SQL : Langage de Manipulation de Données

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

elmouelhi.achref@gmail.com

Plan

MySQL

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table VALUES (valeur_colonne1, ..., valeur_colonneN);
```

MySQL

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table VALUES (valeur_colonne1, ..., valeur_colonneN);
```

Le SGBD affectera les valeurs aux colonnes dans l'ordre.

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

`id` : clé primaire de la table

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

`id` : clé primaire de la table

Insérer une valeur pour chaque colonne

```
INSERT INTO enseignants VALUES (5,'Cooper','David',3000,'Marseille');
```

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

`id` : clé primaire de la table

Insérer une valeur pour chaque colonne

```
INSERT INTO enseignants VALUES (5,'Cooper','David',3000,'Marseille');
```


Nouveau contenu

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
5	Cooper	David	3000	Marseille

Et si on essayait d'insérer une clé qui existe déjà dans la table ?

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille');
```

Et si on essayait d'insérer une clé qui existe déjà dans la table ?

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille');
```

ça provoquerait une violation de clé unique `PRIMARY KEY`.

Et si on essayait d'insérer une clé qui existe déjà dans la table ?

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille');
```

ça provoquerait une violation de clé unique `PRIMARY KEY`.

Une solution possible consiste à utiliser `ON DUPLICATE KEY UPDATE`

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille')
ON DUPLICATE KEY UPDATE nom = VALUES(nom), prenom = VALUES(prenom);
```

Et si on essayait d'insérer une clé qui existe déjà dans la table ?

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille');
```

ça provoquerait une violation de clé unique `PRIMARY KEY`.

Une solution possible consiste à utiliser `ON DUPLICATE KEY UPDATE`

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille')
ON DUPLICATE KEY UPDATE nom = VALUES(nom), prenom = VALUES(prenom);
```

Ou aussi

```
INSERT INTO enseignants
VALUES (5, "EL Mouelhi", "Achref", 3000, 'Marseille')
ON DUPLICATE KEY UPDATE nom = "EL Mouelhi", prenom = "Achref";
```

MySQL

Remarques

- Cette clause est pratique mais peut avoir un coût de performance si elle est utilisée sur de grandes tables.
- On peut utiliser `VALUES()` jusqu'à **MySQL 8.0.19**, remplacé ensuite par `INSERT INTO ... ON DUPLICATE KEY UPDATE col = new_val`.

MySQL

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table SET
nom_colonne1 = valeur_colonne1,
...
nom_colonneN = valeur_colonneN;
```

MySQL

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table SET
nom_colonne1 = valeur_colonne1,
...
nom_colonneN = valeur_colonneN;
```


Le SGBD affectera les valeurs aux colonnes selon le nom.

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

Insérer une valeur pour chaque colonne

```
INSERT INTO enseignants SET
id=5,
nom='Cooper',
prenom='David',
salaire=3000,
ville='Marseille';
```

Considérons la table suivante

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris

Insérer une valeur pour chaque colonne

```
INSERT INTO enseignants SET
id=5,
nom='Cooper',
prenom='David',
salaire=3000,
ville='Marseille';
```

Résultat

enseignants				
id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
5	Cooper	David	3000	Marseille

Insérer quelques valeurs (pas toutes)

```
INSERT INTO nom_table (valeur_colonneP, ..., valeur_colonneQ)
VALUES (valeur_colonneP,...,valeur_colonneQ);
```

Insérer quelques valeurs (pas toutes)

```
INSERT INTO nom_table (valeur_colonneP, ..., valeur_colonneQ)
VALUES (valeur_colonneP,...,valeur_colonneQ);
```

Exemple

```
INSERT INTO enseignants (id, nom, prenom)
VALUES (6, 'Benatia', 'Sonia');
```

Insérer quelques valeurs (pas toutes)

```
INSERT INTO nom_table (valeur_colonneP, ..., valeur_colonneQ)
VALUES (valeur_colonneP,...,valeur_colonneQ);
```

Exemple

```
INSERT INTO enseignants (id, nom, prenom)
VALUES (6, 'Benatia', 'Sonia');
```


id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
5	Cooper	David	3000	Marseille
6	Benatia	Sonia		

Insérer quelques valeurs (pas toutes)

```
INSERT INTO nom_table (valeur_colonneP, ..., valeur_colonneQ)
VALUES (valeur_colonneP,...,valeur_colonneQ);
```

Exemple

```
INSERT INTO enseignants (id, nom, prenom)
VALUES (6, 'Benatia', 'Sonia');
```


id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
5	Cooper	David	3000	Marseille
6	Benatia	Sonia		

Ceci n'est possible que si les champs non-renseignés ne sont pas nuls

MySQL

Considérons une table avec UUID v7 (`BINARY(16)`)

```
CREATE TABLE utilisateurs (
    id BINARY(16) PRIMARY KEY,
    nom VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    date_creation TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

MySQL

Considérons une table avec UUID v7 (`BINARY(16)`)

```
CREATE TABLE utilisateurs (
    id BINARY(16) PRIMARY KEY,
    nom VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    date_creation TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Exemple d'insertion avec UUID v7

```
INSERT INTO utilisateurs (id, nom, email)
VALUES (UUID_TO_BIN(UUID(), 1), 'John Doe', 'john@doe.fr');
```

MySQL

Considérons une table avec UUID v7 (`BINARY(16)`)

```
CREATE TABLE utilisateurs (
    id BINARY(16) PRIMARY KEY,
    nom VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    date_creation TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Exemple d'insertion avec UUID v7

```
INSERT INTO utilisateurs (id, nom, email)
VALUES (UUID_TO_BIN(UUID(), 1), 'John Doe', 'john@doe.fr');
```

L'argument 1 de `UUID_TO_BIN()` pour un tri optimisé.

MySQL

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table VALUES
    (valeur1_colonne1, ..., valeur1_colonneN),
    (valeur2_colonne1, ..., valeur2_colonneN),
    ...
    (valeurQ_colonne1, ..., valeurQ_colonneN);
```

MySQL

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table VALUES
    (valeur1_colonne1, ..., valeur1_colonneN),
    (valeur2_colonne1, ..., valeur2_colonneN),
    ...
    (valeurQ_colonne1, ..., valeurQ_colonneN);
```

Pour l'exemple précédent

```
INSERT INTO enseignants VALUES
    (1, Durand, Philippe, 2000, Marseille),
    (2, Leberre, Bernard, 1500, Paris),
    (3, Benammar, Pierre, 1800, Lyon),
    (4, Hadad, Karim, 1500, Paris),
    (5, Cooper, David, 3000, Marseille);
```

MySQL

Remarques

- En attribuant explicitement 0 à la colonne auto_incrementée, une valeur incrémentale lui sera attribuée.
- En attribuant explicitement `NULL` à la colonne auto_incrementée, une valeur incrémentale lui sera également attribuée.
- Les champs avec des valeurs auto-incrémentées sont pris en charge par le **SGBD**, mais il est possible de spécifier une valeur de notre choix si elle n'est pas présente dans la table.
- Dans cette situation, si la dernière valeur attribuée est supérieure à la précédente, le **SGBD** incrémente cette valeur et l'assigne au prochain tuple.

MySQL

Les deux requêtes suivantes permettent d'attribuer une valeur incrémentale à la clé primaire

```
INSERT INTO enseignants VALUES (0, 'Tancredi', 'Sarah', 3500, 'NY');
INSERT INTO enseignants VALUES (NULL, 'Mahone', 'Alex', 4000, 'NY');
```

MySQL

Pour insérer des données dans une table destination depuis une table source

```
INSERT INTO table_destination (colonne_1, ... colonne_n)
SELECT colonne_1, ... colonne_n
FROM table_source;
```

MySQL

Pour insérer un tuple avec les valeurs par défaut pour certaines colonnes (salaire et ville)

```
INSERT INTO enseignants VALUES (0, 'Tancredi', 'Sarah', DEFAULT,
   DEFAULT);
```

MySQL

Supprimer des tuples respectant une ou plusieurs conditions

```
DELETE FROM nom_table
[WHERE condtions];
```

MySQL

Supprimer des tuples respectant une ou plusieurs conditions

```
DELETE FROM nom_table
[WHERE condtions];
```

Exemple

```
DELETE FROM enseignants
WHERE salaire > 2000 AND ville = 'Marseille';
```

MySQL

Supprimer des tuples respectant une ou plusieurs conditions

```
DELETE FROM nom_table
[WHERE condtions];
```

Exemple

```
DELETE FROM enseignants
WHERE salaire > 2000 AND ville = 'Marseille';
```


id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
5	Cooper	David	3000	Marseille
6	Benatia	Sonia		

MySQL

Supprimer tous les tuples d'une table

```
DELETE FROM nom_table;
```

MySQL

Supprimer tous les tuples d'une table

```
DELETE FROM nom_table;
```

Remarque

Cette requête supprime toutes les données de la table, mais pas la table. Donc, le résultat est une table vide.

MySQL

Considérons une table avec UUID v7 (`BINARY(16)`)

```
CREATE TABLE utilisateurs (
    id BINARY(16) PRIMARY KEY,
    nom VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    date_creation TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

MySQL

Considérons une table avec UUID v7 (`BINARY(16)`)

```
CREATE TABLE utilisateurs (
    id BINARY(16) PRIMARY KEY,
    nom VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    date_creation TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```


Et le tuple suivant

```
INSERT INTO utilisateurs (id, nom, email)
VALUES (UUID_TO_BIN(UUID(), 1), 'John Doe', 'john@doe.fr');
```

Affichage de UUID v7

```
SELECT id as bin,  BIN_TO_UUID(id) AS uuid
FROM utilisateurs;
```

Affichage de UUID v7

```
SELECT id as bin,  BIN_TO_UUID(id) AS uuid
FROM utilisateurs;
```

Résultat

```
+------------------------------------+--------------------------------------+
| bin                                | uuid                                 |
+------------------------------------+--------------------------------------+
| 0x11EFEB0BE85E2633BCBE046C59C3AAFE | 11efeb0b-e85e-2633-bcbe-046c59c3aafe |
+------------------------------------+--------------------------------------+
```

Affichage de UUID v7

```
SELECT id as bin,  BIN_TO_UUID(id) AS uuid
FROM utilisateurs;
```

Résultat

```
+------------------------------------+--------------------------------------+
| bin                                | uuid                                 |
+------------------------------------+--------------------------------------+
| 0x11EFEB0BE85E2633BCBE046C59C3AAFE | 11efeb0b-e85e-2633-bcbe-046c59c3aafe |
+------------------------------------+--------------------------------------+
```

Exemple de suppression

```
DELETE FROM utilisateurs
WHERE id = UUID_TO_BIN('11efeb0b-e85e-2633-bcbe-046c59c3aafe', 1);
```

Affichage de UUID v7

```
SELECT id as bin,  BIN_TO_UUID(id) AS uuid
FROM utilisateurs;
```

Résultat

```
+------------------------------------+--------------------------------------+
| bin                                | uuid                                 |
+------------------------------------+--------------------------------------+
| 0x11EFEB0BE85E2633BCBE046C59C3AAFE | 11efeb0b-e85e-2633-bcbe-046c59c3aafe |
+------------------------------------+--------------------------------------+
```

Exemple de suppression

```
DELETE FROM utilisateurs
WHERE id = UUID_TO_BIN('11efeb0b-e85e-2633-bcbe-046c59c3aafe', 1);
```

Exemple de suppression

```
DELETE FROM utilisateurs
WHERE id = 0x11EFEB0BE85E2633BCBE046C59C3AAFE;
```

MySQL

Remarques

- `BIN_TO_UUID` : conversion binaire ⇒ format lisible.
- `Littéral hexadécimal (0x...)` : correspondance directe en `BINARY(16)`.
- Toujours utiliser `UUID_TO_BIN` et `BIN_TO_UUID` avec le paramètre 1 pour les **UUID**.
- Préférer `BINARY(16)` à `VARCHAR` pour le stockage d'**UUID** (gain de place et rapidité).
- Bien comprendre la différence entre représentation hexadécimale et binaire.

MySQL

TRUNCATE aussi permet de supprimer toutes les données d'une table

```
TRUNCATE [TABLE] nom_table;
```

MySQL

TRUNCATE aussi permet de supprimer toutes les données d'une table

```
TRUNCATE [TABLE] nom_table;
```


Remarques

- TRUNCATE fonctionne comme un DELETE sans WHERE.
- TRUNCTATE est plus rapide que DELETE car elle supprime puis recrée la table.
- DELETE supprime les tuples un par un.
- TRUNCTATE ne prend pas de clause WHERE.

MySQL

Modifier des tuples respectant une ou plusieurs conditions

```
UPDATE nom_table
SET nom_colonne1 = valeur1
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]
WHERE condition;
```

MySQL

Modifier des tuples respectant une ou plusieurs conditions

```
UPDATE nom_table
SET nom_colonne1 = valeur1
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]
WHERE condition;
```

Exemple

```
UPDATE enseignants
SET salaire= 1600, ville  = 'Toulouse'
WHERE nom = 'benatia';
```

MySQL

Modifier des tuples respectant une ou plusieurs conditions

```
UPDATE nom_table
SET nom_colonne1 = valeur1
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]
WHERE condition;
```

Exemple

```
UPDATE enseignants
SET salaire= 1600, ville  = 'Toulouse'
WHERE nom = 'benatia';
```


id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
6	Benatia	Sonia	**1600**	**Toulouse**

MySQL

Modifier tous les tuples

```
UPDATE enseignants
SET ville  = 'Marseille' ;
```

MySQL

Modifier tous les tuples

```
UPDATE enseignants
SET ville  = 'Marseille' ;
```

Le résultat

id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	**Marseille**
2	Leberre	Bernard	1500	**Marseille**
3	Benammar	Pierre	1800	**Marseille**
4	Hadad	Karim	1500	**Marseille**
6	Benatia	Sonia	1600	**Marseille**

Modifier un tuple respectant une ou plusieurs conditions

```
REPLACE INTO nom_table
SET nom_colonne1 = valeur1
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]
```

Modifier un tuple respectant une ou plusieurs conditions

```
REPLACE INTO nom_table
SET nom_colonne1 = valeur1
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]
```

Exemple

```
REPLACE INTO enseignants
SET
salaire=1600,
ville='Toulouse',
id=6;
```

Modifier un tuple respectant une ou plusieurs conditions

```
REPLACE INTO nom_table
SET nom_colonne1 = valeur1
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]
```

Exemple

```
REPLACE INTO enseignants
SET
salaire=1600,
ville='Toulouse',
id=6;
```


id	nom	prenom	salaire	ville
1	Durand	Philippe	2000	Marseille
2	Leberre	Bernard	1500	Paris
3	Benammar	Pierre	1800	Lyon
4	Hadad	Karim	1500	Paris
6	**NULL**	**NULL**	**1600**	**Toulouse**

MySQL

Remarques

- Si la clé est présente, alors `Replace` effectue une modification.
- Si la clé est absente, alors `Replace` effectue une insertion.
- Dans tous les cas, toutes les colonnes non-spécifiées auront la valeur `NULL`.

MySQL

Question

Peut-on modifier la valeur d'une colonne clé primaire ?

MySQL

Question

Peut-on modifier la valeur d'une colonne clé primaire ?

Réponse

- Une clé primaire est par défaut unique et non nulle.
- Si elle n'est référencée par aucune clé étrangère (`FOREIGN KEY`), elle peut être modifiée directement.
- Si elle est référencée, il faut soit :
  - supprimer la contrainte référentielle (`FOREIGN KEY`), modifier la clé, puis la rétablir.
  - utiliser une mise à jour en cascade (`ON UPDATE CASCADE`).

MySQL

Pourquoi est-il déconseillé de modifier une clé primaire ?

- **Perte de cohérence** : risque de casser les relations si la modification n'est pas propagée.
- **Perte de sens sémantique** : une clé primaire représente souvent une identité stable.
- **Problèmes de performances** : surtout sur des tables volumineuses avec de nombreuses références.

MySQL

Pour importer une base de données à partir d'un script SQL (Depuis une console MySQL)

```
SOURCE nom_fichier.sql;
```

MySQL

Pour importer une base de données à partir d'un script SQL (Depuis une console MySQL)

```
SOURCE nom_fichier.sql;
```

ou

```
\. nom_fichier.sql;
```

MySQL

Pour importer une base de données à partir d'un script SQL (Depuis une console Windows ou Linux)

```
mysql -u nom_utilisateur -p nom_bd < nom_fichier.sql
```

MySQL

Pour importer une base de données à partir d'un script SQL (Depuis une console Windows ou Linux)

```
mysql -u nom_utilisateur -p nom_bd < nom_fichier.sql
```

Avant cela, Il faut ajouter MySQL au `Path` de Windows

```
set path=c:\chemin\vers\mysql\bin
```

MySQL

Pour exporter une base de données dans un fichier SQL

```
mysqldump -u nom_utilisateur -p nom_bd -r nom_fichier.sql
```

MySQL

Pour exporter une base de données dans un fichier SQL

```
mysqldump -u nom_utilisateur -p nom_bd -r nom_fichier.sql
```

Avant cela, Il faut ajouter MySQL au `Path` de Windows

```
set path=c:\chemin\vers\mysql\bin
```

MySQL

Options utiles avec `mysqldump`

- `--single-transaction` : pour éviter les blocages (utile avec **InnoDB**).
- `--routines` : pour inclure les procédures stockées.
- `--no-data` : n'exporte que la structure (pas les données).
- `--no-create-info` : n'exporte que les données (pas la structure).
- ...