Jointures avec SQL

Dr N. BAME

Introduction

- Les jointures en SQL permettent **d'associer plusieurs tables** dans une **même requête**.
  - obtenir des résultats qui **combinent les données de plusieurs tables** de manière efficace.
- En général, les tables sont mutuellement **liées** à l'aide de **clés primaires et étrangères**.

Jointure : requêtes multi-tables

```
SELECT liste_colonnes
FROM nomTable1 , nomTable2
WHERE condition;
```

Condition contient des sous-conditions de la forme

- ***nomTable1*.a θ *nomTable2*.b** (**condition de jointure**)
  - a et b sont des attributs respectifs de ***nomTable1*** et ***nomTable2***

- *Schéma avant la projection* = **concaténation** des attributs des différentes tables Ti.

- *Résultats avant la projection* = TOUTES les combinaisons des n-uplets de T1, …, Tn dont **on ne garde que les lignes vérifiant Condition**

Remarque : jointure

Un tuple de la table ***nomTable1*** apparaît dans le résultat de la jointure s'il joint ***avec au moins 1*** tuple de ***nomTable2***

Souvent un tuple joint avec plusieurs, d'où l'utilisation fréquente de ***DISTINCT*** dans le SELECT

S'il existe un **attribut de même nom dans les deux tables**, alors il est nécessaire de **préfixe**r l'attribut **par le nom de la table** (ou par son renommage si la table est renommée)

Si la **condition de jointure est oubliée** → **Produit cartésien**

Donc si ***K* tables dans le FROM**, en général il doit avoir **au moins *K-1* conditions** de jointure dans le WHERE

Renommage de table

- La clause **FROM** déclare les variables (calcul)
  - Par défaut nom de la relation : FROM R, S
  - on peut **renommer** : **FROM R v1, S v2**

- Possibilité de **préfixer** les attributs par le **nom de la table** ou une **variable**
  - Lever les ambiguïtés de noms d'attributs

Exemples

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms et salaire des employés ?

- Noms et titres des employés qui travaillent dans un projet pendant plus de 17 mois?

Exemples

Emp(Eno, Ename, Title, City)
Pay(Title, Salary)

Project(Pno, Pname, Budget, City)
Works(Eno, Pno, Resp, Dur)

- Noms et salaire des employés ?

```
SELECT Ename, Salary
FROM   Emp, Pay
WHERE  Emp.Title = Pay.Title
```

OU

```
SELECT Ename, Salary
FROM   Emp e, Pay p
WHERE  e.Title = p.Title
```

- Noms et titres des employés qui travaillent dans un projet pendant plus de 17 mois?

```
SELECT Ename, Title
FROM   Emp, Works
WHERE  Dur > 17
AND Emp.Eno = Works.Eno
```

OU

```
SELECT Ename, Title
FROM   Emp em, Works wo
WHERE  Dur > 17
AND em.Eno = wo.Eno
```

Exemples

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Numéros et noms des projets dans lesquels a travaillé l'employé 10?

- Noms et titres des employés qui travaillent dans un projet à Paris ?

Exemples

Emp(Eno, Ename, Title, City)
Pay(Title, Salary)

Project(Pno, Pname, Budget, City)
Works(Eno, Pno, Resp, Dur)

- Numéros et noms des projets dans lesquels a travaillé l'employé 10?

```
SELECT Project.Pno, Pname
FROM   Project, Works
WHERE  Eno=10
AND Project.Pno = Works.Pno
```

- Noms et titres des employés qui travaillent dans un projet à Paris ?

```
SELECT Ename, Title
FROM   Emp E, Works W, Project P
WHERE  P.City = 'Paris'
AND    E.Eno = W.Eno
AND W.Pno = P.Pno
```

Exemples

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Analyst
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Analyst
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Analyst

PAY

TITLE	SALARY
Elect. Eng.	55000
Analyst	70000
Mech. Eng.	45000
Programmer	60000

```
SELECT ENO, ENAME, EMP.TITE, SALARY
FROM EMP, PAY
WHERE EMP.TITLE=PAY.TITLE
```

ENO	ENAME	EMP.TITLE	SALARY
E1	J. Doe	Elect. Eng.	55000
E2	M. Smith	Analyst	70000
E3	A. Lee	Mech. Eng.	45000
E4	J. Miller	Programmer	60000
E5	B. Casey	Analyst	70000
E6	L. Chu	Elect. Eng.	55000
E7	R. Davis	Mech. Eng.	45000
E8	J. Jones	Analyst	70000

Types de jointures

- Il y a **plusieurs méthodes pour associer 2 tables** ensemble :
- **INNER JOIN : jointure interne** pour retourner les enregistrements quand la **condition est vrai dans les 2 tables**. C'est l'une **des jointures les plus communes**.
- **CROSS JOIN : jointure croisée** permettant de faire **le produit cartésien** de 2 tables. En d'autres mots, permet de joindre chaque lignes d'une table avec chaque lignes d'une seconde table. Attention, le nombre de résultats est en général très élevé.
- **LEFT JOIN (ou LEFT OUTER JOIN) : jointure externe** pour retourner tous les enregistrements de la **table de gauche** (**LEFT = gauche**) même si la condition n'est pas vérifié dans l'autre table.
- **RIGHT JOIN (ou RIGHT OUTER JOIN) : jointure externe** pour retourner tous les enregistrements de la **table de droite** (RIGHT = droite) même si la condition n'est pas vérifié dans l'autre table.
- **NATURAL JOIN :** jointure naturelle entre 2 tables s'il y a au moins une colonne qui porte le même nom entre les 2 tables SQL
- **FULL JOIN (ou FULL OUTER JOIN) : jointure externe** pour **retourner les résultats** quand la condition est vrai dans au moins une des 2 tables.

Jointure croisée (produit cartésien) : CROSS JOIN

- **CROSS JOIN : ou produit cartésien** est la forme la plus simple de la jointure qui **associe chaque ligne d'une table** de base de données **à toutes** les **lignes d'une autre**.
- **Syntaxe :**

```
SELECT *
FROM TableName1 CROSS JOIN TableName2
```

- Cela nous donne les **combinaisons** de chaque ligne de la première table (**TableName1**) avec tous les enregistrements de la deuxième table (**TableName2**).

Exemple

Employe(matricule, Nom, Prenom, Tel, #numDept)
Departement(numDept, Libelle, Budget)

Produit cartesien entre **Employe** et **Departement**

SELECT *
FROM Employe**,** Departement

Ou

SELECT *
FROM Employe **cross join** Departement

Jointure interne : INNER JOIN

- **INNER JOIN ou JOIN :** la jointure interne est utilisée pour récupérer des enregistrements des deux tables qui **satisfont la condition de jointure**. C'est l'une des jointures les **plus communes**.
- **Syntaxe :**

```
SELECT *
FROM TableName1 INNER JOIN TableName2 ON Condition
```

Où **Condition** : correspond à la **condition de jointure** (qui **compare** des attribut TableName2 et des attributs de TableName2

Exemple

Employe(matricule, Nom, Prenom, Tel, #numDept) **Departement**(numDept, Libelle, Budget)

Noms des Employés et les libellés et budgets de leur Departement.

```
SELECT Nom, libelle as
NomDepartement, Budget
FROM Employe E, Departement D
WHERE E.numDept=D.numDept
```

Ou

```
SELECT Nom, libelle as NomDepartement, Budget
FROM Employe E INNER JOIN Departement D ON
E.numDept=D.numDept
```

Jointure externe : FULL JOIN

FULL JOIN ou FULL OUTER JOIN : permet de faire une **jointure externe entre 2 tables**.

L'utilisation de cette commande permet de **combiner les lignes des 2 tables**, les associer entre eux grâce à une **condition** et remplir avec des valeurs NULL si la condition n'est pas respectée.

Syntaxe

```
SELECT *
FROM TableName1 FULL JOIN TableName2 ON Condition
```

Jointure externe gauche : LEFT JOIN

LEFT JOIN ou LEFT OUTER JOIN

- jointure externe pour retourner **tous les enregistrements de la table de gauche** même si la condition **n'est pas vérifié** dans l'autre table.
- **Syntaxe**

```
SELECT *
FROM TableName1 LEFT JOIN TableName2 ON Condition
```

Exemple

Employe(matricule, Nom, Prenom, Tel, #numDept)
Departement(numDept, Libelle, Budget)

Informations sur les employés et leurs éventuels départements.

```
SELECT *
FROM Employe E LEFT OUTER JOIN Departement D
ON E.numDept=D.numDept
```

Jointure externe droite: RIGHT JOIN

RIGHT JOIN ou RIGHT OUTER JOIN

- jointure externe pour retourner **tous les enregistrements de la table de droitze** même si la condition **n'est pas vérifié** dans l'autre table.
- **Syntaxe**

```
SELECT *
FROM TableName1 RIGHT JOIN TableName2 ON Condition
```

Exemple

Employe(matricule, Nom, Prenom, Tel, #numDept)
Departement(numDept, Libelle, Budget)

Informations sur les employés et leurs éventuels départements.

```
SELECT *
FROM Employe E RIGHT OUTER JOIN Departement D
ON E.numDept=D.numDept
```

Jointure naturelle: NATURAL JOIN

- **NATURAL JOIN :** jointure naturelle entre 2 tables s'il y a au moins **une colonne qui porte le même nom** entre les 2 tables SQL.
  - Cette jointure s'effectue à la ***condition qu'il y ai des colonnes*** du **même nom et de même type** dans les 2 tables.

Syntaxe

```
SELECT *
FROM TableName1 NATURAL JOIN TableName2
```

- L'avantage d'un **NATURAL JOIN** c'est **qu'il n'y a pas besoin** d'utiliser la **clause ON**.

Exemple

Employe(matricule, Nom, Prenom, Tel, #numDept) **Departement**(numDept, Libelle, Budget)

Noms des Employés et les libellés et budgets de leur Departement.

```
SELECT Nom, libelle as
NomDepartement, Budget
FROM Employe E, Departement D
WHERE E.numDept=D.numDept
```

Ou

```
SELECT Nom, libelle as NomDepartement, Budget
FROM Employe E INNER JOIN Departement D ON
E.numDept=D.numDept
```

Exemple

Employe(matricule, Nom, Prenom, Tel, #numDept) **Departement**(numDept, Libelle, Budget)

Noms des Employés et les libellés et budgets de leur Departement.

```
SELECT Nom, libelle as NomDepartement, Budget
FROM Employe E INNER JOIN Departement D ON
E.numDept=D.numDept
```

Ou

```
SELECT Nom, libelle as NomDepartement, Budget
FROM Employe E NATURAL JOIN Departement D
```

Exercices

- Soit le schéma relationnel ci-dessous :

  Emp (Eno, Ename, #Title, City)
  Pay(Title, Salary)
  Project(Pno, Pname, Budget, City)
  Works(#Eno, #Pno, Resp, Dur)

- Exprimer les requêtes suivantes en algèbre relationnelle :

1. Noms et salaires des employés de Dakar ?
2. Noms et villes des employés ayant un salaire supérieur à 700.000 ?
3. Noms et budgets des projets où a travaillé l'employé de numéro 4 ?
4. Noms et professions des employés qui ont travaillé dans un projet pendant moins de 6 mois ?
5. Responsabilités occupées par des informaticiens dans des projets ?
6. Noms, budgets et ville des projets où a travaillé l'employé Alpha Diallo?
7. Noms, budgets et villes des projets où ont travaillé des employés ayant un salaire supérieurs à 1.200.000 ?