Exploitation d'une Base de Données

Administration

Jules Azemar Raphaël Delage Anaïs Durand
Franck Glaziou Julie Rossignol

2023

IUT CLERMONT AUVERGNE
Aurillac - Clermont-Ferrand - Le Puy-en-Velay
Montluçon - Moulins - Vichy

Rôles

- Connexion à une base de données

  $\Rightarrow$ avoir un “compte utilisateur”

- En PostgreSQL (depuis la version 8.1) :

  un *rôle* =

  - un utilisateur

    ou

  - un groupe d'utilisateurs

- Lister les rôles existants :

```
SELECT rolname FROM pg_roles;
```

Création d'un rôle

- *Rôle :*

```
CREATE ROLE nom_role;
```

Possible seulement avec les privilèges d'administration

Création d'un rôle

- *Rôle :*

```
CREATE ROLE nom_role;
```

Possible seulement avec les privilèges d'administration

- *Utilisateur* = rôle avec droit de connexion à la base

```
CREATE ROLE nom_utilisateur LOGIN;
```

ou

```
CREATE USER nom_utilisateur;
```

Connexion avec mot de passe

 Si la base de données est configurée pour demander une authentification par mot de passe, seuls les utilisateurs ayant un mot de passe peuvent se connecter.

- *Définition du mot de passe :*

```
CREATE ROLE nom_utilisateur LOGIN
PASSWORD 'mon_mot_de_passe';
```

Connexion avec mot de passe

⚠ Si la base de données est configurée pour demander une authentification par mot de passe, seuls les utilisateurs ayant un mot de passe peuvent se connecter.

- *Définition du mot de passe :*

```
CREATE ROLE nom_utilisateur LOGIN
PASSWORD 'mon_mot_de_passe';
```

- *Durée de validité :* (optionnel)
  Rajouter l'option :

```
VALID UNTIL '2023-08-31';
```

Droits d'administration

- `SUPERUSER` : tous les droits d'administration
- `CREATEDB` : autorise la création de bases de données
- `CREATEROLE` : autorise la création de rôles

Droits d'administration

- `SUPERUSER` : tous les droits d'administration
- `CREATEDB` : autorise la création de bases de données
- `CREATEROLE` : autorise la création de rôles

Exemple :

```
CREATE ROLE nom_utilisateur LOGIN
PASSWORD 'mon_mot_de_passe'
CREATEDB;
```

Groupe de rôles

Gérer les privilèges utilisateur par utilisateur :

- long
- propice aux erreurs
- pas pratique pour les modifications
- ...

$\Rightarrow$ gestion des privilèges par ***groupes*** d'utilisateurs

Groupe de rôles

Gérer les privilèges utilisateur par utilisateur :

- long
- propice aux erreurs
- pas pratique pour les modifications
- ...

$\Rightarrow$ gestion des privilèges par ***groupes*** d'utilisateurs

- `GRANT role_de_groupe TO role1, role2, ...;`
  Ajoute `role1`, `role2`, ... au groupe `role_de_groupe`.

- `REVOKE role_de_groupe FROM role1, role2, ...;`
  Enlève `role1`, `role2`, ... du groupe `role_de_groupe`.

(Généralement, un groupe est un rôle sans l'option `LOGIN`)

Héritage

Un rôle ***hérite*** des privilèges des groupes auxquels il appartient

SAUF si le rôle a été créé avec l'option `NOINHERIT`.

Héritage

Un rôle *hérite* des privilèges des groupes auxquels il appartient

SAUF si le rôle a été créé avec l'option `NOINHERIT`.

- *Changement temporaire de rôle :* `SET ROLE autre_role;`

Exemple :

```
CREATE ROLE dieu CREATEDB
    CREATEROLE;
CREATE ROLE freyja LOGIN
    NOINHERIT;
GRANT dieu TO freyja;
```

Connexion en tant que `freyja`

Héritage

Un rôle *hérite* des privilèges des groupes auxquels il appartient

SAUF si le rôle a été créé avec l'option `NOINHERIT`.

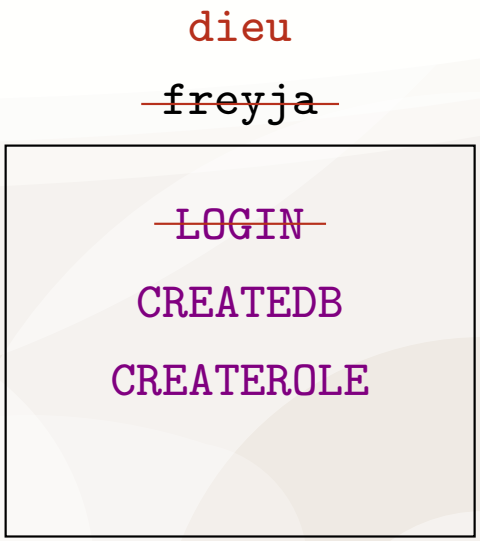
- *Changement temporaire de rôle :* `SET ROLE autre_role;`

Exemple :

```
CREATE ROLE dieu CREATEDB
    CREATEROLE;
CREATE ROLE freyja LOGIN
    NOINHERIT;
GRANT dieu TO freyja;
```

Connexion en tant que `freyja`

```
SET ROLE dieu;
```



Héritage

Un rôle *hérite* des privilèges des groupes auxquels il appartient

SAUF si le rôle a été créé avec l'option `NOINHERIT`.

- *Changement temporaire de rôle :* `SET ROLE autre_role;`

Exemple :

```
CREATE ROLE dieu CREATEDB
    CREATEROLE;
CREATE ROLE freyja LOGIN
    NOINHERIT;
GRANT dieu TO freyja;
```

Connexion en tant que `freyja`

```
SET ROLE dieu;
SET ROLE freyja;
```

Privilèges du propriétaire

Tous les objets (y compris les tables) ont un ***propriétaire***.

Par défaut le propriétaire est le créateur de l'objet.

Le propriétaire est le ***seul*** à pouvoir :

- Modifier l'objet (par exemple : ajouter des colonnes à une table)
- Supprimer l'objet

Privilèges du propriétaire

Tous les objets (y compris les tables) ont un *propriétaire*.

Par défaut le propriétaire est le créateur de l'objet.

Le propriétaire est le *seul* à pouvoir :

- Modifier l'objet (par exemple : ajouter des colonnes à une table)
- Supprimer l'objet

- *Transfert de propriété :*

```
ALTER TABLE nom_table OWNER TO nouveau_proprietaire;
```

Autres privilèges sur un objet

En dehors des privilèges du propriétaire, des privilèges sur les objets peuvent être attribués ou retirés à un rôle.

- *Attribution d'un privilège :*

```
GRANT privilege1, privilege2, ... ON nom_table TO role;
```

- *Retrait d'un privilège :*

```
REVOKE privilege1, privilege2, ... ON nom_table FROM role;
```

`PUBLIC` permet d'attribuer ou de retirer un privilège à tous les rôles.

Exemples de privilèges sur une table

- `SELECT` : droit de lire la table
- `INSERT` : droit d'insérer des lignes dans la table
- `UPDATE` : droit de modifier des lignes de la table
- `DELETE` : droit de supprimer des lignes de la table
- `REFERENCES` : droit de faire référence à la table via une clé étrangère
- ...