôles différents. Ces utilisateurs possèdent des permissions et des restrictions sur la base de données :

Étudiant : L'étudiant a seulement la permission de consulter ses propres notes.

Enseignant : L'enseignant a accès à la table note ainsi qu'aux contrôles pour pouvoir consulter, supprimer, modifier des notes.

Administrateur : L'administrateur possède tous les droits sur la base de données

4) Script de réalisation de la base de données :

Exemple de valeurs insérées pour cette base de données :

Les valeurs qui ont été insérées dans les tables sont des exemples utilisés pour représenter la structure de la base de données ; elles ne représentent pas toutes les valeurs réelles qui pourraient exister dans les tables. Dans la table "matiere", par exemple, il manque pleins de matières, et dans la table "note", seules quelques notes ont été répertoriées à titre d'exemple.

```
-- Valeurs pour la table Enseignant

Voici 15 autres exemples de noms pour les insertions dans la table "enseignant" :

INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Lefort', 'Mathieu');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Girard', 'Sophie');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Lévesque', 'Isabelle');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Côté', 'Simon');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Boucher', 'Caroline');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Tardif', 'Éric');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Pelletier', 'Marie');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Lapointe', 'David');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Gagné', 'Valérie');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Morin', 'Lucie');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Simard', 'Philippe');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Desjardins', 'Isabelle');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Bélanger', 'Martin');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Lauzon', 'Stéphanie');
INSERT INTO enseignant (nom_enseignant, prenom_enseignant) VALUES ('Fournier', 'Jean-François');


-- Valeurs pour la table Etudiant

INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Tremblay', 'Alex', 12212345, 'Tlaloc');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Gagnon', 'Sophie', 12223456, 'Whaithiri');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Lavoie', 'Simon', 12234567, 'Zeus');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Roy', 'Marie', 12245678, 'Shango');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Bergeron', 'Mathieu', 12256789, 'Indra');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Dupuis', 'Sophia', 12267890, 'Tlaloc');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Lemieux', 'Samuel', 12278901, 'Whaithiri');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Gauthier', 'Julie', 12289012, 'Zeus');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Poirier', 'Philippe', 12290123, 'Shango');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Fortin', 'Emma', 12201234, 'Indra');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Bouchard', 'Gabriel', 12212345, 'Tlaloc');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Morin', 'Clara', 12223456, 'Whaithiri');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Lefebvre', 'Thomas', 12234567, 'Zeus');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Martel', 'Olivia', 12245678, 'Shango');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Girard', 'Noah', 12256789, 'Indra');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Pelletier', 'Léa', 12267890, 'Tlaloc');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Rousseau', 'Félix', 12278901, 'Whaithiri');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Lapointe', 'Eva', 12289012, 'Zeus');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Leclerc', 'Nathan', 12290123, 'Shango');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Deschênes', 'Alice', 12201234, 'Indra');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Caron', 'Jacob', 12212345, 'Tlaloc');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Lemay', 'Sarah', 12223456, 'Whaithiri');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Bertrand', 'Édouard', 12234567, 'Zeus');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Bélanger', 'Liam', 12245678, 'Shango');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Hamel', 'Camille', 12256789, 'Indra');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Leduc', 'Zoé', 12267890, 'Tlaloc');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Fortier', 'Gabrielle', 12278901, 'Whaithiri');
INSERT INTO etudiant (nom_etudiant, prenom_etudiant, nip, groupe) VALUES ('Ouellet', 'Maxime', 12289012, 'Zeus');
```

LI LOIC
ARUMUGADAS RAVICHANDRAN

```sql
-- Valeurs pour la table Matière


INSERT INTO matiere (id_enseignant, nom_matiere) VALUES (1, 'Développement orienté objets');
INSERT INTO matiere (id_enseignant, nom_matiere) VALUES (2, 'Graphes');
INSERT INTO matiere (id_enseignant, nom_matiere) VALUES (3, 'Anglais');
INSERT INTO matiere (id_enseignant, nom_matiere) VALUES (4, 'Développement d''applications avec IHM');
INSERT INTO matiere (id_enseignant, nom_matiere) VALUES (5, 'Communication et fonctionnement bas niveau');
INSERT INTO matiere (id_enseignant, nom_matiere) VALUES (6, 'Introduction aux services réseaux');


-- Valeurs pour la table Contrôle

INSERT INTO controle (id_matiere, controle, date_controle) VALUES
(1, 'Petit contrôle', '2023-05-01'),
(1, 'Contrôle connaissance', '2023-05-10'),
(1, 'Contrôle final', '2023-05-18'),
(2, 'Ataraxy', '2023-05-02'),
(2, 'Contrôle final', '2023-05-12'),
(3, 'Toiec', '2023-05-05'),
(3, 'Examen final', '2023-05-15'),
(4, 'Contrôle 1', '2023-05-08'),
(4, 'Contrôle final', '2023-05-20'),
(5, 'Examen 1', '2023-05-03'),
(5, 'Examen 2', '2023-05-13'),
(6, 'TP', '2023-05-06'),
(6, 'Contrôle final', '2023-05-16');
```

```sql
INSERT INTO note (id_etudiant, id_controle, id_matiere, note) VALUES
(1, 1, 1, 15),
(1, 2, 2, 18),
(1, 3, 1, 16),
(1, 4, 2, 13),
(1, 5, 1, 17),
(2, 1, 1, 14),
(2, 2, 2, 16),
(2, 3, 1, 15),
(2, 4, 2, 12),
(2, 5, 1, 14),
(3, 1, 1, 16),
(3, 2, 2, 19),
(3, 3, 1, 17),
(3, 4, 2, 16),
(3, 5, 1, 18),
(4, 1, 1, 12),
(4, 2, 2, 15),
(4, 3, 1, 13),
(4, 4, 2, 14),
(4, 5, 1, 16),
(5, 1, 1, 17),
(5, 2, 2, 14),
(5, 3, 1, 16),
(5, 4, 2, 15),
(5, 5, 1, 13),
(6, 1, 1, 16),
(6, 2, 2, 18),
(6, 3, 1, 17),
(6, 4, 2, 16),
(6, 5, 1, 15),
(7, 1, 1, 13),
(7, 2, 2, 15),
(7, 3, 1, 14),
(7, 4, 2, 13),
(7, 5, 2, 15),
(8, 1, 1, 18),
(8, 2, 2, 16),
(8, 3, 2, 19),
(8, 4, 1, 17),
(8, 5, 1, 16),
(9, 1, 2, 15),
(9, 2, 1, 17),
(9, 3, 2, 15),
(9, 4, 1, 14),
(9, 5, 1, 16),
(10, 1,2, 17),
(10, 2, 1, 16),
(10, 3, 1, 18),
(10, 4, 2, 17),
(10, 5, 1, 19);
```

LI LOIC
ARUMUGADAS RAVICHANDRAN

II – Visualisation de données

1) Ensemble de données à visualiser :

Il s'agit maintenant d'effectuer des procédures ou des vues afin d'accéder aux données de la base de données :

- Sélectionner les notes des étudiants dans une matière précise sur un contrôle en particulier
- Classement des moyennes générales des étudiants
- Moyenne général d'un étudiant
- La moyenne d'un étudiant pour chacune des matières
- Les moyennes de chaque groupe qui a la meilleure moyenne au groupe qui a la moins bonne.
- Les moyennes de chaque matière
- Statistiques de chaque matière (moyenne des étudiants dans cette matière, note minimale et note maximale d'un contrôle)

2) Accéder à ces données :

- *Sélectionner les notes des étudiants dans une matière précise sur un contrôle en particulier*

```sql
CREATE OR REPLACE FUNCTION notes_matiere_controle(p_nom_matiere VARCHAR,
p_nom_controle VARCHAR)
RETURNS TABLE (id_etudiant INTEGER, nom_etudiant VARCHAR, prenom_etudiant VARCHAR,
note FLOAT, controle VARCHAR, date_controle DATE)
AS $$
BEGIN
 RETURN QUERY
 SELECT e.id_etudiant, e.nom_etudiant, e.prenom_etudiant, n.note, c.controle, c.date_controle
 FROM etudiant e
 JOIN note n ON e.id_etudiant = n.id_etudiant
 JOIN controle c ON n.id_controle = c.id_controle
 JOIN matiere m ON c.id_matiere = m.id_matiere
 WHERE m.nom_matiere = p_nom_matiere AND c.controle = p_nom_controle;
END;
$$ LANGUAGE plpgsql;


SELECT * FROM notes_matiere_controle(<nom_matière>, <nom_controle>);
```

Exemple :

SELECT * FROM notes_matiere_controle('Développement orienté objets', 'Petit contrôle');

	id_etudiant integer	nom_etudiant character varying	prenom_etudiant character varying	note double precision	controle character varying	date_controle date
1	1	Tremblay	Alex	15	Petit contrôle	2023-05-01
2	2	Gagnon	Sophie	14	Petit contrôle	2023-05-01
3	3	Lavoie	Simon	16	Petit contrôle	2023-05-01
4	4	Roy	Marie	12	Petit contrôle	2023-05-01
5	5	Bergeron	Mathieu	17	Petit contrôle	2023-05-01
6	6	Dupuis	Sophia	16	Petit contrôle	2023-05-01
7	7	Lemieux	Samuel	13	Petit contrôle	2023-05-01
8	8	Gauthier	Julie	18	Petit contrôle	2023-05-01
9	9	Poirier	Philippe	15	Petit contrôle	2023-05-01
10	10	Fortin	Emma	17	Petit contrôle	2023-05-01

- *Classement des moyennes générales des étudiants*

```
CREATE VIEW moyenne_generale_etudiant AS
SELECT e.id_etudiant, AVG(note) AS moyenne_generale
FROM etudiant e
JOIN note n ON e.id_etudiant = n.id_etudiant
GROUP BY e.id_etudiant
ORDER BY moyenne_generale DESC;
SELECT * FROM moyenne_generale_etudiant;
```

Exemple :

	id_etudiant integer	moyenne_generale double precision
1	10	17.4
2	8	17.2
3	3	17.2
4	6	16.4
5	1	15.8
6	9	15.4
7	5	15
8	2	14.2
9	7	14
10	4	14

(Il n'y a que la moyenne de 10 étudiants car nous n'avons pas insérées les notes de tous les étudiants).

De plus, *cette vue permet aussi d'avoir la moyenne générale d'un étudiant en particulier*, il s'agit tout simplement d'effectuer cette requête :

SELECT * FROM moyenne_generale_etudiant where id_etudiant = <id_etudiant>;

Exemple :

SELECT * FROM moyenne_generale_etudiant where id_etudiant = 1;

	id_etudiant integer	moyenne_generale double precision
1	1	15.8

- *La moyenne d'un étudiant pour chaque matière*

```
CREATE OR REPLACE FUNCTION calculer_moyenne_etudiant_par_matiere(etudiant_id INTEGER)
RETURNS TABLE (id_etudiant INTEGER, nom_etudiant VARCHAR, prenom_etudiant VARCHAR, nom_matiere VARCHAR, moyenne FLOAT) AS $$
DECLARE
    result RECORD;
BEGIN
    FOR result IN
        SELECT e.id_etudiant, e.nom_etudiant, e.prenom_etudiant, m.nom_matiere, AVG(n.note) AS moyenne
        FROM etudiant e
        JOIN note n ON e.id_etudiant = n.id_etudiant
        JOIN matiere m ON n.id_matiere = m.id_matiere
        WHERE e.id_etudiant = etudiant_id
        GROUP BY e.id_etudiant, e.nom_etudiant, e.prenom_etudiant, m.nom_matiere
    LOOP
        id_etudiant := result.id_etudiant;
        nom_etudiant := result.nom_etudiant;
        prenom_etudiant := result.prenom_etudiant;
        nom_matiere := result.nom_matiere;
        moyenne := result.moyenne;
        RETURN NEXT;
    END LOOP;
    RETURN;
END;
$$ LANGUAGE plpgsql;
```

Exemple :

SELECT * FROM calculer_moyenne_etudiant_par_matiere(10);

	id_etudiant integer	nom_etudiant character varying	prenom_etudiant character varying	nom_matiere character varying	moyenne double precision
1	10	Fortin	Emma	Développement orienté objets	17.666666666666668
2	10	Fortin	Emma	Graphes	17

- *La moyenne de chaque groupe classé du groupe qui a la meilleure moyenne au groupe qui a la moins bonne*

```sql
CREATE OR REPLACE VIEW moyenne_groupe AS
SELECT e.groupe, AVG(n.note) AS moyenne
FROM etudiant e
JOIN note n ON e.id_etudiant = n.id_etudiant
GROUP BY e.groupe
ORDER BY moyenne DESC;
```

Exemple :

*SELECT * FROM moyenne_groupe;*

	groupe character varying	moyenne double precision
1	Zeus	17.2
2	Indra	16.2
3	Tlaloc	16.1
4	Shango	14.7
5	Whaithiri	14.1

- *Statistiques de chaque matière (moyenne des étudiants dans chaque matière, note maximale et minimale)*

```sql
CREATE OR REPLACE VIEW moyennes_notes_par_matieres AS
SELECT m.id_matiere, m.nom_matiere, AVG(n.note) AS moyenne_matiere, MAX(n.note) AS
note_max_matiere, MIN(n.note) AS note_min_matiere
FROM note n
JOIN matiere m ON n.id_matiere = m.id_matiere
GROUP BY m.id_matiere, m.nom_matiere;
```

Exemple :

SELECT * FROM moyennes_notes_par_matieres;

	id_matiere integer	nom_matiere character varying	moyenne_matiere double precision	note_max_matiere double precision	note_min_matiere double precision
1	2	Graphes	15.619047619047619	19	12
2	1	Développement orienté objets	15.689655172413794	19	12

III – Restrictions d'accès des données

1) Définir les règles d'accès à ces données :

- Un étudiant ne peut consulter que ses propres notes :

Pour qu'un étudiant puisse seulement consulter ses propres notes, nous devons créer un rôle étudiant.

Création du rôle :

```
CREATE ROLE est_un_etudiant;
```

Donner les droits :

```
GRANT SELECT ON vue_notes_etudiant TO est_un_etudiant;
```

- Un étudiant ne peut pas saisir et modifier des notes :

Retirer les permissions que les étudiants ne doivent pas posséder :

```
REVOKE INSERT,UPDATE,DELETE TO est_un_etudiant;
```

- un enseignant doit pouvoir saisir les notes de ses contrôles, modifier les notes saisies, supprimer des notes et consulter les notes de toute la promotion :

Création du rôle :

CREATE ROLE est_un_enseignant;

Donner les droits :

GRANT (INSERT,UPDATE,SELECT,DELETE) ON (note,controle) TO est_un_enseignant;

GRANT EXECUTION FUNCTION ON saisir_note(int, int, int, float) TO est_un_enseignant;

- Un administrateur a accès à toutes les données de la base :

Création du rôle :

CREATE ROLE est_un_admin LOGIN PASSWORD ‘iut_admin’;

Permission :

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO est_un_admin WITH GRANT OPTION;

2) Procédures et fonctions mises en place pour appliquer ces règles :

Cette vue permet de voir la note selon le type d'utilisateur :

```sql
CREATE VIEW vue_notes_etudiant AS
SELECT * FROM note
JOIN controle ON note.id_controle = controle.id_controle
WHERE note.id_etudiant = current_user;
```

Cette fonction permet d'ajouter des notes pour un enseignant. L'administrateur possède également les permissions nécessaires pour effectuer cette opération.

```sql
CREATE OR REPLACE FUNCTION saisir_note (id_etudiant int, id_controle int, id_matiere int, note FLOAT)
RETURNS VOID AS $$
BEGIN
INSERT INTO note (id_etudiant, id_controle, id_matiere, note)   VALUES ($1, $2, $3, $4);
END;
$$ language plpgsql SECURITY DEFINER
```

Dans le zip, il y a aussi plusieurs fichiers sql :

- Table.sql : ce fichier correspond à toutes les tables de la base de données
- ValeursTables.sql : des exemples de valeurs qu'on a insérées dans les tables
- ProceduresEtVues.sql : ce fichier correspond aux procédures et aux vues qui sont nécessaires pour accéder aux informations de la base de données.
- ProceduresEtVuesPourRestriction.sql : ce fichier correspond aux rôles qu'on a crées pour définir des règles d'accès mais aussi des procédures et vues mises en place pour accèder à ces données.