

420-C42

Langages d'exploitation des bases de données

Partie 7

DML

DML

- Le DML (*Data Manipulation Language*) permet la manipulation des données.
- Les clauses principales sont :
 - INSERT insertion des données
 - UPDATE mise à jour des données
 - DELETE suppression des données

DML

INSERT

- La clause INSERT permet l'insertion de données dans une table.
- Le synopsis simplifié de cette clause est :

```
INSERT INTO nom_table[(colonne [...])]
    { VALUES ({ expr | DEFAULT } [, ...]) [( ... )] | SELECT ... };
```

- Il est possible de spécifier ou non les colonnes où les données sont insérées.
- Il est possible d'insérer plusieurs tuples à la fois à partir de données spécifiées ou d'une requête SELECT.
- Attention aux colonnes à incrément automatique (SERIAL).

DML

INSERT

```
CREATE TABLE employe (  
    id                SERIAL                PRIMARY KEY,  
    nom               VARCHAR(32)          NOT NULL,  
    date_embauche     DATE                 NOT NULL DEFAULT CURRENT_DATE,  
    salaire           NUMERIC(5, 2)  
);
```

-- Insertion simple.

```
INSERT INTO employe VALUES (1000, 'Lebel', '2000-01-01', 20.00);
```

-- Insertion avec identification des colonnes

```
INSERT INTO employe(id, nom, date_embauche, salaire)  
    VALUES (1001, 'Miron', '2000-01-02', 25.00);
```

-- Encore, attention aux champs à incrément automatique!

DML

INSERT

-- Insertion avec utilisation explicite des valeurs par défaut

```
INSERT INTO employe(id, nom, date_embauche, salaire)
VALUES (DEFAULT, 'Labonté', DEFAULT, 25.00);
```

-- Insertion avec utilisation implicite des valeurs par défaut

```
INSERT INTO employe(nom, salaire) -- < certaines colonnes sont omises
VALUES ('Labonté', 25.00);
```

-- Multiple insertions (on remarque l'ordre des colonnes est inversé)

```
INSERT INTO employe(salaire, nom)
VALUES (25.00, 'Laroche'),
       (DEFAULT, 'Gravel'),
       (35.00, 'Lapierre');
```

-- Multiple insertions à l'aide d'une requête

```
INSERT INTO employe(nom, salaire)
SELECT nom, 25.00 FROM employe WHERE nom LIKE '%a%';
```

DML

UPDATE

- La clause UPDATE permet la modification (mise à jour, édition) de données existantes dans une table.
- Le synopsis simplifié de cette clause est :

UPDATE *nom_table*

SET *colonne* = { *expr* | DEFAULT } [, ...]

[WHERE *condition*];

- Attention, il est possible de modifier toutes les valeurs d'une colonne avec une seule requête.
- Le paramètre WHERE permet de préciser *n* ligne(s) à la fois.

DML

UPDATE

-- augmentation de 5% à tous les employés

```
UPDATE employe  
  SET salaire = salaire * 1.05;
```

-- mettre le salaire de l'employé ayant l'id 1 à 50.0

```
UPDATE employe  
  SET salaire = 50.00  
  WHERE id = 1;
```

-- mettre en majuscule tous les noms et une augmentation de 10\$

-- des employés ayant la lettre 'a' dans leur nom

```
UPDATE employe  
  SET    nom = UPPER(nom),  
        salaire = salaire + 5.00  
  WHERE nom LIKE '%a%';
```


DML

DELETE

- La clause DELETE permet la suppression de données existantes dans une table. La structure de la table reste inchangée.
- Le synopsis simplifié de cette clause est :

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

- Attention, il est possible d'effacer toutes les données de la table avec une seule requête.
- Le paramètre WHERE permet de préciser *n* ligne(s) à la fois.

DML

DELETE

-- supprime tous les employés ayant la lettre 'a' dans leur nom

```
DELETE FROM employe  
WHERE nom LIKE '%a%';
```

-- supprime toutes les lignes de la table

-- la table existe toujours mais elle est vide

```
DELETE FROM employe;
```

DML

dépendances circulaires

- Les dépendances circulaires amènent une difficulté autant pour la création des tables (DDL) que lors de la manipulation des données (DML).
 - la cause : les clés étrangères
 - exemple de suppression et solutions
 - exemple de mise à jour et solutions
 - exemple d'insertion ... solutions peu intéressantes
- Nous verrons plus loin qu'il est possible de gérer ces situations adéquatement mais pour l'instant, utilisons la possibilité de désactiver et réactiver les contraintes de clés étrangères :

```
ALTER TABLE nom_table DISABLE TRIGGER ALL; -- désactive tous les
...                                           -- déclencheurs incluant les
ALTER TABLE nom_table ENABLE TRIGGER ALL;  -- contraintes de clé étrangère
```