Transition de l'Algèbre Relationnelle (AR) à SQL

Ce cours vise à fournir une transition claire et efficace entre les concepts fondamentaux de l'Algèbre Relationnelle (AR) et leur application pratique dans le langage SQL.

Introduction à SQL

SQL (Structured Query Language) est un langage de programmation spécialement conçu pour la gestion et la manipulation des bases de données relationnelles. Bien qu'il soit inspiré de l'Algèbre Relationnelle (branche des mathématiques et de l'informatique qui fournit un ensemble d'opérations permettant de manipuler des relations de bases de données), il a sa propre syntaxe et ses propres spécificités.

1 Équivalence entre AR et SQL

En se basant sur une base de données bibliothécaire suivante, nous explorerons comment les concepts et opérations de l'Algèbre Relationnelle se traduisent en requêtes SQL.
Étudiant (ID, Nom, Prenom, # ID_Groupe)
Groupe (ID, NomG)
Livre (ID, Titre, Auteur, Annee)
Emprunt (ID, #ID_Livre, #ID_Etudiant, Date_Emprunt , Date_Retour)

1.1 Projection

La projection en AR implique de sélectionner des colonnes spécifiques d'une relation.
En SQL, cela se fait en listant simplement les noms des colonnes désirées.
Exemple : Les Titres et les Années de tous les livres.
En AR : $\pi_{Titre,Annee}(Livres)$
Équivalent SQL : `SELECT Titre, Annee FROM Livres;`

1.2 Sélection

En AR, la sélection est utilisée pour récupérer des tuples basés sur une condition donnée.
En SQL, cela se fait avec la clause `WHERE`.
Exemple : Les livres de l'auteur A_1.
En AR : $\sigma_{Auteur='A_1'}(Livres)$
Équivalent SQL : `SELECT * FROM Livres WHERE Auteur = 'A1';`

1.3 Renommage

En AR, le renommage permet de modifier le nom d'un ou plusieurs attributs d'une relation.
En SQL, cette opération est effectuée à l'aide du mot-clé `AS`.
Exemple : Dans la relation Livres, renommer la colonne $'Titre'$ en $'NomLivre'$.
En AR : $\alpha_{Titre \leftarrow NomLivre}(Livres)$
Équivalent SQL : `SELECT ID, Titre AS NomLivre, Auteur, Annee FROM Livres;`

1.4 Union

En AR, l'union permet de fusionner les résultats de deux requêtes qui ont le même nombre et le même type de colonnes, en éliminant les doublons.
En SQL, cette opération est effectuée à l'aide de l'opérateur `UNION`.
Exemple : Les titres des livres des auteurs A_1 et A_2.
En AR : $\pi_{Titre}(\sigma_{Auteur='A_1'}(Livres)) \cup \pi_{Titre}(\sigma_{Auteur='A_2'}(Livres))$
Équivalent SQL :

```
SELECT Titre FROM Livres WHERE Auteur = 'A1'
UNION
SELECT Titre FROM Livres WHERE Auteur = 'A2';
```

1.5 Intersection

En AR, l'intersection récupère les tuples qui apparaissent à la fois dans les résultats de deux requêtes, tout en respectant le schéma de leurs relations.
En SQL, cette opération est effectuée à l'aide de l'opérateur `INTERSECT`.
Exemple : Les étudiants du groupe 1 qui ont emprunté des livres.
En AR : $\pi_{id}(\sigma_{id\_groupe=1}(Etudiant)) \cap \pi_{id\_etudiant}(Emprunt)$
Équivalent SQL :

```
SELECT id FROM Etudiant WHERE id_groupe = 1 INTERSECT SELECT id_Etudiant FROM Emprunt;
```

1.6 Différence

En AR, la différence récupère les tuples qui sont présents dans la première requête mais absents de la seconde, tout en respectant le schéma de leurs relations.
En SQL, cette opération est effectuée à l'aide de l'opérateur `EXCEPT` ou `MINUS` selon les systèmes.
Exemple : Les étudiants qui n'ont pas emprunté des livres.
En AR : $\pi_{id}(Etudiant)) - \pi_{id\_etudiant}(Emprunt)$
Équivalent SQL :

```
SELECT id FROM Etudiant EXCEPT SELECT id_Etudiant FROM Emprunt;
```

1.7 Produit Cartésien

En AR, le produit cartésien combine chaque tuple d'une relation avec chaque tuple d'une autre relation, créant ainsi une nouvelle relation.
En SQL, cette opération est effectuée simplement par une liste de tables dans la clause `FROM`, sans condition de jointure spécifique.
Exemple : Combinaison de tous les étudiants avec tous les livres.
En AR : $Etudiant \times Livre$
Équivalent SQL : `SELECT * FROM Etudiant, Livre;`

1.8 Jointure

En AR, la jointure combine les tuples de deux relations basées sur une condition commune entre elles, créant ainsi une nouvelle relation.
En SQL, cette opération est effectuée à l'aide de la clause `JOIN` avec une condition de jointure spécifique.
Exemple : Combinaison des étudiants avec les emprunts basée sur l'ID de l'étudiant.
En AR : $Etudiant \bowtie_{Etudiant.ID=Emprunt.ID\_Etudiant} Emprunt$
Équivalent SQL :

```
SELECT * FROM Etudiant JOIN Emprunt ON Etudiant.ID = Emprunt.ID_Etudiant;
-- ou bien
SELECT * FROM Etudiant , Emprunt WHERE Etudiant.ID = Emprunt.ID_Etudiant;
```

2. Exercice Répondre en SQL aux questions suivantes :

1. Quels sont les étudiants qui ont emprunté au moins un livre.
2. Quels sont les titres des livres qui n'ont jamais été empruntés par aucun étudiant ?
3. Quels sont les étudiants qui ont emprunté des livres et qui font également partie du groupe G_1 ou G_2.
4. Quels sont les étudiants qui ont emprunté des livres publiés avant 2010 et qui font également partie des groupes G_1 et G_2.
5. Identifiez les étudiants qui ont emprunté des livres écrits par un auteur A_1 mais n'ont jamais emprunté des livres publiés avant 2010.
6. Quels groupes n'ont pas d'étudiants ayant emprunté des livres de la bibliothèque ?
7. Quels sont les groupes qui n'ont pas emprunté de livres après la date 01/01/2022 ?
8. Quels sont les livres écrits par un auteur A_3 qui n'ont jamais été empruntés avant 2015 ?
9. Traduire cette expression en SQL : $\pi_{Titre,Auteur}(\sigma_{Annee>2000}(Livre \bowtie Emprunt))$