Rapport : EFF – Gestion des véhicules de service de la société DKM

Module : Base de Données MySQL

Nom: Aya ouazzani akkar hammoudi

SOMMAIRE :

1. Introduction et objectif de l'atelier

2. Création de la base et des tables (captures)

3. Implémentation de chaque question (code + exécution)

4. Résultats et observations

1. Introduction et objectif de l'atelier:

Cet atelier a pour objectif de mettre en pratique la manipulation des **procédures stockées**, **fonctions** et **déclencheurs (triggers)** dans MySQL.
À partir du schéma relationnel de la société **DKM**, nous allons automatiser certaines opérations de gestion liées aux salariés, aux voitures et à leurs utilisations.
L'objectif est de renforcer la maîtrise du langage SQL procédural et la gestion de la logique métier directement au niveau de la base de données.

2. Création de la base et des tables:

Création de la base :

```
1 CREATE DATABASE dkm;
2 USE dkm;
```

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0013 seconde(s).)

```
CREATE DATABASE dkm;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Error: #1046 Aucune base n'a été sélectionnée

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0006 seconde(s).)

```
USE dkm;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Création des tables

```sql
CREATE TABLE service (
  codeSer INT PRIMARY KEY AUTO_INCREMENT,
  nomSer VARCHAR(50)
);

CREATE TABLE salarie (
  codeSal INT PRIMARY KEY AUTO_INCREMENT,
  nomSal VARCHAR(50),
  prenomSal VARCHAR(50),
  dateEmbauche DATE,
  dateNaissance DATE,
  fonction VARCHAR(50),
  codeSer INT,
  FOREIGN KEY (codeSer) REFERENCES service(codeSer)
);
```

```sql
CREATE TABLE voiture (
  matricule VARCHAR(20) PRIMARY KEY,
  marque VARCHAR(50),
  couleur VARCHAR(30),
  dateMiseEnCirculation DATE
);

CREATE TABLE utilisation (
  matricule VARCHAR(20),
  codeSal INT,
  dateDebutUtilisation DATE,
  dateFinUtilisation DATE,
  PRIMARY KEY (matricule, codeSal, dateDebutUtilisation),
  FOREIGN KEY (matricule) REFERENCES voiture(matricule),
  FOREIGN KEY (codeSal) REFERENCES salarie(codeSal)
);
```

```sql
CREATE TABLE garagiste (
  codeGar INT PRIMARY KEY AUTO_INCREMENT,
  nomGar VARCHAR(50),
  adresse VARCHAR(100)
);

CREATE TABLE demandeReparation (
  codeDem INT PRIMARY KEY AUTO_INCREMENT,
  dateDem DATE,
  codeSal INT,
  descriptionDem TEXT,
  codeGar INT,
  syntheseReparation TEXT,
  dateFinReparation DATE,
  FOREIGN KEY (codeSal) REFERENCES salarie(codeSal),
  FOREIGN KEY (codeGar) REFERENCES garagiste(codeGar)
);
```

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0209 seconde(s).)

```
CREATE TABLE service ( codeSer INT PRIMARY KEY AUTO_INCREMENT, nomSer VARCHAR(50) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0859 seconde(s).)

```
CREATE TABLE salarie ( codeSal INT PRIMARY KEY AUTO_INCREMENT, nomSal VARCHAR(50), prenomSal VARCHAR(50), dateEmbauche DATE,
dateNaissance DATE, fonction VARCHAR(50), codeSer INT, FOREIGN KEY (codeSer) REFERENCES service(codeSer) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0214 seconde(s).)

```
CREATE TABLE voiture ( matricule VARCHAR(20) PRIMARY KEY, marque VARCHAR(50), couleur VARCHAR(30), dateMiseEnCirculation DATE );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0803 seconde(s).)

```
CREATE TABLE utilisation ( matricule VARCHAR(20), codeSal INT, dateDebutUtilisation DATE, dateFinUtilisation DATE, PRIMARY KEY
(matricule, codeSal, dateDebutUtilisation), FOREIGN KEY (matricule) REFERENCES voiture(matricule), FOREIGN KEY (codeSal) REFERENCES
salarie(codeSal) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0194 seconde(s).)

```
CREATE TABLE garagiste ( codeGar INT PRIMARY KEY AUTO_INCREMENT, nomGar VARCHAR(50), adresse VARCHAR(100) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0956 seconde(s).)

```
CREATE TABLE demandeReparation ( codeDem INT PRIMARY KEY AUTO_INCREMENT, dateDem DATE, codeSal INT, descriptionDem TEXT, codeGar INT,
syntheseReparation TEXT, dateFinReparation DATE, FOREIGN KEY (codeSal) REFERENCES salarie(codeSal), FOREIGN KEY (codeGar) REFERENCES
garagiste(codeGar) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

3. Implémentation de chaque question :

1) afficher le nombre de salariés de chaque service :

```
DELIMITER $$

CREATE PROCEDURE nb_salaries_par_service()
BEGIN
  SELECT s.nomSer AS Service, COUNT(sa.codeSal) AS Nombre_Salaries
  FROM service s
  LEFT JOIN salarie sa ON s.codeSer = sa.codeSer
  GROUP BY s.nomSer;
END $$

DELIMITER ;
```

```
INSERT INTO service (nomSer) VALUES
('Informatique'),
('RH'),
('Marketing');

INSERT INTO salarie (nomSal, prenomSal, dateEmbauche, dateNaissance, fonction, codeSer) VALUES
('Dupont', 'Jean', '2020-01-01', '1990-05-05', 'Technicien', 1),
('Martin', 'Sophie', '2019-03-15', '1988-07-10', 'Chef de projet', 1),
('Durand', 'Paul', '2021-02-10', '1992-09-12', 'RH', 2),
('Bernard', 'Claire', '2022-04-20', '1995-11-22', 'Marketing', 3);
```

```
1 CALL nb_salaries_par_service();
```

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0013 seconde(s).)

```
CALL nb_salaries_par_service();
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Tout afficher | Nombre de lignes : 25 Filtrer les lignes: Chercher dans cette table

Options supplémentaires

Service	Nombre_Salaries
Informatique	2
Marketing	1
RH	1

2)retourner le code du salarié utilisant une voiture selon matricule + date :

```
1  DELIMITER //
2
3  CREATE FUNCTION getCodeSalarie(matricule_voiture VARCHAR(20), date_utilisation DATE)
4  RETURNS INT
5  DETERMINISTIC
6  BEGIN
7    DECLARE code INT;
8
9    SELECT codeSal INTO code
10   FROM utilisation
11   WHERE matricule = matricule_voiture
12     AND date_utilisation BETWEEN dateDebutUtilisation AND dateFinUtilisation;
13
14   RETURN code;
15 END //
16
17 DELIMITER ;
```

```
1 SELECT getCodeSalarie('A12345', '2024-10-01') AS Code_Salarie;
```

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0187 seconde(s).)

```
CREATE FUNCTION getCodeSalarie(matricule_voiture VARCHAR(20), date_utilisation DATE) RETURNS INT DETERMINISTIC BEGIN DECLARE code INT;
SELECT codeSal INTO code FROM utilisation WHERE matricule = matricule_voiture AND date_utilisation BETWEEN dateDebutUtilisation AND
dateFinUtilisation; RETURN code; END;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Affichage des lignes 0 - 0 (total de 1, traitement en 0,0019 seconde(s).)

```
SELECT getCodeSalarie('A12345', '2024-10-01') AS Code_Salarie;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actualiser]

Tout afficher | Nombre de lignes : 25 Filtrer les lignes: Chercher dans cette table

Options supplémentaires

Code_Salarie
NULL

3) empêcher une affectation de voiture à un salarié du service ‘informatique’ :

```sql
DELIMITER $$

CREATE TRIGGER verif_service_informatique
BEFORE INSERT ON utilisation
FOR EACH ROW
BEGIN
  DECLARE nomService VARCHAR(50);

  SELECT s.nomSer INTO nomService
  FROM service s
  JOIN salarie sa ON s.codeSer = sa.codeSer
  WHERE sa.codeSal = NEW.codeSal;

  IF nomService = 'informatique' THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Erreur : Les salariés du service informatique ne peuvent pas utiliser de voiture.';
  END IF;
END $$

DELIMITER ;
```

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0251 seconde(s).)

```sql
CREATE TRIGGER verif_service_informatique BEFORE INSERT ON utilisation FOR EACH ROW BEGIN DECLARE nomService VARCHAR(50); SELECT s.nomSer INTO nomService FROM service s JOIN salarie sa ON s.codeSer = sa.codeSer WHERE sa.codeSal = NEW.codeSal; IF nomService = 'informatique' THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Erreur : Les salariés du service informatique ne peuvent pas utiliser de voiture.'; END IF; END;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

4. Résultats et observations:

La procédure a permis d'obtenir automatiquement le nombre de salariés dans chaque service sans écrire plusieurs requêtes.

La fonction facilite la recherche du salarié à partir du matricule et de la date d'utilisation.

Le déclencheur assure la **sécurité des règles métiers** et empêche des insertions incohérentes (respect de la politique de l'entreprise).