📚 GUIDE DE RÉVISION - JOINTURES SQL

🔍 1. CONCEPT DE BASE DES JOINTURES

```sql
SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2.column2;
```

🎯 TYPES DE JOINTURES

1. Equijointure
2. Non-équijointure
3. Jointure externe
4. Autojointure

📋 DÉTAILS PAR TYPE DE JOINTURE

A. EQUIJOINTURE

```sql
/* Structure de base */
SELECT e.empno, e.ename, d.dname
FROM emp e, dept d
WHERE e.deptno = d.deptno;
```

Points Clés

- Joint deux tables sur des colonnes de valeurs égales
- Utilise généralement clé primaire et clé étrangère
- Condition d'égalité dans la clause WHERE

Exemple Pratique

```sql
SELECT emp.empno, emp.ename, dept.dname
FROM emp, dept
WHERE emp.deptno = dept.deptno;
```

B. NON-EQUIJOINTURE

```sql
/* Structure de base */
SELECT e.ename, e.sal, s.grade
FROM emp e, salgrade s
WHERE e.sal BETWEEN s.losal AND s.hisal;
```

Points Clés

- Utilise des opérateurs de comparaison autres que l'égalité
- Souvent utilisée avec BETWEEN, >, <, >=, <=
- Utile pour les plages de valeurs

Exemple Pratique

```sql
SELECT e.ename, e.sal, s.grade
FROM emp e, salgrade s
WHERE e.sal >= s.losal
AND e.sal <= s.hisal;
```

C. JOINTURE EXTERNE

```sql
/* Structure de base */
SELECT table1.col1, table2.col2
FROM table1, table2
WHERE table1.col1 = table2.col2(+);  -- Jointure externe droite
```

Points Clés

- Utilise le symbole (+) pour inclure les lignes non correspondantes
- Peut être à droite ou à gauche
- Affiche NULL pour les valeurs sans correspondance

Exemple Pratique

```
SELECT e.ename, d.deptno, d.dname
FROM emp e, dept d
WHERE e.deptno(+) = d.deptno;  -- Montre tous les départements, même sans employés
```

D. AUTOJOINTURE

```
/* Structure de base */
SELECT w.ename || ' works for ' || m.ename
FROM emp w, emp m
WHERE w.mgr = m.empno;
```

Points Clés

- Joint une table avec elle-même
- Nécessite des alias distincts
- Utile pour les relations hiérarchiques

Exemple Pratique

```
SELECT worker.ename as "Employé",
       manager.ename as "Manager"
FROM emp worker, emp manager
WHERE worker.mgr = manager.empno;
```

⚠ PRODUIT CARTÉSIEN

À Éviter

```
/* Produit cartésien (à éviter) */
SELECT *
FROM emp, dept;  -- Sans condition WHERE
```

Causes

1. Absence de condition de jointure
2. Condition de jointure incorrecte

Solution

Toujours spécifier une condition de jointure appropriée dans la clause WHERE

💡 BONNES PRATIQUES

1. Utilisation des Alias

```
/* Au lieu de */
SELECT emp.empno, emp.ename, dept.dname
FROM emp, dept
WHERE emp.deptno = dept.deptno;

/* Préférer */
SELECT e.empno, e.ename, d.dname
FROM emp e, dept d
WHERE e.deptno = d.deptno;
```

2. Gestion des Colonnes Ambiguës

- Toujours préfixer les colonnes avec l'alias de table quand elles existent dans plusieurs tables
- Utiliser des alias explicites pour les colonnes de même nom

3. Jointures Multiples

```
SELECT c.name, o.ordid, i.itemid
FROM customers c, orders o, items i
WHERE c.custid = o.custid
AND o.ordid = i.ordid;
```

📝 EXERCICES TYPES

1. Equijointure Simple

```
/* Lister les employés avec leur département */
SELECT e.ename, d.dname
FROM emp e, dept d
WHERE e.deptno = d.deptno;
```

2. Non-Equijointure

```
/* Catégoriser les salaires */
SELECT e.ename, e.sal, s.grade
FROM emp e, salgrade s
WHERE e.sal BETWEEN s.losal AND s.hisal;
```

3. Jointure Externe

```
/* Lister tous les départements et leurs employés */
SELECT d.dname, e.ename
FROM dept d, emp e
WHERE d.deptno = e.deptno(+);
```

4. Autojointure

```
/* Lister les employés et leurs managers */
SELECT w.ename "Employé", m.ename "Manager"
FROM emp w, emp m
WHERE w.mgr = m.empno;
```

🎯 POINTS CLÉS À RETENIR

1. Toujours utiliser des conditions de jointure appropriées
2. Préférer les alias de table pour la lisibilité
3. Bien identifier le type de jointure nécessaire
4. Éviter les produits cartésiens
5. Optimiser les requêtes avec des conditions additionnelles après la jointure