CHAPITRE 6
SQL-LMD

Jointures et sous requêtes

Mr. CHAOULID

1. La jointure

- La jointure consiste à rechercher entre deux tables ayant un attribut commun (même type et même domaine de définition) toutes les lignes pour lesquels ces attributs ont la même valeur.

- les jointures vous permettent de récupérer des données de deux ou de plusieurs tables en fonction des **relations** logiques existant entre ces tables.

- **Syntaxe:**

FROM Table1 **Type de Jointure** Table2 **[ON (condition de jointure)]**

1.1 La jointure interne

- Une jointure interne est une jointure dans laquelle les valeurs des colonnes jointes sont comparées à l'aide d'un opérateur de comparaison.

- **Exemple**: *Afficher les informations du service de chaque employé.*

```
Select *
From Employe inner join Service
on Employe.Num_Service=Service.Num_Service
```

```
Select *
From Employe e inner join Service s
on e.Num_Service=s.Num_Service
```

- **Exemple**: *Afficher le nom, prénom et le nom de service de chaque employé*

```
Select e.Nom, e.Prenom, s.Nom_Service
From Employe e inner join Service s
on e.Num_Service=s.Num_Service
```

```
select e.nom,e.prenom,s.nom_service
from employe e INNER JOIN Service s
USING (num_service);
```

2. La jointure externe

- La jointure externe spécifie que en plus des les lignes retournées par la jointure interne, toutes les lignes de la table (gauche/droite) ne respectant pas la condition de jointure sont comprises dans l'ensemble de résultats et que les colonnes de sortie de l'autre table ont des valeurs NULL.
- Il y a trois types de jointures externes :
  - **Jointure externe gauche** : Toutes les lignes de la table gauche sont retournées LEFT OUTER JOIN.
  - **Jointure externe droite** : Toutes les lignes de la table droite sont retournées RIGHT OUTER JOIN.

2. La Jointure externe

- **Exemple 1:** *Jointure Externe gauche*

```
Select *
From Employe e left outer join Service s
on e.Num_Service=s.Num_Service
```

- **Exemple 2**: *Jointure externe droite*

```
Select *
From Employe e right outer join Service s
on e.Num_Service=s.Num_Service
```

3. La Jointure croisée

- Dans une jointure croisée, chaque ligne de la table de gauche est combinée à toutes les lignes de la table de droite.

- **Exemple :**

```
Select *
From Employe e cross join Service s
```

- **Remarque :** Les jointures croisées doivent être utilisées avec prudence car elles peuvent utiliser beaucoup de ressources.

4. Les sous requêtes:

- Une sous-requête est une requête imbriquée à l'intérieur d'une instruction SELECT.
- Les sous-requêtes **(Requêtes internes)** peuvent être spécifiées à de nombreux endroits, notamment à la place d'une expression.
- Une sous requête peut retourner une valeur unique (sous requêtes scalaire) comme elle peut retourner un ensemble de valeurs (sous requêtes tabulaires)
- Une sous-requête peut être utilisée partout où une expression est utilisée et doit être fermée entre parenthèses.

4. Les sous requêtes:

- **Exemple 1**: *sous requête scalaire*

```
select *
from Employe
where Salaire = (Select MAX(Salaire) from Employe)
```

- **Exemple 2**: *Sous requête tabulaire*

```
select *
from Employe
where Ville !='CASA'
and Salaire in (Select distinct Salaire from Employe where Ville='CASA')
```

4. Les sous requêtes

- Opérateurs utilisables:

Opérateur	Description
IN	Teste si un élément est présent dans les lignes du résultat d'une sous-requête.
NOT IN	Teste si un élément n'est pas présent dans les lignes du résultat d'une sous-requête
ANY	Teste si une ou plusieurs lignes du résultat d'une sous-requête répondent à la condition spécifiée
ALL	Teste si toutes les lignes du résultat d'une sous-requête répondent à la condition spécifiée.

4. Les sous requêtes:

- **Exemples:**

```
select *
from Employe
where Salaire not in (Select distinct Salaire from Employe where Ville='CASA')
```

```
select *
from Employe
where Salaire > ALL(Select distinct Salaire from Employe where Ville='CASA')
```

```
select *
from Employe
where Salaire < ANY(Select distinct Salaire from Employe where Ville='CASA')
```

5. Les opérateurs ensemblistes

- L'opérateur Ensembliste est utilisé pour combiner le jeu de résultats de deux ou plusieurs instructions SELECT.
  - Les instructions SELECT doivent avoir le même nombre de colonnes.
  - Les colonnes doivent également avoir des types de données similaires.
  - Les colonnes de chaque instruction SELECT doivent également être dans le même ordre.
- Les opérateurs ensemblistes sont les suivants :
  - l'union (**UNION et UNION ALL**)
  - l'intersection (**INTERSECT**) (Non pris en charge par **MYSQL**)
  - la différence (**MINUS**) (Non pris en charge par **MYSQL**)

5. Les opérateurs ensemblistes

- **Exemples:**

```
Select Ville from Employe
union
Select Ville_Service from Service
```

```
Select Ville from Employe
union all
Select Ville_Service from Service
```

```
select DISTINCT Ville from employe
where ville in
(select DISTINCT ville_service from Service);
```

```
select DISTINCT Ville from employe
where ville not in
(select DISTINCT ville_service from Service);
```

FIN