S2.04 Exploitation d'une base de donnée

Partie I - Modélisation de Données

Cahier des charges :

Implémentation de la table étudiant dans la base de donnée
La table étudiant contient 6 champs :

- num_etudiant qui est la clé primaire, doit commencer par 122
- nom_etudiant
- prenom_etudiant
- date_naissance
- groupe doit correspondre à la liste des groupe en première année

Implémentation de la table enseignant dans la base de donnée
La table enseignant contient 4 champs :

- num_enseignant qui est la clé primaire
- nom_enseignant
- prenom_enseignant
- num_module est la clé étrangère de la table module

Implémentation de la table module dans la base de donnée
La table module contient 3 champs :

- num_module qui est la clé primaire, doit être compris entre 1.01 et 1.12 ou entre 2.01 et 2.14
- resp_module est la clé étrangère de la table enseignant
- nom_module

Implémentation de la table contrôle dans la base de donnée
La table compétences contient 3 champs :

- num_controle est la clé primaire
- nom_controle
- num_module est la clé étrangère de la table module

Implémentation de la table compétence dans la base de donnée
La table compétences contient 3 champs :

- UE qui est la clé primaire, doit être compris entre 11 et 16 ou entre 21 et 26
- nom_compétence

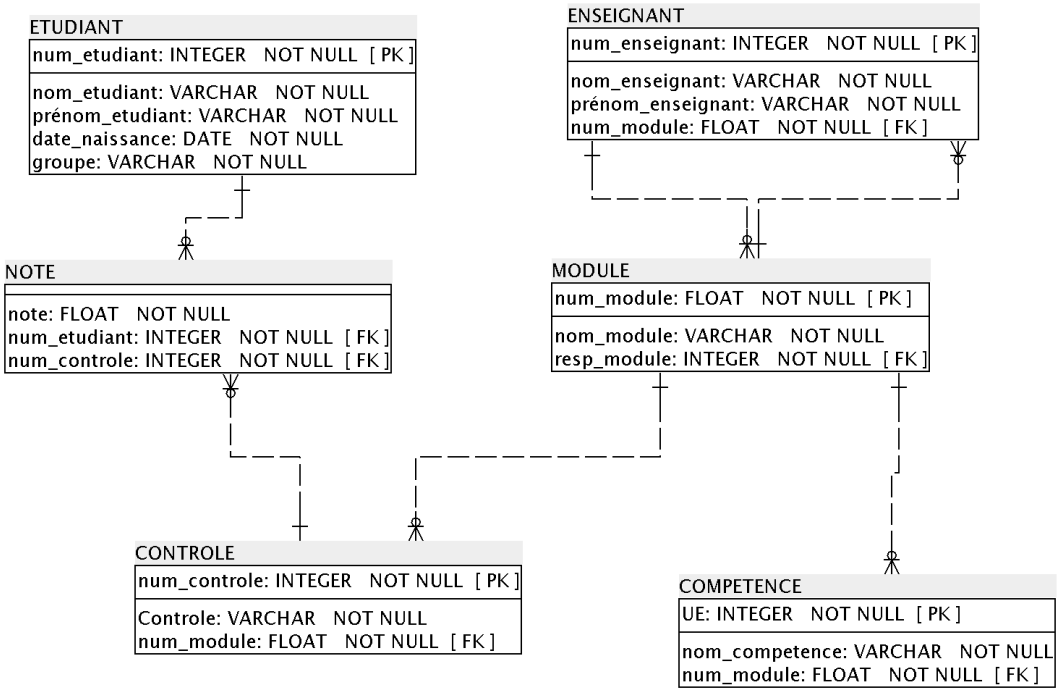
- num_module est la clé étrangère de la table module

Implémentation de la table note dans la base de donnée

La table note contient 3 champs :

- num_etudiant est la clé étrangère de la table etudiant
- num_controle est la clé étrangère de la table contrôle
- note

Modèle de donnée :



Règle de gestion des données :

Dans notre base de donnée, on aura trois rôles :

- étudiant
- enseignant
- responsable module

L'étudiant peut se connecter à notre base de données et aura les droits de consulter ses propres notes mais non celles des autres.

L'enseignant peut se connecter à notre base de données, consulter les notes de chaque étudiant, entrer ou modifier les notes de ces étudiants.

Le responsable module a les mêmes droits que l'enseignant mais il peut aussi créer des contrôles.

CREATE ROLE enseignant login;
CREATE ROLE etudiant login;
CREATE ROLE resp_module login;

Script SQL :

```
ALTER TABLE enseignant drop if exists num_module;
drop table if exists note;
drop table if exists etudiant;
drop table if exists competence;
drop table if exists controle;
drop table if exists module;
drop table if exists enseignant;


CREATE TABLE etudiant (
    num_etudiant int PRIMARY KEY ,
    nom_etudiant varchar not null,
    prénom_etudiant varchar not null,
    date_naissance date,
    groupe varchar not null,
    check (
        num_etudiant <= 12299999 and num_etudiant >= 12200000
        and
        groupe='tlaloc' or groupe='zeus' or groupe='shango' or groupe='indra' or
groupe='whaitiri')
    );

CREATE TABLE enseignant (
    num_enseignant int PRIMARY KEY ,
    nom_enseignant varchar not null,
    prénom_enseignant varchar not null
);

CREATE TABLE module (
    num_module float PRIMARY KEY,
```

```
    resp_module int REFERENCES enseignant (num_enseignant) ON
DELETE CASCADE,
    nom_module varchar not null,
    check(
        (num_module <= 1.12 and num_module >= 1.01)
        or
        (num_module <= 2.14 and num_module >= 2.01))
);

CREATE TABLE competence (
    UE int PRIMARY KEY,
    nom_competence varchar not null,
    num_module float REFERENCES module (num_module) ON DELETE
CASCADE,
    check (
        (UE <= 16 and UE >= 11)
        or
        (UE <= 26 and UE >= 21))
);

CREATE TABLE controle (
    num_controle int PRIMARY KEY,
    num_module float REFERENCES module (num_module) ON DELETE
CASCADE,
    nom_controle varchar
);

CREATE TABLE note (
    num_etudiant int REFERENCES etudiant (num_etudiant) ON DELETE
CASCADE,
    num_controle int REFERENCES controle (num_controle) ON DELETE
CASCADE,
    note float
);


ALTER TABLE enseignant ADD
    num_module float REFERENCES module (num_module) ON DELETE
CASCADE ;
```

Partie II - Visualisée de Données

- 1) Relevé des moyennes dans chaque module de l'étudiant qui s'est connecté à la session
- 2) Relevé des notes de l'étudiant qui s'est connecté à la session
- 3) Relevé des moyennes de chaque étudiant de la promo

1) CREATE or REPLACE VIEW Releve_etudiant
AS
SELECT c.UE,m.num_module, m.nom_module, avg(n.note) as moyenne
FROM etudiant e, competence c, module m, controle co, note n
WHERE LOWER(e.prénom_etudiant) = session_user
AND m.num_module = co.num_module
AND co.num_controle = n.num_controle
AND e.num_etudiant = n.num_etudiant
AND m.num_module = c.num_module
GROUP BY c.UE, m.num_module, m.nom_module;

2) CREATE or REPLACE VIEW Releve_note_etudiant
AS
SELECT m.num_module, m.nom_module, co.nom_controle, n.note
FROM etudiant e, module m, controle co, note n
WHERE LOWER(e.prénom_etudiant) = session_user
AND m.num_module = co.num_module
AND co.num_controle = n.num_controle
AND e.num_etudiant = n.num_etudiant
GROUP BY m.num_module, m.nom_module, co.nom_controle, n.note ;

3) CREATE or REPLACE VIEW Releve_promo
AS
SELECT e.nom_etudiant, e.prénom_etudiant, e.groupe, avg(n.note) as moyenne
FROM etudiant e, note n
WHERE e.num_etudiant = n.num_etudiant
GROUP BY e.nom_etudiant, e.prénom_etudiant, e.groupe;

Partie III - Restrictions d'accès aux Données

- 1) un étudiant ne peut consulter que ses notes de contrôles
- 2) un étudiant peut consulter ses moyennes dans chaque module
- 3) un enseignant peut consulter les relevés de notes de la promo
- 4) un enseignant peut entrer ou modifier les notes des contrôles
- 5) un responsable module peut consulter ses contrôles et ajouter des contrôles

1) CREATE or REPLACE VIEW Releve_etudiant
AS
SELECT c.UE,m.num_module, m.nom_module, avg(n.note) as moyenne
FROM etudiant e, competence c, module m, controle co, note n
WHERE LOWER(e.prénom_etudiant) = session_user
AND m.num_module = co.num_module
AND co.num_controle = n.num_controle
AND e.num_etudiant = n.num_etudiant
AND m.num_module = c.num_module
GROUP BY c.UE, m.num_module, m.nom_module;

GRANT SELECT ON Releve_etudiant to etudiant;

2) CREATE or REPLACE VIEW Releve_note_etudiant
AS
SELECT m.num_module, m.nom_module, co.nom_controle, n.note
FROM etudiant e, module m, controle co, note n
WHERE LOWER(e.prénom_etudiant) = session_user
AND m.num_module = co.num_module
AND co.num_controle = n.num_controle
AND e.num_etudiant = n.num_etudiant
GROUP BY m.num_module, m.nom_module, co.nom_controle, n.note ;

GRANT SELECT ON Releve_note_etudiant to etudiant;

3) CREATE or REPLACE VIEW Releve_promo
AS
SELECT e.nom_etudiant, e.prénom_etudiant, e.groupe, avg(n.note) as moyenne
FROM etudiant e, note n
WHERE e.num_etudiant = n.num_etudiant
GROUP BY e.nom_etudiant, e.prénom_etudiant, e.groupe;

GRANT SELECT ON Releve_promo to enseignant;

4)
CREATE OR REPLACE FUNCTION Notes (in num_etudiant, in num_controle, note)
AS
$$
DECLARE
a INT;

BEGIN
SELECT a = COUNT(*) FROM note
WHERE num_etudiant= $1 and num_controle= $2;

IF a > 0 THEN
UPDATE Note
SET note = $3
WHERE num_etudiant= $1 and num_controle= $2;
ELSE
INSERT INTO note (num_etudiant,num_controle,note)
VALUES ($1, $2, $3);
END IF;
END;
$$
LANGUAGE PLPGSQL
GRANT SELECT,INSERT,UPDATE ON note to enseignant;

5)

```
CREATE OR REPLACE VIEW resmodule
AS
        SELECT c.nom_controle, c.num_controle, c.num.module
        FROM contrôle c, module m, enseignant e
        WHERE c.num_module = m.num_module
        AND m.resp_module = e.num_enseignant
        AND LOWER(e.nom_enseignant) = session_user;

GRANT SELECT, INSERT, UPDATE on controle to resp_module;
```