TP 6 — Jointures

L'intérêt majeur des bases de données est de pouvoir établir des liens entre les tables et de croiser les informations. Nous allons apprendre, dans ce TP, à écrire des requêtes qui permettent de joindre les données deux tables (ou davantage !) et à extraire des données issues de tables différentes.

1 Mise en place

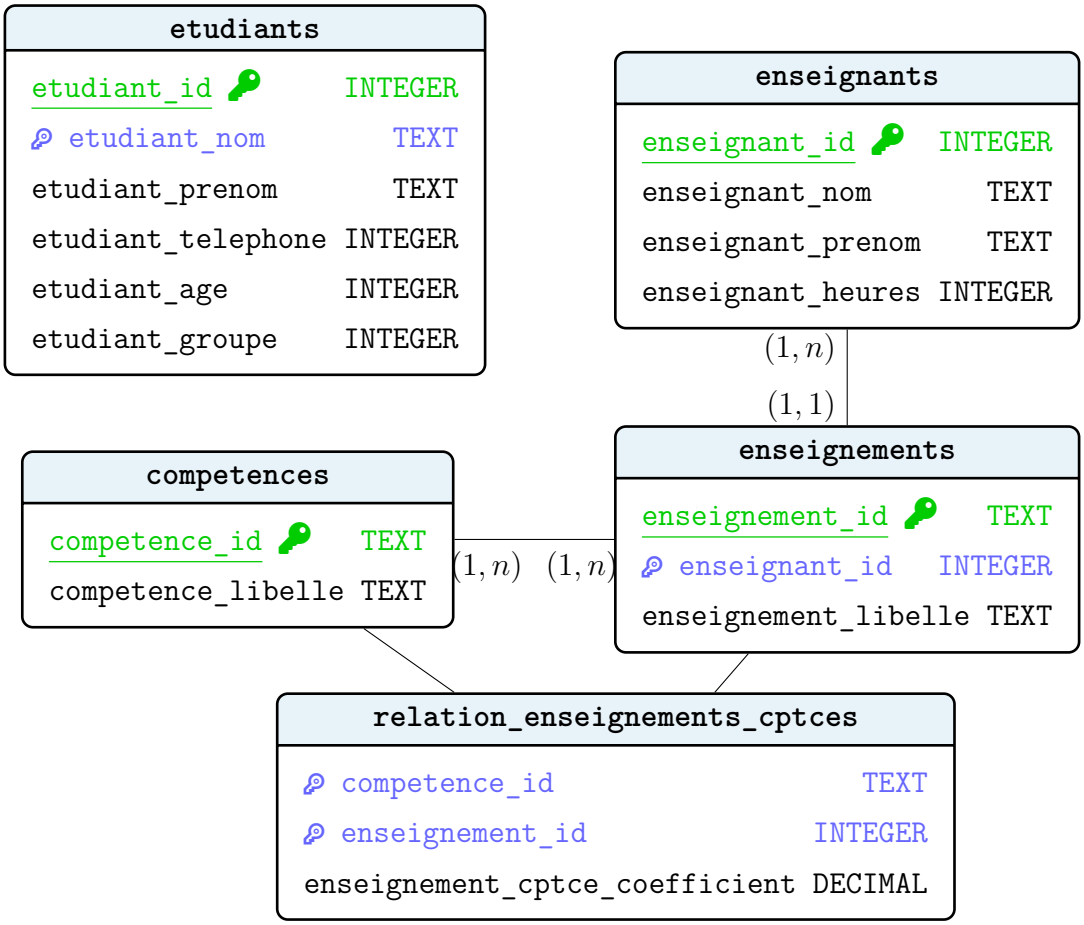
Dans l'introduction, les exemples s'appuient sur la base de donneés qui modélise (partiellement) le BUT informatique. Pour la partie exercices, nous utiliserons la base `Northwind` déjà utilisée lors des TP précédents.

À rendre

Vous déposerez sur EPREL le fichier `tp6_<nomDeFamille>.sql` contenant les requêtes écrites dans la partie exercices.

2 Notion de jointure

Nous allons utiliser la base de données qui modélise le BUT informatique, dont on rappelle le diagramme entité-relations (cf. TP nº 4) :



Reprenons une requête écrite dans le cadre du TP 2 :

```SQL
SELECT enseignements.enseignement_id, enseignements.enseignement_libelle,
       enseignants.enseignant_nom, enseignants.enseignant_prenom
FROM enseignants, enseignements
WHERE enseignements.enseignant_id = enseignants.enseignant_id;
```

Cette requête connecte les deux tables `enseignements` et `enseignants`, grâce à la clé `enseignant_id` (qui est primaire pour la table `enseignants` et étrangère pour la table `enseignements`). L'opération que nous venons de réaliser s'appelle une *jointure*. SQL propose une syntaxe spécifique pour cela, syntaxe qu'il est conseillé d'utiliser :

```SQL
SELECT enseignements.enseignement_id, enseignements.enseignement_libelle,
       enseignants.enseignant_nom, enseignants.enseignant_prenom
FROM enseignants
INNER JOIN enseignements
ON enseignements.enseignant_id = enseignants.enseignant_id;
```

Distinction entre `WHERE` et `JOIN`

Utiliser la clause `WHERE` pour créer une jointure n'est pas une bonne pratique, car ce n'est pas son rôle. La clause `WHERE` est faite pour *filtrer selon des conditions* sur les valeurs des attributs, pas pour *comparer entre elles les valeurs des attributs*.

Il est bien sûr possible de combiner les deux, si l'on souhaite effectuer une jointure *puis* filtrer les résultats obtenus selon un critère donné. Par exemple, n'afficher que les enseignements de Mme Bouaoune :

```SQL
SELECT enseignements.enseignement_id, enseignements.enseignement_libelle,
       enseignants.enseignant_nom, enseignants.enseignant_prenom
FROM enseignants
INNER JOIN enseignements
ON enseignements.enseignant_id = enseignants.enseignant_id
WHERE enseignants.enseignant_nom = "BOUAOUNE";
```

3 Exercices

Pour tous les exercices, nous allons travailler avec la base de données `Northwind`.

Exercice 1 (Produits et fournisseurs associés)**.** Afficher `ProductID`, `ProductName`, et `CompanyName` du fournisseur. Trier par `ProductID`.

Exercice 2. Afficher `OrderID`, `OrderDate`, et `CompanyName` de l'expéditeur, et trier par `OrderID`. De plus, afficher uniquement les lignes telles que `OrderID < 10255`.

Exercice 3. Trouver le nombre total de produits dans chaque catégorie. Afficher `CategoryName` et le nombre total de produits.

Une commodité de syntaxe

Vous remarquez que les requêtes deviennent de plus en plus longues à écrire et peuvent être lourdes à relire. SQL propose une syntaxe qui permet de *raccourcir les noms* des tables utilisées dans les requêtes :

```SQL
SELECT T2.enseignement_id, T2.enseignement_libelle,
       T1.enseignant_nom, T1.enseignant_prenom
FROM enseignants T1
INNER JOIN enseignements T2
ON T2.enseignant_id = T2.enseignant_id
WHERE T1.enseignant_nom = "BOUAOUNE";
```

Exercice 4. Afficher toutes les commandes d'une valeur supérieure à 12 000 $ et passées en 2016. Afficher `OrderID` et la valeur totale de la commande (ignorer `Discount`).

Exercice 5. Quels produits Michael Suyama a-t-il déjà vendus ? Afficher `EmployeeID`, `FirstName`, `LastName`, `ProductID` et `ProductName`.

Exercice 6. Trouver des clients qui n'ont jamais passé de commande.

Exercice 7. Trouver des clients qui n'ont jamais passé de commande auprès de Margaret Peacock (`EmployeeID=4`).

Exercice 8. Trouver les clients qui ont passé au moins une commande d'une valeur totale supérieure ou égale à 10 000 $ (ignorer la remise) en 2016.