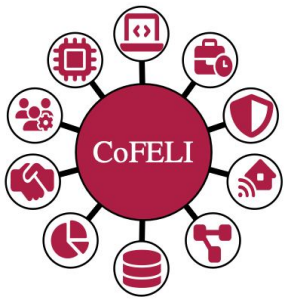

Collectif francophone pour l'enseignement libre de l'informatique

SQL

Rôles et contrôle d'accès

MCED_SQL_12

Luc LAVOIE *(luc.lavoie@usherbrooke.ca)*

—

CoFELI/Scriptorum/SQL_12-Controle-d-acces, version 1.0.1.a, en date du 2025-05-06

— document de travail, ne pas citer —

Sommaire

Ce module présente les mécanismes et les éléments de langages propres à SQL relativement au contrôle d'accès.

Mise en garde

Le présent document est en cours d'élaboration ; en conséquence, il est incomplet et peut contenir des erreurs.

Historique

diffusion	resp.	description
2025-05-06	LL	Ménage.
2023-11-01	LL	Ébauche initiale selon les formats contenus dans /Users/Shared/DGS/CoLOED/AsciiDoc/format.

Table des matières

Introduction

La norme ISO reprend pour l'essentiel le modèle présenté dans le module SGBD_12. Malheureusement plusieurs dialectes s'en écartent, ce qui rend difficilement transportable la mise en oeuvre d'une politique de sécurité.

Les éléments de langages et les dispositifs du standard ISO sont donc présentés dans le but de faciliter la comparaison entre les différents dialectes. Nous documenterons les autres langages en les comparant chacun avec la norme ISO, plutôt qu'entre eux.

1. Le cas PostgreSQL

PostgreSQL a fusion les concepts de USER, GROUP et ROLE.

Un USER est un ROLE doté des privilèges et paramètres de connexion. Un GROUP est un ROLE définis par une liste de ROLE. Les autres ROLE sont des ... ROLE.

1.1. Éléments de langage

1.1.1. ROLE (USER, GROUP)

Définir (indéfinir) un rôle et lui associer des droits d'accès.

- CREATE ROLE nom [[WITH] Option [...]]
- DROP ROLE [IF EXISTS] nom [, ...]
- Option - de type de rôle (au niveau du SGBD)
  - [NO]SUPERUSER
  - [NO]CREATEDB
  - [NO]CREATEROLE
  - [NO]LOGIN
  - [NO]REPLICATION
  - [NO]BYPASSRLS
- Option - de connexion
  - PASSWORD 'motdepasse' | PASSWORD NULL
  - CONNECTION LIMIT limite_connexion
  - VALID UNTIL 'heuredate'
- Option - de comportement
  - [NO]INHERIT
    Quand l'instruction GRANT est utilisé pour définir l'appartenance d'un rôle à un autre, l'instruction GRANT peut utiliser la clause WITH INHERIT pour indiquer si les droits du rôle bénéficiaire doivent être « hérités » par le nouveau membre. Si l'instruction GRANT n'indique pas explicitement le comportement de l'héritage, l'instruction GRANT sera créée avec WITH INHERIT TRUE si le rôle membre est configuré avec INHERIT et avec WITH INHERIT FALSE s'il est configuré avec NOINHERIT.
    Avant PostgreSQL 16, l'instruction GRANT n'acceptait pas WITH INHERIT. Du coup, modifier la propriété au niveau rôle changeait aussi le comportement des droits déjà existants. Ce n'est plus le cas.
  - IN ROLE nom_role [, ...]
    La clause IN ROLE fait que le nouveau rôle se voit ajouté automatiquement comme membre des rôles existants cités.
    Il n'existe pas d'option pour ajouter le nouveau rôle en tant qu'administrateur ; il faut utiliser une

commande GRANT séparée pour cela.

- ROLE nom_role [, ...]
  La clause ROLE fait que les rôles existants cités sont ajoutés automatiquement comme membre du nouveau rôle. Ceci a pour effet de transformer le nouveau rôle en « groupe ».
- ADMIN nom_role [, ...]
  Cette clause est équivalente à la clause ROLE, à la différence que les rôles nommés sont ajoutés au nouveau rôle avec l'option WITH ADMIN OPTION. Cela leur confère le droit de promouvoir à d'autres rôles l'appartenance à celui-ci.

Liste des rôles (psql)

La commande suivante `\du` permet d'obtenir la liste courante de tous les rôles ; elle ne peut être exécutée que depuis une session ouverte par un administrateur du SGBD ayant tous les privilèges, typiquement l'utilisateur `postgres`.

1.1.2. ALTER

Modifier un rôle, en modifier les droits d'accès associés.

Les possibilités sont les suivantes :

- ALTER ROLE nom RENAME TO nouveau_nom
- ALTER ROLE spécification_rôle [WITH] Option [...]
- ALTER ROLE spécification_rôle_complet SET paramètre_configuration { TO | = } { value | DEFAULT }
- ALTER ROLE spécification_rôle_complet SET paramètre_configuration FROM CURRENT
- ALTER ROLE spécification_rôle_complet RESET paramètre_configuration
- ALTER ROLE spécification_rôle_complet RESET ALL

où

spécification_rôle_complet ::= { spécification_rôle | ALL } [IN DATABASE nom_base]

spécification_rôle ::= nom_rôle | CURRENT_ROLE | CURRENT_USER | SESSION_USER

1.1.3. GRANT, REVOKE

Attribution (retrait) d'un rôle à un autre.

GRANT nom_role [, ...] TO spécification_rôle [, ...]
[WITH { ADMIN | INHERIT | SET }]
[GRANTED BY spécification_rôle]

où

```
spécification_rôle ::=
    nom_rôle
  | PUBLIC
  | CURRENT_ROLE
  | CURRENT_USER
  | SESSION_USER
```

Attribution (retrait) d'un droit d'accès à un rôle

GRANT droit-accès
ON type-d'objet objet [, ...]
TO spécification_rôle
[WITH GRANT OPTION]

[GRANTED BY spécification_rôle]

où

```
droit-accès ::=
    SELECT
  | INSERT
  | UPDATE
  | DELETE
  | TRUNCATE
  | REFERENCES
  | TRIGGER
  | CREATE
  | CONNECT
  | TEMPORARY
  | EXECUTE
  | USAGE
  | SET
  | ALTER SYSTEM
  | ...

type-d'objet ::=
    TABLE
  | TYPE
  | ROUTINE
  | ...
```

Voir la documentation de PostgreSQL pour déterminer les paires (droit-accès, type-objet) légitimes et le sens précis du droit en regard du type d'objet.

1.1.4. SET ROLE

Changement de role en cours de session.

Voir la documentation de PostgreSQL pour déterminer modalités du changement de rôle.

1.1.5. REASSIGN

Tout objet se voit attribué un propriétaire au moment de sa création, cette instruction permet d'en changer. Le propriétaire d'un objet hérite automatiquement de certains droits.

Voir la documentation de PostgreSQL pour déterminer modalités du changement de propriétaire.

1.1.6. Les roles prédéfinis

Certains rôles sont prédéfinis a niveau du SGBD. Ceux-ci varient d'un dialecte à un autre.

Voir la documentation de PostgreSQL pour connaître les rôles prédéfinis.

1.1.7. La définition des routines (INVOKER et DEFINER)

> [EXTERNAL] SECURITY INVOKER
> [EXTERNAL] SECURITY DEFINER
> SECURITY INVOKER indique que la fonction est exécutée avec les droits de

> l'utilisateur qui l'appelle. C'est la valeur par défaut. SECURITY DEFINER spécifie que la fonction est exécutée avec les droits de l'utilisateur qui en est le propriétaire. […]
>
> Le mot clé EXTERNAL est autorisé pour la conformité SQL mais il est optionnel car, contrairement à SQL, cette fonctionnalité s'applique à toutes les fonctions, pas seulement celles externes.
>
> — PG, Partie VI - Référence - CREATE FUNCTION

1.2. Authentification

> PostgreSQL offre quantité de méthodes d'authentification différentes. La méthode utilisée pour authentifier une connexion client particulière peut être sélectionnée d'après l'adresse (du client), la base de données et l'utilisateur.
>
> Les noms d'utilisateur de bases de données sont séparés de façon logique des noms d'utilisateur du système d'exploitation sur lequel tourne le serveur. Si tous les utilisateurs d'un serveur donné ont aussi des comptes sur la machine serveur, il peut être pertinent d'attribuer aux utilisateurs de bases de données des noms qui correspondent à ceux des utilisateurs du système d'exploitation. Cependant, un serveur qui accepte des connexions distantes peut avoir des utilisateurs de bases de données dépourvus de compte correspondant sur le système d'exploitation. Dans ce cas, aucune correspondance entre les noms n'est nécessaire.
>
> — PG, chapitre 21

> PostgreSQL offre quantité de méthodes d'authentification différentes. La méthode utilisée pour authentifier **le client associé à une connexion** particulière peut être sélectionnée d'après l'adresse **d'origine de la connexion**, la base de données et le **nom d'utilisateur associé à la connexion**.
>
> — LL, 2024-04-01

Produit le 2025-05-06 12:52:05 +0100

Collectif francophone pour l'enseignement libre de l'informatique