Memento SQL

Jérôme DESMOULINS
18 septembre 2007

Table des matières

Commandes simples

UPDATE

```
UPDATE table_name SET col_name='xxx' WHERE condition;
```

WHERE

```
WHERE col_name1 = 'xxx' AND col_name2 = 'xxx';

WHERE col_name1 = 'xxx' OR col_name2 = 'xxx';
```

SELECT

```
SELECT [DISTINCT | ALL] * FROM table_name [ WHERE condition ] [
GROUP BY expr [, expr] ... ] [ HAVING condition ] [ {UNION |
UNION ALL | INTERSECT | MINUS} SELECT command ] [ ORDER BY
{expr|position} [ASC | DESC] [, {expr | position} [ASC | DESC]]
...] [ FOR UPDATE OF column ];
```

INSERT

La commande INSERT permet d'insérer une ligne dans une table en spécifiant les valeurs à insérer.

```
      INSERT INTO table_name(nom_col1, nom_col2, ...) VALUES
(val1, val2...);
```

La liste des noms de colonne est optionnelle. Si elle est omise, la liste des colonnes sera par défaut la liste de l'ensemble des colonnes de la table dans l'ordre de la création de la table.
Si une liste de colonnes est spécifiée, les colonnes ne figurant pas dans la liste auront la valeur NULL.

```
INSERT INTO nom_table (nom_col1, nom_col2, ...) SELECT ...
```

Il est possible d'insérer dans une table des lignes provenant d'une autre table.
Le SELECT peut contenir n'importe quelle clause sauf un ORDER BY qui impliquerait un classement des lignes contraire à l'esprit du relationnel.

DELETE

Supprime un ou plusieurs éléments dans une relation.

```
DELETE [ <element > ] FROM < nom_de_relation > [WHERE ... ]
```

Jointure

```
SELECT * FROM table_name1, tableName2 WHERE table_name1.id =
table_name2.id;
```

Union - R1 U R2

```
SELECT * FROM t1 UNION SELECT * FROM t2;
```

Différence - R1 / R2

```
SELECT * FROM R1 WHERE not exists (SELECT * FROM R2 WHERE R2.C
= R1.A and R2.D = R1.B )
```

Requète imbriquées

Requête imbriquées - IN / NOT IN

```
SELECT col_name FROM table_name WHERE col_name IN ('XXX1',
'XXX2');
```

Requêtes imbriquées - ALL

Compare chacune des valeurs de l'ensemble à une valeur particulière et retourne "VRAI" si la comparaison est évaluée pour chacun des éléments.
Les comparateurs sont: <, <=, >, >=, =, !=

```
SELECT Nom FROM PERSONNEL WHERE Salaire => ALL (SELECT Salaire
FROM PERSONNEL );
```

Requête imbriquées - ANY

Compare chacune des valeurs de l'ensemble à une valeur particulière et retourne "VRAI" si la comparaison est évaluée à "VRAI" pour au moins un des éléments.
Les comparateurs sont: <, <=, >, >=, =, !=

```
SELECT Nom FROM PERSONNEL WHERE Nom = ANY (SELECT Nom FROM
ABONNE );
```

Requête imbriquées - EXISTS

Retourne "VRAI" si une requête imbriquée retourne au moins une ligne.

```
SELECT NumAbo FROM ABONNE WHERE NOT EXISTS (PRET);
```

Prédicats

Prédicat BETWEEN

Teste l'appartenance d'une valeur à un intervalle.
Ex : Nom et prénom des salairés qui gagne entre 10000 et 12000 f.

```
SELECT Nom, Prenom FROM PERSONNEL WHERE Salaire BETWEEN 10000
and 12000;
```

Prédicat LIKE

Permet de faire une recherche approximative.
Ex : Nom des abonnés qui habitent en Isère.

```
SELECT Nom FROM ABONNE WHERE CodeP LIKE '38---' OR Ville LIKE
'%ISERE%'
```

Prédicat IS NULL

Permet de tester si un champ a été affecté.
Ex : Liste des abonnés qui n'ont pas le téléphone (ou qui sont sur liste rouge).

```
SELECT Nom FROM ABONNE WHERE Telephone IS NULL
```

Clauses

Clause GROUP BY

Application de fonction agégats à des collections d'enregistrements reliées sémentiquement.
Ex : Nombre d'abonné dans chaque ville.

```
SELECT Ville, count(*) FROM ABONNE GROUP BY Ville
```

Clause HAVING

Cette clause ne s'emploie qu'avec un "GROUP BY".
Exprime une condition sur le groupe d'enregistrement associé à chaque valeur du groupage.
Ex : Nombre de prets effectués avant le 22 mai par abonné.

```
SELECT NumAbo, count(*) FROM PRET GROUP BY NumAbo HAVING
DatePret <= '22/05/00'
```

Clause ORDER BY

Permet l'ordonnancement du résultat avant l'affichage.
Ex : Liste des salaires annuels classés par ordre décroissant.

```
SELECT Salaire * 12 FROM PERSONNEL ORDER BY Salaire
```

Clause DISTINCT

Elimine les doublons avant d'utiliser une fonction agrégat.
Ex : Liste de toutes les villes où habite au moins un abonné.

```
SELECT DISTINCT Ville FROM ABONNE
```

Fonctions agrégats

Ces fonctions ne peuvent être utilisées que dans une clause SELECT ou dans une clause HAVING .
On peut préfixer expr par les mots clés [DISTINCT | ALL].

Agrégat - COUNT

Dénombre les lignes sélectionnées.

```
COUNT ( expr )
```

Ex : Nombre de salariés.

```
SELECT COUNT( * ) FROM PERSONNEL
```

Agrégat - SUM

Additionne les valeurs de type numérique.

```
SUM ( expr )
```

Ex : Somme des salaires des employé dont le prénom est "Pierre".

```
SELECT SUM( Salaire ) FROM PERSONNEL WHERE Prenom='Pierre'
```

Agrégat - MIN

Retourne la valeur minimale d'une colonne de type caractère ou numérique.

```
 MIN ( expr )
```

Ex : Abonné le plus ancien (plus petit numéro d'abonné).

```
SELECT MIN( NumAbo ) FROM ABONNE
```

Agrégat - MAX

Retourne la valeur maximale d'une colonne de type caractère ou numérique.

```
MAX ( expr )
```

Ex : Plus gros salaire parmis les employés Grenoblois.

```
SELECT MAX( Salaire ) FROM PERSONNEL WHERE Adresse LIKE
'%Grenoble%'
```

Agrégat - AVG

Calcule la moyenne d'une colonne de type numérique.

```
AVG ( expr )
```

Ex : Moyenne des salaires des bibliothécaires.

```
SELECT AVG( Salaire ) FROM PERSONNEL WHERE
Fonction='Bibliothécaire'
```

Commandes spécifiques

Oracle

Obtenir des informations

Définition d'une table

Obtenir la définition d'une table:

```
desc ma_table;
```

Tables importantes

Liste des tables

Liste de toutes les tables:

```
select * from all_tables;
```

Liste de toutes les tables de l'utilisateur avec lequel on est connecté:

```
select * from user_tables;
```

alternative:

```
select * from cat;
```

Liste des objets

Liste de tous les objets:

```
select * from all_objects;
```

Liste de tous les objets de l'utilisateur avec lequel on est connecté:

```
select * from user_objects;
```

alternative:

```
select * from obj;
```

Liste des privilèges

Obtenir les privilèges système de l'utilisateur:

```
select * from user_sys_privs;
```

Obtenir les privilèges sur tous les objets accessibles:

```
select * from user_tab_privs;
```