TP SQL :

Gestion sécurisée des données – Sup de Vinci

Contexte

SupdeVinci souhaite mettre en place une base de données pour gérer :

- **Les étudiants**,
- **Les cours**,
- **Les inscriptions**.

La sécurité est essentielle :

- Seuls les professeurs peuvent ajouter ou modifier des informations.
- Les étudiants peuvent uniquement consulter leurs données et les cours disponibles.

Ce TP permet de comprendre :

- Le **contrôle d'accès** en SQL,
- La gestion des **rôles**,
- La **sécurisation des données**.

Étape 1 : Création des utilisateurs et rôles

Énoncé

1. Créez deux utilisateurs :
    - `prof_alice` (professeur)
    - `etudiant_bob` (étudiant)
2. Créez deux rôles : `professeur` et `etudiant`.
3. Attribuez chaque rôle aux utilisateurs correspondants.

Aide pratique

Commande	Usage	Exemple
`CREATE USER <nom> IDENTIFIED BY <motdepasse>;`	Crée un utilisateur	`CREATE USER prof_alice IDENTIFIED BY 'Password123';`
`CREATE ROLE <nom>;`	Crée un rôle	`CREATE ROLE professeur;`
`GRANT <role> TO <utilisateur>;`	Attribue un rôle	`GRANT professeur TO prof_alice;`
`REVOKE <role> FROM <utilisateur>;`	Retire un rôle	`REVOKE professeur FROM prof_alice;`

Étape 2 : Création des tables

Énoncé

Créez les tables suivantes :

1. **Etudiants** : `id_etudiant`, `nom`, `prenom`, `email`
2. **Cours** : `id_cours`, `nom_cours`, `enseignant`
3. **Inscriptions** : `id_inscription`, `id_etudiant`, `id_cours`

- Avec **clés étrangères** vers `Etudiants` et `Cours`.

Aide pratique

Commande	Usage	Exemple
`CREATE TABLE <nom>`	Crée une table	`CREATE TABLE Etudiants (id_etudiant INT PRIMARY KEY, nom VARCHAR(50), prenom VARCHAR(50), email VARCHAR(100));`
`PRIMARY KEY`	Définir la clé primaire	`id_etudiant INT PRIMARY KEY`
`FOREIGN KEY`	Définir une clé étrangère	`FOREIGN KEY (id_etudiant) REFERENCES Etudiants(id_etudiant)`

Étape 3 : Attribution des droits

Énoncé

- Étudiants : **lecture seule** sur toutes les tables.
- Professeurs : **lecture, ajout et modification**, mais **interdiction de suppression**.

Aide pratique

Commande	Usage	Exemple
GRANT SELECT ON <table> TO <role>;	Autorise la lecture	`GRANT SELECT ON Etudiants TO etudiant;`
GRANT INSERT ON <table> TO <role>;	Autorise l'ajout	`GRANT INSERT ON Etudiants TO professeur;`
GRANT UPDATE ON <table> TO <role>;	Autorise la modification	`GRANT UPDATE ON Etudiants TO professeur;`
REVOKE DELETE ON <table> FROM <role>;	Retire la suppression	`REVOKE DELETE ON Etudiants FROM professeur;`

Étape 4 : Tests pratiques

Énoncé

1. Connectez-vous en tant qu'étudiant (`etudiant_bob`) :
    - Lecture → autorisée
    - Ajout / modification → refusée
2. Connectez-vous en tant que professeur (`prof_alice`) :
    - Lecture, ajout, modification → autorisées
    - Suppression → refusée

Aide pratique – Requêtes

Étudiant :

-- Lecture autorisée

```
SELECT * FROM Etudiants;
SELECT * FROM Cours;
```

-- Ajout ou modification interdit

```
INSERT INTO Etudiants (id_etudiant, nom, prenom, email) VALUES (1,
'Dupont', 'Alice', 'alice@supdevinci.fr');
UPDATE Etudiants SET email='alice2@supdevinci.fr' WHERE id_etudiant=1;
```

Professeur :

-- Lecture autorisée

```
SELECT * FROM Etudiants;
```

-- Ajout autorisé

```
INSERT INTO Etudiants (id_etudiant, nom, prenom, email) VALUES (2,
'Martin', 'Bob', 'bob@supdevinci.fr');
```

-- Modification autorisée

```
UPDATE Etudiants SET email='bob2@supdevinci.fr' WHERE id_etudiant=2;
```

-- Suppression interdite

```
DELETE FROM Etudiants WHERE id_etudiant=2;
```

Aide pratique – Vérification des droits

```
SHOW GRANTS FOR 'etudiant_bob'@'localhost';
SHOW GRANTS FOR 'prof_alice'@'localhost';
```

Étape 5 : Sécurisation supplémentaire

Partie audit optionnel (cf TP précédent)

Énoncé

- Chiffrez les emails des étudiants.
- Créez une table **Audit** pour enregistrer les modifications faites par les professeurs.

Aide pratique – Exemples

Chiffrement (MySQL) :

```
INSERT INTO Etudiants (id_etudiant, nom, prenom, email)
VALUES (3, 'Durand', 'Claire', AES_ENCRYPT('claire@supdevinci.fr',
'ma_cle'));
```

Table Audit :

```
CREATE TABLE Audit (
    id_audit INT PRIMARY KEY,
    utilisateur VARCHAR(50),
    action VARCHAR(50),
    table_modifiee VARCHAR(50),
    date_action TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Connexion et exécution des requêtes

MySQL Workbench

1. Crée une nouvelle connexion avec `etudiant_bob` ou `prof_alice`.
2. Sélectionne la base `SupdeVinci`.
3. Ouvre un onglet SQL et exécute les requêtes.

VS Code

1. Installe l'extension **MySQL** ou **SQL Server** selon ton SGBD.
2. Crée une connexion vers `localhost` avec l'utilisateur choisi.
3. Ouvre un fichier `.sql` et lance les requêtes (F5 ou "Run Query").