Exercices

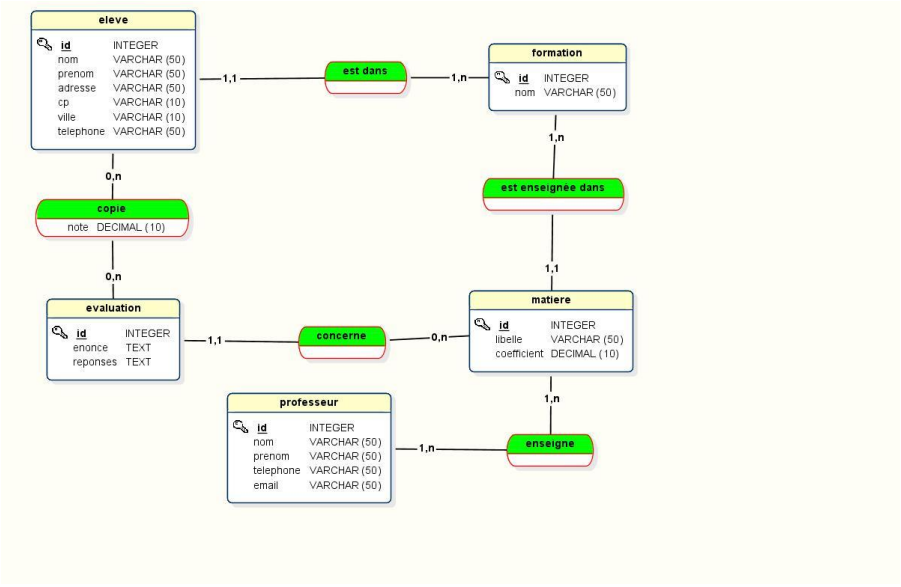
Entités :

- élèves
- formations
- évaluations
- copies
- matières
- professeurs

Règles :

- Un élève est dans une seule classe
- Dans une classe on voit plusieurs matières
- Une matière peut être enseignée par un ou plusieurs profs
- Une matière n'est enseignée que dans une seule formation
- Une matière n'est évaluée qu'une seule fois
- Une copie est le résultat de la liaison entre une évaluation et un élève. Elle porte la donnée "note"

MCD :



Exercices :

1 - Lister les différentes formations

```
SELECT * FROM formation;
```

2 - Lister les élèves de la formation "TP Dev Web"

```
SELECT *
FROM `eleve`
INNER JOIN formation ON formation.id = eleve.id_formation
AND formation.nom LIKE "%Dev Web%";
```

3 - Insérer un élève nommé "Petit Nouveau" dans la formation TP Dev Web, mettez-lui les données que vous voulez.

```
INSERT INTO eleve (nom, prenom, adresse, cp, ville, telephone, id_formation
) VALUES ("NOUVEAU", "Petit", "Rue des
nains", "03000", "Moulins", "0615211581", 1);
```

4 - Afficher le ou les profs qui enseignent la matière "Bootstrap"

```
SELECT p.*, m.libelle
FROM professeur p
INNER JOIN professeur_matiere pm ON pm.id_professeur = p.id
INNER JOIN matiere m on m.id = pm.id_matiere
WHERE m.libelle = "Bootstrap";
```

5 - Compter le nombre d'élèves total

```
SELECT COUNT(*) as nbEleve FROM `eleve`;
```

6 - Compter le nombre d'élèves dans chaque classe et les regrouper par formation (Afficher le nom de la formation et le nombre d'élèves dans une colonne à côté

```
SELECT f.nom, COUNT(*) as nbEleve
FROM `eleve` e
INNER JOIN formation f ON f.id = e.id_formation
GROUP BY f.nom;
```

7 - Lister toutes les matières que peut enseigner le professeur N°2

```sql
SELECT m.*
FROM matiere m
INNER JOIN professeur_matiere pm ON pm.id_matiere = m.id
WHERE pm.id_professeur = 2;
```

8 - Insérer une évaluation pour la matière "Bootstrap". (Reprenez l'ID de la matière Bootstrap, et mettez un Lorem Ipsum si vous n'avez pas d'inspiration pour les champs textes…)

```sql
INSERT INTO evaluation (enonce, reponses, id_matiere)
VALUES ("Faire une jolie page HTML avec Bootstrap", "Bootstrap Magic", 1);
```

9 - Donner une note à chaque élève sur la nouvelle évaluation Bootstrap (Il va falloir reprendre l'ID de l'évaluation Bootstrap que vous avez inséré et faire une insertion dans la table "Copie") Attention il y a 8 élèves donc 8 lignes à insérer !

```sql
INSERT INTO copie (id_evaluation, id_eleve, note)
VALUES (3, 7, 20.00),
(3, 8, 20.00),
(3, 9, 20.00),
(3, 10, 20.00),
(3, 11, 20.00),
(3, 12, 20.00),
(3, 13, 20.00),
(3, 14, 18.00);
```

10 - Lister les notes des élèves de la pire à la meilleure (Il faut afficher le nom de l'élève à côté de sa note. Et seulement sur l'évaluation Bootstrap que l'on vient de créer !)

```sql
SELECT e.nom, e.prenom, c.note
FROM eleve e
INNER JOIN copie c ON c.id_eleve = e.id
INNER JOIN evaluation ON evaluation.id = c.id_evaluation
INNER JOIN matiere m ON m.id = evaluation.id_matiere
WHERE m.libelle = "Bootstrap"
ORDER BY c.note;
```

11 - Oups, il y a eu une erreur de notation, l'élève "Petit Nouveau" a eu deux points oubliés sur sa copie ! Utiliser UPDATE pour mettre à jour sa note

```sql
UPDATE copie
SET note = note + 2
WHERE id_evaluation = 3
AND id_eleve = 14;
```

12 - Il est l'heure d'attribuer les diplômes : Lister tous les élèves qui ont eu au-dessus de 10

```sql
SELECT e.*, c.note
FROM eleve e
INNER JOIN copie c ON c.id_eleve = e.id
WHERE c.id_evaluation = 3
AND note > 10;
```

13 - Utiliser ALTER TABLE pour modifier la table "élève" et ajouter un champ "examen_reussi". Ce sera un booléen qui prendra 0 comme valeur par défaut.

```sql
ALTER TABLE eleve ADD examen_reussi BOOLEAN DEFAULT 0;
```

14 - Finalement, seule l'évaluation Bootstrap compte pour l'examen. On va donc utiliser UPDATE pour mettre à jour la table élève et mettre examen_reussi à 1 lorsque la note de l'évaluation Bootstrap est supérieure à 10 (Utilisez une sous-requête dans la condition WHERE. La requête de l'exercice numéro 12 par exemple…)

```sql
UPDATE eleve e
SET examen_reussi = 1
WHERE id IN (SELECT id_eleve
             FROM copie
             WHERE id_evaluation = 3
             AND note > 10);
```

Solution optimale 2x plus performante :

```sql
UPDATE eleve e INNER JOIN copie c ON c.id_eleve = e.id SET examen_reussi =
1 WHERE c.id_evaluation = 3 AND c.note > 10;
```

15 - Compter le nombre d'élèves qui ont réussi l'examen, puis compter ceux qui ne l'ont pas réussi.

```sql
SELECT COUNT(*) as nbEleve, examen_reussi as reussite
FROM eleve
WHERE id_formation = 1
GROUP BY examen_reussi ;
```