420-C42

Langages d'exploitation des bases de données

Partie 15

DML II

Mise à jour par jointure, DML avec ETC, importation et exportation de données

DML UPDATE … FROM

- L'instruction UPDATE ... FROM étend les capacités de mise à jour en permettant d'utiliser les données d'autres tables.
- Cette approche effectue une jointure entre la table cible et les tables source définies dans la clause FROM.
- La syntaxe est restreinte à l'utilisation de la syntaxe fondamentale du produit cartésien où :
  - la clause UPDATE indique la table cible
  - la clause FROM permet d'énumérer la ou les tables sources (séparées par des virgules)
  - la clause WHERE permet :
    - de définir formellement les conditions de jointures (revoir la notion de produit cartésien)
    - de définir les critères de filtrage supplémentaires
  - les nombreuses syntaxes utilisant … JOIN … ON … ne sont pas disponibles
  - toutes les tables impliquées font partie de la jointure (clauses UPDATE et FROM)

DML UPDATE … FROM

- Le synopsis devient ainsi :

UPDATE *nom_table_cible*
SET *colonne* = { *expr* | DEFAULT | *table_source.colonne* } [, …]
FROM *nom_table_source1* [*, nom_table_source2, ...*]
WHERE *join_condition* [*selection_condition*];

DML UPDATE … FROM

- Exemple 1 – Les employés déménagent dans la même ville que le département où ils travaillent

```
UPDATE employee
        SET city = department.city
        FROM department
        WHERE employee.dept_id = department. id;
```

- Exemple 2 – Les employés sont assigné au même département que celui pour lequel travaille leur superviseur

```
UPDATE employee AS emp
        SET dept_id = department.id
        FROM employee AS sup, department AS dep
        WHERE emp.supervisor_id = sup.id AND sup.dept_id = department.id
            AND emp.supervisor_id IS NOT NULL AND sup.dept_id IS NOT NULL;
```

DML

DML & CTE

- La clause WITH peut être utilisée avec les opérations du DML. Cela permet d'écrire des requêtes plus complexes, lisibles et modulaires.
- Cas d’usage courant :
  - INSERT avec WITH:
    - Préparation de données : Transformation ou filtrage des données avant de les insérer dans une table.
    - Génération de données : Utilisation de rand(), generate_series() ou d'autres fonctions utilitaires pour créer des ensembles de données à insérer.

```
-- Prépare et insère une série de nombres générés
WITH new_data AS (
    SELECT generate_series(1, 10) AS num_serie, rand() AS num_alea
)
INSERT INTO ma_table (valeur1, valeur2, valeur3)
    SELECT num_serie, num_alea, num_serie + num_alea FROM new_data;
-- Cet exemple est très pertinent car on voit comment générer un nombre aléatoire et
-- l’utiliser par la suite dans plusieurs colonnes.
```

DML

DML & CTE

- Cas d’usage courant :
  - UPDATE avec WITH:
    - Mises à jour élaborées : Calcule de nouvelles valeurs basées sur des données d'autres tables ou sur des opérations complexes.
    - Mises à jour conditionnelles : Identification les lignes à mettre à jour en fonction de critères plus complexes.

```
-- Met à jour les salaires d'employés spécifiques en utilisant un pré-calcul.
WITH updated_salaries AS (
    SELECT emp_id, salaire * 1.1 AS new_salary
    FROM employés
    WHERE dept_id = 10
)
UPDATE employes
SET salaire = new_salary
FROM updated_salaries
WHERE employés.emp_id = updated_salaries.emp_id;
```

DML

- Cas d’usage courant :
  - DELETE avec WITH:
    - Suppressions élaborées : Identification les lignes à supprimer en fonction de critères plus étayés ou de relations avec d'autres tables.
    - Suppressions en cascade : Suppression des lignes liées dans plusieurs tables.

```
-- Supprime les commandes obsolètes en identifiant d'abord les ID à supprimer
WITH old_orders AS (
    SELECT commande_id
    FROM commandes
    WHERE date_commande < '2023-01-01’
)
DELETE FROM commandes
WHERE commande_id IN (SELECT commande_id FROM old_orders);
```

Importation/Exportation COPY

- La commande COPY est utilisée pour importer ou exporter efficacement des données entre une table PostgreSQL et un fichier de données externe (ou le flux standard).
- Elle est beaucoup plus rapide que des commandes INSERT ou SELECT classiques pour de gros volumes.
- Généralités:
  - COPY permet de transférer des données vers ou depuis une table (ou le résultat d'une requête).
  - Les formats pris en charge sont : texte, CSV et binaire

Importation/Exportation COPY

- La syntaxe de base est :

```
COPY table_name [ (column_list) ] TO|FROM
   'fichier'
   [WITH (options)];
```

- Les options principales sont :
  - FORMAT :
    - TEXT (par défaut : format texte simple, lignes séparées par des retours à la ligne, colonnes par tabulations)
    - CSV (format texte : gère séparateurs, guillemets, etc.)
    - BINARY : Format binaire propriétaire PostgreSQL, plus rapide mais non lisible/manipulable manuellement.

Importation/Exportation COPY

- Les options générales sont :
  - ENCODING '<encodage>' : Détermine l'encodage du fichier (ex : 'UTF8').
  - DELIMITER '<caractère>' (TEXT et CSV) : Définit le séparateur de colonnes. Un seul caractère : par défaut, tabulation pour TEXT et virgule pour CSV
  - NULL '<chaîne>' (TEXT et CSV) : Spécifie la chaîne représentant la valeur nulle.
    - FROM : comment les données seront interprétées à la lecture
    - TO : comment les données seront produites à l'écriture
  - HEADER '<FALSE/TRUE/MATCH>' (TEXT ou CSV : seulement pour FROM) : Détermine si le fichier possède un en-tête.
    - FALSE : lit la première ligne comme étant des données
    - TRUE (valeur par défaut) : ignore la première ligne
    - MATCH : lit la première ligne et s'assure que tous les noms correspondent exactement aux colonnes de sortie dans l'ordre.
  - QUOTE '<caractère>' (CSV) : Caractère utilisé pour entourer les champs textes (par défaut, le guillemet ")

Importation/Exportation COPY

- Exemple :

```
COPY clients FROM '/data/clients.csv'
WITH (
    FORMAT CSV,
    HEADER TRUE,
    DELIMITER ',',
    ENCODING 'UTF8'
);
```

Approche classique d'import/export COPY

- On présente ici une approche classique pour l'importation ou l'exportation des données avec la commande COPY. L'exemple utilise plutôt l'importation mais le principe ce généralise avec l'exportation.
- Lorsqu'on importe des données provenant de fichiers externes, il est souvent essentiel d'appliquer une stratégie en plusieurs étapes afin de faciliter l'adaptation des données. Principalement, on utilise une étape intermédiaire nommée *étape de préparation* (ou *staging* en anglais). Ce processus est celui utilisé dans des processus de migration de données.
- L'objectif est d'utiliser cette étape intermédiaire afin d'effectuer des opérations sur les données (transformation, filtrage et autres).
- Évidemment, si aucune opération n'est requise, ce processus est inutile.

Approche classique d’import/export COPY

- Étapes de l’approche :
  1. Créer une table temporaire (généralement sans contraintes).
  2. Charger les données dans cette table avec COPY.
  3. Analyser, transformer et valider les données.
  4. Insérer ou mettre à jour les tables finales.
  5. Optionnellement, suppression de la table temporaire.
- Il est possible qu’un seul COPY requiert plusieurs opérations DML pour effectuer les opérations requises. Migration de données!

Approche classique d'import/export COPY

- L'objectif est, qu'à l'étape 3, on puisse effectuer toutes les opérations nécessaires à garantir l'intégrité des données. Principalement :
  - Adaptation de types: TEXT → INTEGER, TEXT → DATE, TEXT → BOOLEAN, …
  - Nettoyage des données: Suppression des espaces, homogénéisation de la casse, … (TRIM, UPPER, ...)
  - Filtrage des lignes invalides : Exclusion de lignes indésirables selon le contexte (WHERE …).
  - Validation de format: Contrôle par expressions régulières (emails, codes postaux, identifiants...).
  - Normalisation des valeurs: Uniformisation d'énumération, standardisation utilisant un dictionnaire de valeurs spécifiques, …
  - Génération de valeurs manquantes : Calcul de valeurs par défaut, dates de référence, ...
  - Contrôle de doublons: Détection par DISTINCT, ROW_NUMBER(), ou jointure sur la table cible.
  - Vérification de référentiels : Jointure avec des tables de référence pour s'assurer que les clés étrangères ou catégories sont valides.
  - Création de référentiels : Insertion en plusieurs étapes afin de générer les ID nécessaires aux données référées.
  - …