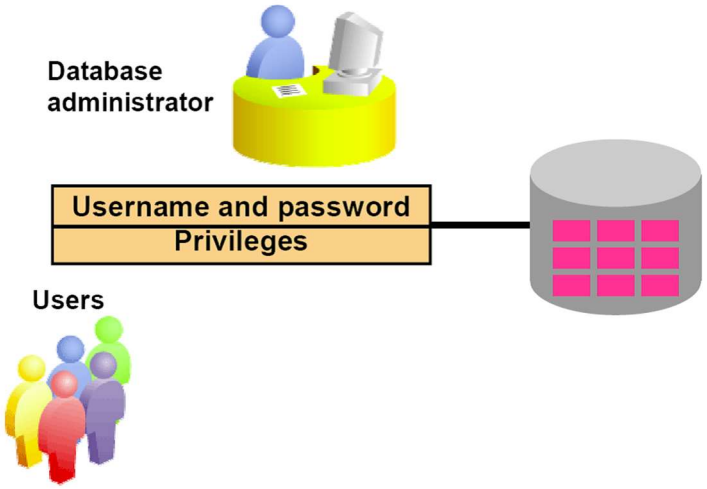
I. Présentation

Dans un environnement multiutilisateur, vous voulez certainement gérer l'accès à la base de données et la sécurité d'utilisation de la dernière.
Avec la sécurité du serveur de base de données Oracle, vous pouvez réalisez les actions suivantes :

- Contrôler l'accès à la base de données
- Donner l'accès à des objets spécifiques de la base de données
- Vérifier les privilèges donnés et reçus avec le dictionnaire des données Oracle
- Créer les synonymes pour les objets de la base de données

La sécurité de la base peut être classifiée en deux catégories : la sécurité du système et la sécurité des données.
La sécurité du système couvre l'accès à la base de données ainsi que l'utilisation de la base de données au niveau du système. Par exemple, le nom d'utilisateur et le mot de passe, l'espace disque alloué aux utilisateurs et les opérations que les utilisateurs peuvent réaliser sur le système.
La sécurité des données couvre l'accès et l'utilisation des objets de la base de données ainsi que les actions que les utilisateurs peuvent effectuer sur les objets.



Privilèges

Le droit d'exécuter la requête SQL spécifique s'appelle un privilège.
L'administrateur de base de données (DBA) est un utilisateur haut niveau qui a le droit de créer des utilisateurs et de leur donner des droits d'accès à la base de données et aux objets. Les utilisateurs ont besoin des *privilèges système* pour accéder à la base de données et des *privilèges objet* pour manipuler le contenu des objets de la base de données. Les utilisateurs peuvent recevoir le privilège permettant d'accorder des privilèges à d'autres utilisateurs ou à des *rôles* qui sont des groupes nommés de privilèges.

Les schémas

Un *schéma* est une collection d'objets comme des tables, des vues ou des séquences. Le schéma appartient à un utilisateur et porte son nom.

II. Les instructions de gestion des utilisateurs

Instruction CREATE USER

Crée un utilisateurs

Syntaxe

CREATE USER *utilisateur* IDENTIFIED BY *motdepasse*

L'instruction CREATE USER se compose des éléments suivants :

Composant	Description
utilisateur	Nom de l'utilisateur
motdepasse	Mot de passe à associer au nom d'*utilisateur* spécifié.

Notes

deux *utilisateurs* ne peuvent pas porter le même nom.

le *mot de passe* est facultatif mais fortement recommandé pour chaque *utilisateur* créé.

Instruction ALTER USER

Change le mot de passe d'un utilisateur ou d'une base de données existante.

Syntaxe

ALTER USER user IDENTIFIED BY password;

L'instruction ALTER USER se compose des éléments suivants :

Composant	Description
utilisateur	Nom de l'utilisateur
nouveaumotdepasse	Nouveau mot de passe à associer au nom d'*utilisateur*.

Instruction DROP USER

Supprime un *utilisateur* existant

Syntaxe

DROP USER *utilisateur*

L'instruction DROP USER

Composant	Description
utilisateur	Nom d'un utilisateur à supprimer

III. Les privilèges spécifiques des utilisateurs

Une fois que le DBA ait créé un utilisateur, il peut lui accorder des privilèges.
Voici les privilèges système généralement accordés aux développeurs d'applications.

Privilège système	Operations autorisées
CREATE SESSION	se connecter à la base de données
CREATE TABLE	créer des tables dans le schéma de l'utilisateur
CREATE SEQUENCE	créer des séquences dans le schéma de l'utilisateur
CREATE VIEW	créer des vues dans le schéma de l'utilisateur
CREATE	créer des procédures stockées, des fonctions ou des packages dans le schéma

Privilège système	Operations autorisées
PROCEDURE	de l'utilisateur

Attribuer des privilèges à un utilisateur : Instruction GRANT

Accorde des privilèges spécifiques à un utilisateur ou à un groupe existant.

Syntaxe
GRANT *privilege* **ON** *nom de la table ou autre objet* **TO** *utilisateur*

GRANT object_priv [(columns)]|ALL ON object
TO {user|role|PUBLIC}
[WITH GRANT OPTION];

Dans la syntaxe :

- *object_priv* est un privilège objet
- **ALL** spécifie que tous les privilèges objet
- *columns* spécifie que les colonnes de la table ou de la vue auxquelles les privilèges s'appliquent.
- **ON** *object* est l'objet sur lequel les privilèges s'appliquent
- **TO** identifie à qui les privilèges ont assignés
- **PUBLIC** assigne les privilèges objets à tous les utilisateurs

WITH GRANT OPTION permet à l'utilisateur de donner les privilèges objets à d'autres utilisateurs et rôles

L'instruction GRANT se compose des éléments suivants :

Composant	Description
privilège	Le ou les privilèges à accorder. Les privilèges sont spécifiés à l'aide des mots-clés suivants : SELECT, DELETE, INSERT, UPDATE, DROP, SELECTSECURITY, UPDATESECURITY, DBPASSWORD, UPDATEIDENTITY, CREATE, SELECTSCHEMA, SCHEMA et UPDATEOWNER.
utilisateur	Le nom de l'utilisateur à qui le privilège est accordé

Si la requête **GRANT** inclut **WITH GRANT OPTION** alors la personne recevant le privilège en question pourra à son tour donner ce privilège à d'autres utilisateurs. Si la clause **WITH GRANT OPTION** est absente l'utilisateur ayant reçu un privilège objet ne peut pas l'assigner à quelqu'un d'autre.

Règles :

- Vous pouvez assigner les privilèges sur un objet s'il est dans votre schéma ou si le privilège sur l'objet en question vous a été assigné en utilisant **WITH GRANT OPTION**.
- Le propriétaire de l'objet peut accorder n'importe quel privilège sur cet objet à tout utilisateur ou rôle de la base de données.
- Le propriétaire de l'objet acquière automatiquement tous les privilèges objet sur ce dernier.

La requête suivante donne aux utilisateurs Sue et Rich le privilège permettant d'interroger votre table **EMPLOYEES**

```
GRANT select ON employees
TO sue, rich;
```

La requête ci-dessous donne le privilège **UPDATE** sur les colonnes **DEPARTMENT_NAME** et **LOCATION_ID** de la table **DEPARTMENTS** à l'utilisateur Scott ainsi qu'au rôle manager.

```
GRANT update (department_name, location_id) ON departments
TO scott, manager;
```

Maintenant, si Sue et Rich veulent utiliser la requête **SELECT** pour obtenir des données de la table **EMPLOYEES**, ils doivent utiliser la syntaxe suivante :
SELECT * FROM HR.employees;
Autrement, ils peuvent créer un synonyme pour la table et lancer une requête **SELECT** sur le synonyme :

```
CREATE SYNONYM emp FOR HR.employees;
SELECT * FROM emp;
```

WITH GRANT OPTION
L'utilisateur ayant reçu un privilège avec la clause **WITH GRANT OPTION** peut l'assigner à un autre utilisateur ou à un autre rôle. Si le privilège est retiré à cet utilisateur, tous les utilisateurs à qui il avait donné ce privilège se le verront enlever de manière automatique.

L'exemple suivant donne à l'utilisateur Scott le privilège **SELECT** et **INSERT** sur votre table **DEPARTMENTS**. Scott pourra accorder ces privilèges aux autres.

```
GRANT select, insert ON departments
TO scott
WITH GRANT OPTION;
```

Instruction REVOKE

Retire des privilèges spécifiques à un utilisateur ou à un groupe existant.

Syntaxe

```
REVOKE {privilege [, privilege...]|ALL}
ON object
FROM {utilisateur[,utilisateur...]|role|PUBLIC}
[CASCADE CONSTRAINTS];
```

L'instruction REVOKE se compose des éléments suivants :

Composant	Description
privilège	Le ou les privilèges à retirer. Les privilèges sont spécifiés à l'aide des mots-clés suivants : SELECT, DELETE, INSERT, UPDATE, DROP, SELECTSECURITY, UPDATESECURITY, DBPASSWORD, UPDATEIDENTITY, CREATE, SELECTSCHEMA, SCHEMA et UPDATEOWNER.

objet	Tout objet tel qu'une table, une requête stockée (vue ou procédure) par exemple.
CASCADE CONSTRAINTS	supprime toutes les contraintes d'intégrité référencées

Si un employé a quitté l'entreprise et que vous supprimez ses privilèges, vous devez réassigner tous les privilèges que cet utilisateur a donnés aux autres. Si vous supprimez le compte de l'utilisateur sans lui retirer les privilèges alors les privilèges systèmes que celui-ci a donnés aux autres utilisateurs ne sont pas affectés par l'action.

L'utilisateur Alice peut retirer les privilèges **SELECT** et **INSERT** donnés à l'utilisateur Scott sur la table **DEPARTMENTS** en utilisant la requête suivante.

```
REVOKE select, insert
ON departments
FROM scott;
```

Si un utilisateur a reçu le privilège avec la clause **WITH GRANT OPTION**, cet utilisateur peut aussi donner ce privilège avec la clause **WITH GRANT OPTION**, donc une longue chaine d'assignations des privilèges est possible. Or, les assignations circulaires ne sont pas autorisées (accorder le privilège à la personne qui l'a accordé en premier).

Si le propriétaire de l'objet retire un privilège de l'utilisateur qui l'a accordé à d'autres utilisateurs, alors la suppression est répercutée en cascade.

Par exemple, si l'utilisateur **A** donne le privilège **SELECT** sur une table à l'utilisateur **B** en incluant la clause **WITH GRANT OPTION**, l'utilisateur **B** peut à son tour donner ce privilège **SELECT** avec la clause **WITH GRANT OPTION** à l'utilisateur **C** qui peut donner le privilège **SELECT** à l'utilisateur **D**. Si l'utilisateur **A** retire le privilège à l'utilisateur **B** alors les privilèges sont automatiquement retirés aux utilisateurs **C** et **D**.

Renommer un utilisateur :

```
RENAME USER old_name TO new_name;
```

IV. Privilèges système

Il existe plus de 100 privilèges système disponibles pour les utilisateurs et les rôles. Les privilèges système sont habituellement donnés par l'administrateur de la base de données.

Les privilèges spécifiques du DBA

Voici quelques privilèges système dont le DBA dispose :

Privilège système	Operations autorisées
CREATE USER	Permet de créer d'autres utilisateurs Oracle
DROP USER	Permet de supprimer d'autres utilisateurs
DROP ANY TABLE	Permet de supprimer une table dans n'importe quel schéma
BACKUP ANY	Permet de sauvegarder et de restaurer n'importe quelle table dans n'importe quel

Privilège système	Operations autorisées
TABLE	schéma avec l'utilitaire d'export
SELECT ANY TABLE	Permet d'interroger les tables, les vues ou les vues matérialisées dans n'importe quel schéma
CREATE ANY TABLE	Permet de créer des tables dans n'importe quel schéma

L'administrateur de la base de données a la plupart des privilèges permettant, entre autre, de :

- Créer de nouveaux utilisateurs
- Supprimer des utilisateurs
- Supprimer des tables
- Récupérer les tables

Après qu'un utilisateur soit créé, le DBA peut lui donner des privilèges systèmes en utilisant la syntaxe suivante :

Dans la syntaxe :

- *privilege* est le privilège système qui sera assigné
-

Les privilèges systèmes de l'utilisateur courant peuvent être trouvés dans la vue du dictionnaire des données **SESSION_PRIVS**.

Assigner des privilèges système

Le DBA utilise la commande **GRANT** pour donner des privilèges système à un utilisateur. Après que l'utilisateur ait reçu les privilèges, il peut immédiatement les utiliser.

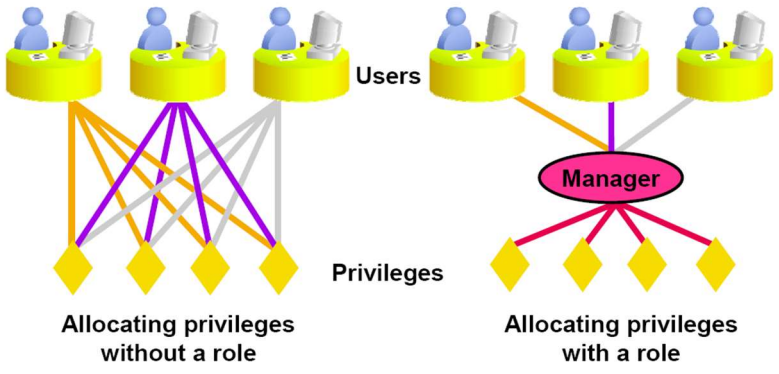
Dans l'exemple, l'utilisateur Scott a reçu les privilèges permettant d'ouvrir des sessions, de créer des tables, des séquences et des vues.

GRANT create session, create table, create sequence, create view
TO scott;

V. Les rôles

Un rôle est un groupe nommé de privilèges qui peuvent être assignés à un utilisateur. Cette méthode facilite la gestion des privilèges.

Un utilisateur peut avoir accès à plusieurs rôles et plusieurs utilisateurs peuvent recevoir le même rôle. Les rôles sont spécialement créés pour une application de la base de données.



Créer et assigner un rôle

Tout d'abord, le DBA doit créer le rôle. Ensuite, il peut assigner des privilèges au rôle et assigner le rôle à des utilisateurs.

Syntaxe :

CREATE ROLE role;

Dans la syntaxe :

- *role* est le nom du rôle qui sera créé

Après que le rôle soit créé, le DBA peut utiliser la requête **GRANT** pour assigner les privilèges au rôle tout comme assigner ce rôle aux utilisateurs.

Dans l'exemple, un rôle manager est créé, ensuite les privilèges permettant de créer des tables et des vues lui sont assignés. Le rôle est assigné à De Haan et Kochhar. Désormais, De Haan et Kochhar peuvent créer des tables et des vues.

Si un utilisateur a plusieurs rôles, il reçoit tous les privilèges associés à tous ces rôles.

Exemple

Créer un rôle

- CREATE ROLE manager;

Assigner des privilèges au rôle

- GRANT create table, create view TO manager;

Assigner le rôle aux utilisateurs

- GRANT manager TO Magengo, Guttembert;