Secteur Tertiaire Informatique

Filière « Etude et développement »

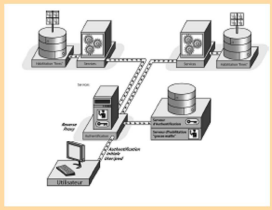
Séquence « Développer des composants d'accès aux données »

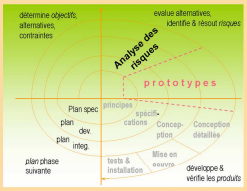
SQL Server et Transact SQL
Le Langage DML partie 1
(insert, update, delete)

Apprentissage

Mise en pratique

Evaluation





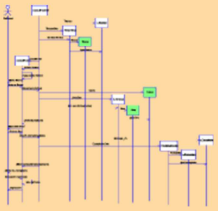


TABLE DES MATIERES

Objectifs

Après étude de ce document, le stagiaire sera capable de traduire en langage SQL un besoin de mise à jour de données en tables de bases de données relationnelle.

Pré requis

Connaissance de base des SGBD relationnelles comme les tables, attributs ou champs, domaine et types de données.

Outils de développement

SQL Server Management Studio (version 2008 minimum).

Méthodologie

SQL Server permet d'effectuer la mise à jour des données soit directement à travers SQL Server Management Studio (menu contextuel, 'Modifier les 200 lignes du haut'), soit à travers des ordres de manipulation de données SQL spécifiques (LMD SQL).

Ce document décrit et illustre les principaux usages des instructions SQL de mise à jour des données en tables.

Pour modifier les données d'une base, l'utilisateur dispose de 3 ordres SQL :

- INSERT
- UPDATE
- DELETE

Le développeur met au point et teste ses requêtes SQL à travers une fenêtre de requête dans SQL Server Management Studio.

Mode d'emploi

Symboles utilisés :

 Renvoie à des supports de cours, des livres ou à la documentation en ligne constructeur.

 Propose des exercices ou des mises en situation pratiques.

 Point important qui mérite d'être souligné !

Ressources

L’aide en ligne Microsoft de référence accessible facilement par la touche F1 dans SQL Server Management Studio.

Lectures conseillées

1. L'ORDRE INSERT

L'ajout de lignes dans une table (ou une vue) répond à la syntaxe suivante :

```
INSERT [INTO] <NOM DE TABLE> [(<NOMS DE COLONNES>)]
{ DEFAULT VALUES
| VALUES     (<LISTE DE VALEURS>)
| < REQUETE SELECT >
| EXECUTE <NOM DE PROCEDURE STOCKEE>}
```

Les modes principaux disponibles dans l'ordre INSERT pour ajouter des lignes sont :

- **l'insertion directe avec la clause VALUES**

L'attribution de valeurs est faite aux colonnes.

Exemple 1: Insérer l'employé 00140, de nom REEVES, de prénom HUBERT dans le département A00, de salaire 2100€ dans la table EMPLOYES de structure EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)

```
INSERT INTO EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)
    VALUES (00140,'REEVES','HUBERT','A00',2100)
```

On donne une valeur pour chacun des attributs spécifiés dans l'ordre INSERT ; les valeurs de la clause VALUES doivent correspondre avec la liste des colonnes, les attributs non spécifiés prennent la valeur NULL.

Exemple 2: Insérer l'employé 00140, de nom REEVES, de prénom HUBERT dans le département 'A00'

```
INSERT INTO EMPLOYES (NOEMP, NOM, PRENOM, DEPT)
    VALUES (00140,'REEVES','HUBERT','A00')
```

La colonne Salaire prendra la valeur NULL pour cette ligne.

Si cette colonne n'a pas été spécifiée comme pouvant être nulle, une erreur sera générée.

La liste des colonnes peut être omise à condition que l'ordre d'insertion concerne toutes les colonnes de la table.

Exemple 3: Insérer l'employé 00140, de nom REEVES, de prénom HUBERT dans le département A00, de salaire 2100€

```
INSERT INTO EMPLOYES
    VALUES (00140,'REEVES','HUBERT','A00', 2100)
```

- **l'insertion à l'aide de sous requêtes en utilisant SELECT**

Les données à insérer sont issues d'une table.

Exemple 4: Insérer dans la table EMPLOYES_A00 préalablement créée de structure EMPLOYES_A00 (NOEMP, NOM, PRENOM, SALAIRE)

tous les employés de la table EMPLOYES attachés au département A00.

```
INSERT INTO EMPLOYES_A00 (NOEMP, NOM, PRENOM, SALAIRE)
    SELECT NOEMP, NOM, PRENOM, SALAIRE
    FROM EMPLOYES
    WHERE WDEPT = 'A00'
```

ou

```
INSERT INTO EMPLOYES_A00
    SELECT * FROM EMPLOYES
    WHERE WDEPT = 'A00'
```

- **l'insertion à l'aide d'une procédure stockée**

L'insertion est déléguée à la procédure stockée appelée.

Exemple 5: Insérer dans la table EMPLOYES_A00 préalablement créée de structure EMPLOYES_A00 (NOEMP, NOM, PRENOM, SALAIRE)

tous les employés de la table EMPLOYES attachés au département A00.

```
INSERT INTO EMPLOYES_A00 (NOEMP, NOM, PRENOM, SALAIRE)
    EXECUTE EMP_A00
```

Les employés concernés sont sélectionnés dans la procédure stockée EMP_A00

Une colonne ayant une propriété IDENTITY ne fait pas partie de la liste des colonnes :

On peut toutefois forcer sa valeur en utilisant l'instruction

SET IDENTITY_INSERT *nom_table*{ ON | OFF }

On désactivera la propriété IDENTITY, on insérera la ligne et on réactivera la propriété IDENTITY.

La clause DEFAULT VALUES force la nouvelle ligne à prendre les valeurs par défaut définies pour chaque colonne

(Voir l'aide en ligne SQL Server pour plus d'informations).

2. L'ORDRE UPDATE

L'ordre UPDATE est utilisé pour modifier des lignes existantes en tables et est composé de trois clauses :

```
UPDATE <NOM DE TABLE>
SET <NOM COLONNE 1> = <VALEUR 1> [,… <NOM COLONNE n> = <VALEUR n>]
FROM <NOM DE TABLE 1>[, … <NOM DE TABLE n>]
WHERE <PREDICAT>
```

- **SET** Nom des colonnes et leurs valeurs ou expressions mises à jour.
- **FROM** Nom d'autres tables utilisées pour fournir des critères
- **WHERE** Critère de sélection pour la mise à jour d'une ligne (optionnel)
  Si la clause WHERE n'est pas codée, la table entière sera mise à jour.

Exemple 1: Augmenter le salaire de 20% de tous les employés pour la table EMPLOYES de structure
EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)

```
UPDATE EMPLOYES
    SET SALAIRE = SALAIRE * 1,2
```

Exemple 2: Augmenter le salaire de 20% de l'employé de matricule 00040.

```
UPDATE EMPLOYES
    SET SALAIRE = SALAIRE * 1,2
    WHERE NOEMP = 00040
```

Exemple 3: Modifier le salaire (augmentation de 20%) de l'employé de matricule 00040, et son affectation dans le service A40

```
UPDATE EMPLOYES
    SET SALAIRE = SALAIRE * 1,2, DEPT= 'A40'
    WHERE NOEMP = 00040
```

On pourra utiliser l'ordre UPDATE combiné à une sous-requête.

Exemple 4: Augmenter le salaire de 20% des employés du service informatique (repéré par son nom dans la table DEPART), pour les tables EMPLOYES et DEPART de structure
EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)
DEPART (NODEPT, NOMDEPT)

```
UPDATE EMPLOYES
    SET SALAIRE = SALAIRE * 1,2
    WHERE DEPT IN (SELECT NODEPT FROM DEPART
                        WHERE NOMDEPT LIKE 'SERVICE%')
```

On pourra également utiliser l'ordre UPDATE et sa clause FROM pour utiliser des informations provenant d'une autre table.

Exemple 5: Augmenter le salaire de 20% des employés du service informatique (repéré par son nom dans la table DEPART), pour les tables EMPLOYES et DEPART de structure
EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)
DEPART (NODEPT, NOMDEPT)

```
UPDATE EMPLOYES
    SET SALAIRE = SALAIRE * 1, 2
    FROM EMPLOYES
    JOIN DEPART
    ON DEPT = NODEPT
    WHERE NOMDEPT LIKE 'SERVICE%'
```

3. L'ORDRE DELETE

L'ordre DELETE utilise trois clauses pour supprimer une ou plusieurs lignes d'une table.

```
DELETE [FROM]  <NOM DE TABLE>
FROM  <NOM DE TABLE> [, … <NOM DE TABLE n>]
WHERE <PREDICAT>
```

- **FROM** Spécifie le nom de la table où les lignes seront supprimées
- **FROM** Une 2eme clause FROM pour spécifier le nom d'autres tables utilisées pour fournir des critères
- **WHERE** Spécifie le critère de sélection (optionnel)
  Si la clause WHERE n'est pas codée, toutes les lignes seront supprimées.

Exemple 1: Supprimer tous les employés de la table EMPLOYES de structure
EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)

```
DELETE FROM EMPLOYES
```

Exemple 2: Supprimer les employés du département 'E21'

```
DELETE FROM EMPLOYES
    WHERE WDEPT ='E21'
```

On pourra utiliser l'ordre DELETE combiné à une sous requête.

Exemple 3: Supprimer tous les employés du service informatique (repéré par son nom dans la table DEPART), pour les tables EMPLOYES et DEPART de structure
EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)
DEPART (NODEPT, NOMDEPT)

```
DELETE FROM EMPLOYES
    WHERE DEPT IN (SELECT NODEPT FROM DEPART
                        WHERE NOMDEPT LIKE 'SERVICE%')
```

On pourra de la même façon utiliser l'ordre DELETE et sa deuxième clause FROM pour utiliser des informations provenant d'une autre table.

Exemple 5: Supprimer tous les employés du service informatique pour les tables EMPLOYES et DEPART de structure
EMPLOYES (NOEMP, NOM, PRENOM, DEPT, SALAIRE)
DEPART (NODEPT, NOMDEPT)

```
DELETE FROM EMPLOYES
    FROM EMPLOYES
    JOIN DEPART
    ON DEPT = NODEPT
    WHERE NOMDEPT LIKE 'SERVICE%'
```

CREDITS

ŒUVRE COLLECTIVE DE l'AFPA
Sous le pilotage de la DIIP et du centre d'ingénierie sectoriel Tertiaire-Services

Equipe de conception (IF, formateur, mediatiseur)
E. Cattanéo - Formatrice
B.Hézard – Formateur (mise en forme)
Chantal Perrachon – Ingénieure de formation

Date de mise à jour : 23/09/15