SGBD TP2 :

Gestion des utilisateurs et des droits, Séquence

- Durée 1h30
- Pour ce TP vous avez besoin de la BD GestAsso.

Travail à faire :

Gestion des utilisateurs et des droits

1. On veut donner à l'utilisateur **tpsgbd** tous les privilèges d'un administrateur, alors en étant **system** donnez le privilège **ALL PRIVILEGES** à l'utilisateur **tpsgbd.**

2. Etant **TpSGBD,**

2.1. Créer un utilisateur oracle ayant comme nom **invit1_user** avec les caractéristiques suivantes :

- tablespace par défaut : users
- tablespace temporaire : temp
- mot de passe : 1234

2.2. Créer un utilisateur oracle ayant comme nom **invit2** avec les caractéristiques suivantes :

- tablespace par défaut : users
- tablespace temporaire : temp
- mot de passe : 4321

2.3. Consulter les données concernant les deux utilisateurs que vous venez de crées

```
ⓘ SELECT * FROM all_users WHERE username LIKE name_user;
```

3. Connectez vous comme étant **invit1**.

3.1. Que constatez vous ?

3.2. Remédier au problème pour que l'utilisateur **invit1** réussi à se connecter**.**

3.3. Etant l'utilisateur **TpSGBD,** consulter la liste des privilèges systèmes et objets de **invit1.**

```
ⓘ  SELECT PRIVILEGE FROM dba_sys_privs WHERE grantee=name_user;
   SELECT PRIVILEGE FROM dba_tab_privs WHERE grantee=name_user;
```

4. Etant **invit1**, tentez de créer la table **Tab_Act_familles :**

```
CREATE TABLE tab_act_familles (
nofam NUMERIC(7) NOT NULL,
noact VARCHAR(10) NOT NULL,
descr VARCHAR(80)
);
```

4.1. Que constatez vous?

4.2. Remédier au problème et essayer de créer la table **Tab_Act_familles.**

4.3. Essayer de créer à nouveau la table **Tab_Act_familles.**

5. Etant l'utilisateur **TpSGBD** donner à l'utilisateur **invit1** le privilège de faire référence à la clé primaire de la table **Action** et de la table **Famille**.

```
ⓘ GRANT REFERENCES(attribut) ON table TO user ;
```

6. Etant l'utilisateur **invit1**

6.1. Ajouter les deux contraintes à la table **Tab_Act_familles** permettant de faire référence aux deux tables **Action** et **Famille**.

6.2. Insérer le tuple suivant dans la table **Tab_Act_familles :**

```
SQL>INSERT INTO tab_act_familles VALUES(4,'Act4','3 niveaux scolaires: 1er primaire(F), 3eme primaire(G), 4eme primaire(G)');
```

6.3. Supprimer le privilège attribué à l'utilisateur invit1 dans la question 5.

ⓘ *Pour que REVOKE s'exécute il faut fermer la session de l'utilisateur invit1*

6.4. Etant l'utilisateur **TpSGBD** exécuter la requête ci_dessous. Que remarquez vous ?

```
SQL> SELECT CONSTRAINT_NAME,SEARCH_CONDITION
FROM DBA_CONSTRAINTS
WHERE TABLE_NAME LIKE UPPER('Tab_Act_familles') ;
```

7. Etant l'utilisateur **TpSGBD**

7.1. Créer un role **RL_tresorier** qui permet de consulter, d'insérer et de modifier des données dans la table **Donation.**

7.2. Créer un role **RL_membre** qui permet de consulter la table **donation** et d'insérer et de consulter des données dans la table **Action.**

8. Attribuer le Rôle **RL_tresorier** à l'utilisateur **invit1** et le rôle **RL_membre** à à l'utilisateur **invit2.**

ⓘ Pour consulter les deux rôles **RL_tresorier et RL_membre :**

```
SQL>SELECT role, Table_name,column_name, privilege, owner from role_tab_privs
WHERE role LIKE UPPER('rl_tresorier')
OR role LIKE UPPER('rl_membre');
```

9. Etant l'utilisateur **invit2** consulter le nombre d'actions par type**.**

10. Etant l'utilisateur **TpSGBD** supprimer le rôle **RL_membre.**

11. Etant l'utilisateur **invit2** consulter la liste d'actions**.** Que constatez vous ?

12. Etant l'utilisateur **TpSGBD,** consulter l'état des rôles attribués aux autres utilisateurs**.**

ⓘ *SQL>SELECT role, Table_name,column_name, privilege from role_tab_privs WHERE owner LIKE UPPER('courssgbd');*

Les séquences

ⓘ Avant de commencer cette partie il faut exécuter les requêtes ci dessous :

```
SQL> DELETE FROM Donation ;
SQL> DELETE FROM Adherent ;
SQL> DELETE  FROM Action;
```

13. Créer une séquence Seq_joker qui sera utilisée pour les valeurs de la clé primaire de la table **Adherant** et de la table **Action** (sans cycle, sans cache et qui commence par la valeur 1).

14. Insérer les données ci-dessous dans les tables Adherents et Actions en faisant appel à la séquence crée. (*Faire attention à l'ordre de l'insertion des données dans les tables) :*

Les Adherents :
- 1, Ben Yahya, Amine, 54778899, Actif, bya@gmail.com, 56743817
- 3, Ben said, Sondes, 92058391, Porte parole, bss@gmail.com, 90123897

Les Actions
- Act2, Rentrée scolaire 2015, 08/15/2014,09/15/2014, , 100
- Act 4, Rentrée scolaire 2016, 08/15/2015,09/15/2015, ,100

15. Supprimer l'adhèrent numéro 1 de la table Adhèrent puis consulter les données de la table.

16. Ajouter l'adhérant : Benahmed, amir, 96098764, Actif,baa@gmail.com, 90125697 et consulter de nouveau les données de la table.

ⓘ *Les valeurs supprimées de la clé primaire ne sont pas réutilisées par la séquence car la séquence ne revient pas en arrière, et lors de l'insertion suivante, une nouvelle valeur est générée.*

17. consulter les informations concernant la séquence Seq_joker.

```
ⓘ SQL>SELECT sequence_name, min_value, max_value
increment_by, last_number
FROM user_sequences;
                                    ou
SQL>SELECT min_value, max_value
increment_by, last_number
FROM user_sequences
WHERE sequence_name LIKE nomSeq;
```

18. Modifier le pas de la sequence Seq_joker de +1 à +5

19. Insérer les données suivantes dans la table Action en faisant appel à la séquence Seq_joker.

```
SQL>INSERT INTO Action VALUES ('Act'|| (To_CHAR(Seq_joker.NEXTVAL)),'Lunette
de vue', '01/02/2017', '20/02/2017', 'collecte de fond pour les frais de
lunettes de vue : prix 60dt', 102);
```

20. Consulter avec la requête suivante la dernière valeur de la séquence.

```
ⓘ SQL> SELECT nom_seq.CURRVAL FROM SYS.dual;
```