SAE S2.04 Exploitation d'une Base de Données

Mathis VERCHERE
Michael ATICI
Axel DELLUC

II Modélisation de Données

1) Cahier des charges :

Les rôles :

- ➔ Étudiant :
  - ◆ Identifiant unique (ID)
  - ◆ Nom
  - ◆ Prénom
  - ◆ Date de naissance
  - ◆ Adresse

- ➔ Promotion :
  - ◆ Code unique
  - ◆ Nom de la promotion

- ➔ Groupe :
  - ◆ Nom unique
  - ◆ ID de la promotion associée

- ➔ Matière :
  - ◆ Code unique
  - ◆ Nom de la matière
  - ◆ Professeur assigné

- ➔ Professeur :
  - ◆ Identifiant unique (ID)
  - ◆ Nom
  - ◆ Prénom

- ➔ Contrôle :
  - ◆ Identifiant unique (ID)
  - ◆ Matiere (ID)
  - ◆ Date

- ➔ Note :
  - ◆ ID de l'étudiant associé
  - ◆ ID du Contrôle
  - ◆ Code de la matière associée
  - ◆ Date du contrôle
  - ◆ Note obtenue

Les Fonctionnalités :

Ajouter, modifier et supprimer des étudiants, promotions, groupes, matières et professeurs.
Associer les étudiants à leurs groupes et les groupes à leurs promotions.
Enregistrer les notes des étudiants pour chaque matière et chaque contrôle.
Calculer les moyennes des étudiants par matière, groupe, promotion et au global.
Obtenir les meilleures et les pires notes pour chaque matière.
Effectuer des recherches et des requêtes pour filtrer les données selon des critères spécifiques.
Générer des rapports sur les résultats des étudiants, les moyennes, etc.

Gestion des données dérivées :

- ➔ Relevé de notes :
  Générer un relevé de notes pour chaque étudiant, comprenant toutes les matières étudiées et les notes obtenues.
  Le relevé de notes devrait inclure les informations supplémentaires telles que le nom de l'étudiant, son identifiant, la promotion, etc.
  Les relevés de notes peuvent être générés pour une période spécifique (par exemple, un semestre, une année scolaire, etc.).
- ➔ Bilans :
  Générer des bilans de performance globale pour chaque matière, groupe, promotion, etc.
  Les bilans peuvent inclure des statistiques telles que les moyennes, les écarts types, les médianes, les taux de réussite, etc.

Restrictions d'accès :

- ➔ Étudiant :
  - ◆ Un étudiant devrait pouvoir accéder uniquement à ses propres notes et informations personnelles.
  - ◆ L'accès à d'autres données, telles que les notes d'autres étudiants ou les évaluations des enseignants, devrait être restreint.

- ➔ Enseignant :
  - ◆ Un enseignant devrait pouvoir accéder aux notes des étudiants des matières qu'il enseigne.
  - ◆ L'enseignant peut entrer, modifier et consulter les notes pour les contrôles qu'il administre.

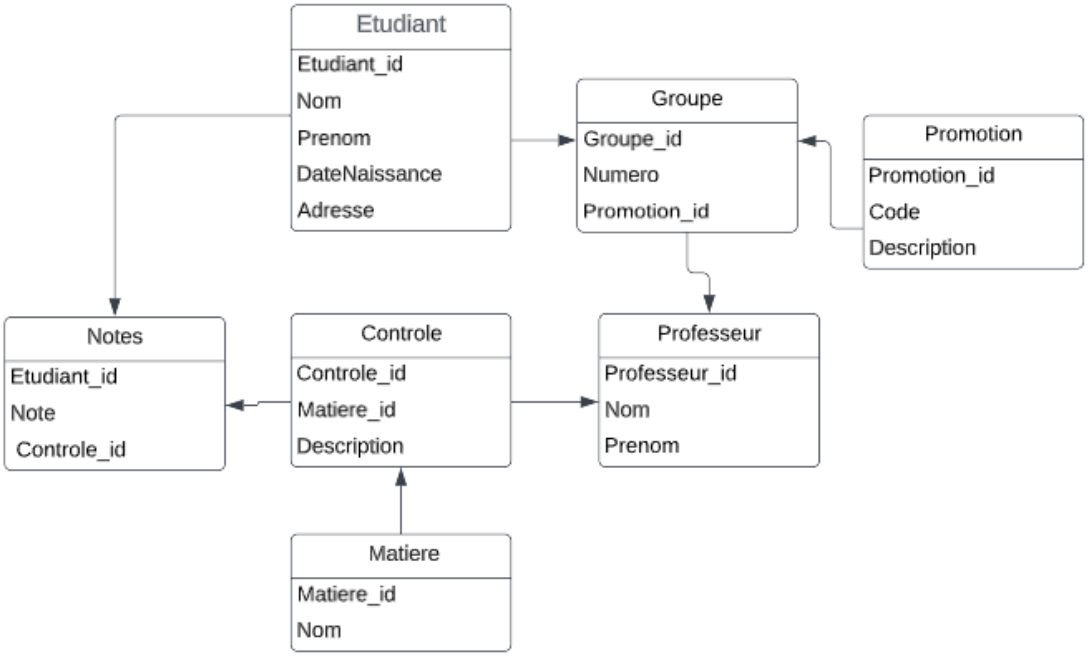
- ➔ Responsable de matière :
  - ◆ Le responsable de matière devrait avoir accès aux notes de tous les étudiants dans la matière dont il est responsable.

- ◆ Il peut consulter les statistiques, générer des rapports et effectuer des analyses sur les résultats des étudiants dans cette matière.

➔ Administrateur :

- ◆ L'administrateur aura un accès complet à toutes les fonctionnalités et à toutes les données de la base de données.
- ◆ Il sera responsable de la gestion des utilisateurs, des droits d'accès et de la maintenance générale du système.

2) La Base de données:



3)Les regles de gestion et leurs mises en oeuvre:

Ajouter une nouvelle promotion :

```
CREATE PROCEDURE AjouterPromotion(
   IN p_Code varchar(10),
   IN p_Description varchar(100)
)
AS
BEGIN
   INSERT INTO Promotion (Code, Description)
   VALUES (p_Code, p_Description);
END;
```

Associer un groupe à une promotion :

```
CREATE PROCEDURE AssocierGroupePromotion(
    IN p_GroupeNumero varchar(10),
    IN p_PromotionCode varchar(10)
)
AS
BEGIN
    DECLARE @PromotionId int;

    SELECT @PromotionId = Promotion_id
    FROM Promotion
    WHERE Code = p_PromotionCode;

    INSERT INTO Groupe (Numero, Promotion_id)
    VALUES (p_GroupeNumero, @PromotionId);
END;
```

Insérer un nouvel étudiant :

```
CREATE PROCEDURE AjouterEtudiant(
    IN p_Nom varchar(50),
    IN p_Prenom varchar(50),
    IN p_DateNaissance date,
    IN p_Adresse varchar(100)
)
AS
BEGIN
    INSERT INTO Etudiant (Nom, Prenom, DateNaissance, Adresse)
    VALUES (p_Nom, p_Prenom, p_DateNaissance, p_Adresse);
END;
```

Enregistrer une note pour un étudiant dans un contrôle :

```
CREATE PROCEDURE EnregistrerNote(
    IN p_EtudiantId int,
    IN p_ControleId int,
    IN p_Note decimal(4,2)
)
AS
BEGIN
    INSERT INTO Notes (Etudiant_id, Controle_id, Note)
    VALUES (p_EtudiantId, p_ControleId, p_Note);
END;
```

4) script de création de la base de données

```
CREATE TABLE Etudiant
(
   Etudiant_id serial PRIMARY KEY,
   Nom varchar(50),
   Prenom varchar(50),
   DateNaissance date,
   Adresse varchar(100)
);

CREATE TABLE Promotion
(
   Promotion_id serial PRIMARY KEY,
   Code varchar(10),
   Description varchar(100)
);

CREATE TABLE Groupe
(
   Groupe_id serial PRIMARY KEY,
   Numero varchar(10),
   Promotion_id int REFERENCES Promotion(Promotion_id)
);

CREATE TABLE Matiere
(
   Matiere_id varchar(10) PRIMARY KEY,
   Nom varchar(100)
);

CREATE TABLE Professeur
(
   Professeur_id serial PRIMARY KEY,
   Nom varchar(50),
   Prenom varchar(50)
);

CREATE TABLE Controle
(
   Controle_id serial PRIMARY KEY,
   Matiere_id varchar(10) REFERENCES Matiere(Matiere_id),
   Description varchar(100)
);

CREATE TABLE Notes
(
   Etudiant_id int REFERENCES Etudiant(Etudiant_id),
   Controle_id int REFERENCES Controle(Controle_id),
   Note decimal(4,2),
```

```
    PRIMARY KEY(Etudiant_id, Controle_id)
);


CREATE OR REPLACE PROCEDURE AjouterEtudiant(
    IN p_Nom varchar(50),
    IN p_Prenom varchar(50),
    IN p_DateNaissance date,
    IN p_Adresse varchar(100)
)
AS
BEGIN
    INSERT INTO Etudiant (Nom, Prenom, DateNaissance, Adresse)
    VALUES (p_Nom, p_Prenom, p_DateNaissance, p_Adresse);
END;


CREATE OR REPLACE PROCEDURE AjouterPromotion(
    IN p_Code varchar(10),
    IN p_Description varchar(100)
)
AS
BEGIN
    INSERT INTO Promotion (Code, Description)
    VALUES (p_Code, p_Description);
END;


CREATE OR REPLACE PROCEDURE EnregistrerNote(
    IN p_EtudiantId int,
    IN p_ControleId int,
    IN p_Note decimal(4,2)
)
AS
BEGIN
    INSERT INTO Notes (Etudiant_id, Controle_id, Note)
    VALUES (p_EtudiantId, p_ControleId, p_Note);
END;
```

III Visualisation de Données

1) **Ensemble de données dérivées**

➔ Relevé des notes de chaque étudiant :

- ◆ Afficher le relevé des notes de chaque étudiant pour l'ensemble des contrôles et matières.

- Regrouper les notes par étudiant et inclure les informations telles que le nom de l'étudiant, l'identifiant, la promotion, etc.
- Afficher les notes en ordre chronologique ou par matière, selon les besoins.

➔ Relevé de notes d'un groupe :

- Afficher le relevé des notes de tous les étudiants d'un groupe spécifique.
- Regrouper les notes par étudiant et inclure les informations sur le groupe et la promotion.
- Afficher les notes en ordre chronologique ou par matière, selon les besoins.

➔ Bilan de performance par matière :

- Calculer la moyenne des notes obtenues par les étudiants dans chaque matière.
- Inclure des statistiques supplémentaires telles que l'écart-type, la médiane, le taux de réussite, etc.
- Afficher les résultats par matière, en mettant en évidence les matières où les étudiants ont obtenu les meilleures et les pires notes.

➔ Bilan de performance par groupe ou promotion :

- Calculer la moyenne des notes obtenues par les étudiants dans chaque groupe ou promotion.
- Inclure des statistiques supplémentaires pour évaluer les performances globales du groupe ou de la promotion.
- Afficher les résultats par groupe ou promotion, en mettant en évidence les groupes ou promotions ayant obtenu les meilleures et les pires notes.

➔ Évolution des notes au fil du temps :

- Suivre l'évolution des notes des étudiants dans le temps pour chaque matière ou contrôle.
- Tracer des graphiques ou des diagrammes pour visualiser les tendances et les variations des notes.
- Permettre la sélection d'un étudiant, d'une matière ou d'un contrôle spécifique pour une visualisation plus détaillée.

2) Procédures, vues ou vues matérialisées

Procédure stockée : Obtention du relevé des notes de chaque étudiant

```
CREATE OR REPLACE PROCEDURE ObtenirReleveNotesEtudiant(
   IN p_EtudiantId int
)
AS
BEGIN
   SELECT *
   FROM Moyennes_matiere
   WHERE etudiant_id = p_EtudiantId;
END;
```

Cette procédure stockée prend l'identifiant d'un étudiant en paramètre (p_EtudiantId) et renvoie le relevé des notes de cet étudiant en utilisant la vue "Moyennes_matiere".

```
CREATE OR REPLACE PROCEDURE CalculerMoyennePonderee(
   IN p_EtudiantId int
)
AS
DECLARE
   totalCoefficients decimal(4,2);
   totalPonderedSum decimal(4,2);
BEGIN
   -- Calcul du total des coefficients
   SELECT SUM(coefficients)
   INTO totalCoefficients
   FROM Matiere;

   -- Calcul de la somme pondérée des notes
   SELECT SUM(n.note * m.coefficients)
   INTO totalPonderedSum
   FROM Notes n
   JOIN Matiere m ON n.Matiere_id = m.Matiere_id
   WHERE n.Etudiant_id = p_EtudiantId;

   -- Calcul de la moyenne pondérée
   IF totalCoefficients > 0 THEN
      RAISE NOTICE 'Moyenne pondérée de l''étudiant % : %',
         p_EtudiantId, totalPonderedSum / totalCoefficients;
   ELSE
      RAISE NOTICE 'Aucun coefficient trouvé.';
   END IF;
END;
```

Cette procédure stockée prend l'identifiant d'un étudiant en paramètre (p_EtudiantId). Elle calcule d'abord le total des coefficients de toutes les matières, puis calcule la somme pondérée des notes de l'étudiant en utilisant la table Notes et la table Matiere. Ensuite, elle calcule la moyenne pondérée en divisant la somme pondérée par le total des coefficients.

IV Restrictions d’accès aux données

1) Définir des règles daccès aux données

1. Accès restreint aux données des étudiants :
    - Création d’un rôle "étudiant" avec des privilèges limités, qui peut uniquement accéder à ses propres données (notes, informations personnelles, etc.).
    - Appliquation des contraintes de sécurité pour empêcher les étudiants d'accéder aux données des autres étudiants.

2. Accès restreint aux données des enseignants :
    - Création d'un rôle "enseignant" avec des privilèges spécifiques pour accéder et gérer les notes des étudiants dans les matières qu'ils enseignent.
    - Limitation d''accès des enseignants aux seules données pertinentes pour leurs cours en utilisant des filtres dans les requêtes ou des vues spécifiques.
3. Accès restreint aux données des responsables de matière :
    - Création d'un rôle "responsable_matiere" avec des privilèges étendus pour gérer les matières, y compris l'accès aux notes des étudiants dans ces matières.
    - Définir des contraintes pour limiter l'accès des responsables de matière aux données de leurs matières spécifiques.
4. Accès restreint aux données sensibles :
    - Identification des données sensibles (par exemple, les données personnelles des étudiants) et appliquez des contraintes de confidentialité pour limiter leur accès uniquement aux rôles autorisés.
    - Utilisation des vues avec des filtres pour afficher uniquement les informations nécessaires pour chaque rôle.

2) Les procédures ou vue pour mettre en oeuvre ces règles

Obtention du relevé des notes d'un étudiant spécifique :

```
CREATE OR REPLACE PROCEDURE ObtenirReleveNotesEtudiant(
   IN p_EtudiantId int
)
AS
BEGIN
   SELECT *
   FROM Notes
   WHERE Etudiant_id = p_EtudiantId;
END;
```

Calcul de la moyenne pondérée des notes d'un étudiant :

```
CREATE OR REPLACE PROCEDURE CalculerMoyennePonderee(
   IN p_EtudiantId int
)
AS
DECLARE
   totalCoefficients decimal(4,2);
   totalPonderedSum decimal(4,2);
BEGIN

   SELECT SUM(Coefficient)
   INTO totalCoefficients
   FROM Matiere;
```

```
    SELECT SUM(n.Note * m.Coefficient)
    INTO totalPonderedSum
    FROM Notes n
    JOIN Matiere m ON n.Matiere_id = m.Matiere_id
    WHERE n.Etudiant_id = p_EtudiantId;

    IF totalCoefficients > 0 THEN
        RAISE NOTICE 'Moyenne pondérée de l''étudiant % : %',
            p_EtudiantId, totalPonderedSum / totalCoefficients;
    ELSE
        RAISE NOTICE 'Aucun coefficient trouvé.';
    END IF;
END;
```

Relevé des notes d'un groupe spécifique :

```
CREATE VIEW ReleveNotesGroupe AS
SELECT Groupe.Numero AS GroupeNumero, Etudiant.Etudiant_id, Etudiant.Nom,
Etudiant.Prenom, Matiere.Matiere_id, Notes.Note
FROM Groupe
JOIN Etudiant ON Groupe.Groupe_id = Etudiant.Groupe_id
JOIN Notes ON Etudiant.Etudiant_id = Notes.Etudiant_id
JOIN Matiere ON Notes.Matiere_id = Matiere.Matiere_id;
```

Savoir si l’utilisateur a le droit d’ajouter ou modifier une note

```
CREATE OR REPLACE PROCEDURE AjouterNote(
    IN p_EtudiantId int,
    IN p_ControleId int,
    IN p_Note decimal(4,2)
)
AS
BEGIN
    -- Vérifier les privilèges de l'utilisateur
    IF (SELECT COUNT(*) FROM UtilisateurPrivileges WHERE Utilisateur_id = session_user
AND Privilege = 'AjouterNote') > 0 THEN
        -- Insérer la note
        INSERT INTO Notes (Etudiant_id, Controle_id, Note)
        VALUES (p_EtudiantId, p_ControleId, p_Note);
        RAISE NOTICE 'Note ajoutée avec succès.';
    ELSE
        RAISE EXCEPTION 'Accès refusé. Vous n''avez pas les droits nécessaires pour ajouter
une note.';
    END IF;
END;
```