|  |  |
| --- | --- |
| image001 | **Année : 2020/2021** |
|  | **TP-2** |
| **EFREI - M1-APP-LSI** | **Bases de Données Réparties** |
|  |  |
| **A. Lahlou** | [lahloukarim@free.fr](mailto:lahloukarim@free.fr) |

**TP-2**

**Cas Notes – SQL Avancé et PL/SQL**

**Gestion des vues, des droits et transactions et PL/SQL**

***Objectif :***

***SQL/SQL Avancé :Contraintes, Vues, Droits, transactions et concurrence d’accès***

***PL/SQL : Procédures, Fonctions et Triggers***

1. Création et manipulation de vues mono-table ; multi-tables et contenant des groupements

Commandes CREATE VIEW ; RENAME, …

2. Gestion des droits d’accès

Commandes GRANT ; REVOKE.

3. Gestion des transactions

Commandes COMMIT ; ROLLBACK.

4. PL/SQl

Commandes CREATE, REPLACE, PROCEDURE, FUNCTION, TRIGGER.

**I – Description et création de la base de données :**

Considérons la base de données de gestion de notes d’étudiants contenant les tables suivantes (script fourni en annexe).

Etudiant (**NumEtu**, Nom, Prenom, DateNaiss, Rue, CP, Ville)

Matiere (**CodeMat**, Libelle, Coef)

Epreuve (**numepreuve**, DateEpreuve, Lieu, #CodeMat)

Notation (**#NumEtu, #NumEpreuve,** Note)

Mettre le script fourni en Annexe1 dans un fichier .sql sous gedit ou notepad++ ou sqldevelopper ou datagrip ou autre et l’exécuter sous Oracle une fois que vous auriez ajouté les contraintes suivantes :

1. Ajouter les contraintes de clés primaires et de clés étrangères
2. Ajouter les contraintes d’intégrité suivantes dans le script SQL :

- la note d’un étudiant doit être comprise entre 0 et 20.

- Nom de l’étudiant toujours renseigné

- Libellé de la matière unique

- date de l’épreuve par défaut vaut la date système

**II – Instanciation de la base de données :**

1. Insérer les données correspondantes dans différentes tables et valider votre insertion par COMMIT.
2. Insérer l’étudiant :

*(1100 | TOTO | Albert | 01-07-1992| Rue de Crimée | 75019|Paris)*.

Cet étudiant a obtenu la note *13* à l’épreuve de numéro « *31031* », mettre à jour cette information dans la base de données. Que constatez-vous ? Justifier.

1. Mettre à jour le coefficient de Statistique à 0,3 et Informatique à 0,5 sachant qu’on doit toujours garder la somme des coefficients de toutes les matières doit rester à 1
2. Mettre à jour toutes les notes de l’épreuve 11031 à 14
3. Est-il possible de rajouter une épreuve avec participation de tous les étudiants (Certification Oracle de la matière Informatique avec une note de 10 + 2% du numéro d’étudiant s’il est inferieur 300 et de 10 + 1% du numéro d’étudiant s’il est supérieur ou égal à 300 en considérant que 18 sera la note maximale). Comment ?

**II – Interrogation :**

1. Vérifier la création de vos 4 tables et leur instanciation (Nombres de lignes par table par exemple).

2. Liste des notes supérieures ou égales à 10.

3. Liste des épreuves dont la date se situe entre le 1er janvier et le 30 juin 2019.

4. Nombre total d'épreuves.

5. Nombre de notes indéterminées (NULL).

6. Liste des notes en précisant pour chacune le nom et le prénom de l'étudiant qui l'a obtenue.

7. Moyennes des notes de chaque étudiant (indiquer le nom et le prénom), classées de la meilleure à la moins bonne (les notes à NULL sont à considérer comme 0).

8. Liste des étudiants qui ont obtenu la plus petite moyenne en Informatique

9. Liste des étudiants qui ont obtenu la meilleure moyenne

10. Liste des étudiants qui ont participé à toutes les épreuves

**III – Gestion des vues**

1. Créer la vue renfermant tous les étudiants ayant eu des épreuves en Informatique ainsi que les notes obtenues.

2. Donner la moyenne et le nombre d’épreuves en informatique de chaque étudiant ayant passé au moins une épreuve dans cette matière.

4. Donner les noms d’étudiants ainsi que leur moyenne ayant eu une moyenne supérieure ou égale à 10 en Informatique et classés par ordre de mérite.

5. Donner les noms d’étudiants qui n’ont passé aucune épreuve en Informatique.

**IV – Gestion de la confidentialité (Vues et Droits d’accès)**

1. Créer la vue EtudLyonnais renfermant tous les étudiants lyonnais (La vue obtenue contient tous les champs de la table Etudiant).

2. Donner la moyenne en informatique des étudiants lyonnais

3. L’étudiant Dupond a quitté l’établissement. Est-il possible de le supprimer depuis la vue EtudLyonnais ? Vérifier ?

4. L’étudiant Durand déménage lors de son stage à 1, Rue de Lyon Paris 75002. Est-il possible de mettre à jour ses informations depuis la vue EtudLyonnais ? Vérifier ?

5. Est-il possible d’ajouter l’étudiant (700, Gates, Bill, 1980-09-1, Rue de Paris, 69005, Lyon) depuis la vue EtudLyonnais ?  Vérifier ?

6. Donner la vue MoyLyonnais contenant la moyenne par matière des étudiants lyonnais.

7. Est-il possible de mettre à jour la moyenne en informatique de l’étudiant Dupont depuis la vue MoyLyonnais ?

8. Donnez un droit d’accès en consultation à la vue EtudLyonnais à votre binôme (Un second compte que vous allez créer si nécessaire sous Oracle). Vérifier le contenu de la vue depuis les 2 comptes.

9. Mettez à jour le nom de rue de Dupont qui passe à Rue de la Meuse, 69008, Lyon. Vérifier sur les 2 comptes si la mise à jour est faite. Le second compte essaye de remettre Dupont à son ancienne adresse. Que se passe –t-il ? Quoi faire ?

10. Votre binôme insère un nouvel étudiant lyonnais. Que doit-il faire pour que vous puissiez consulter ce nouvel étudiant ?

11. Une erreur de saisie entre 2 notes de 2 étudiants sur une même épreuve doit être corrigée. Comment procéder ?

12. Que doit-on faire pour éviter des mises à jour erronées des notes des étudiants par vous ou votre binôme (conflit de mise à jour) ?

**Annexe 1 :**

-- ============================================================

-- Nom de la base : BD\_Notation

-- Nom de SGBD : ORACLE

-- Date de création : Mars 2020

-- Auteur : Abdelkrim LAHLOU

-- ============================================================

-- ============================================================

ALTER SESSION SET NLS\_DATE\_FORMAT = 'DD-MM-YYYY' ;

-- ============================================================

-- Destruction des tables

DROP TABLE Etudiant CASCADE CONSTRAINTS;

DROP TABLE Matiere CASCADE CONSTRAINTS;

DROP TABLE Epreuve CASCADE CONSTRAINTS;

DROP TABLE Notation CASCADE CONSTRAINTS;

-- Creation des tables

-- Mettez les contraintes de cles primaires et contraintes de domaines

CREATE TABLE Etudiant (

NumEtu NUMBER(8),

Nom VARCHAR2(20),

Prenom VARCHAR2(20),

DateNais DATE,

Rue VARCHAR2(50),

CP CHAR(5),

Ville VARCHAR2(25)

);

CREATE TABLE Matiere (

CodeMat CHAR(3),

Libelle VARCHAR2(20),

Coef NUMBER(3,2)

);

CREATE TABLE Epreuve (

NumEpreuve NUMBER(10),

DateEpreuve DATE,

Lieu VARCHAR2(20),

CodeMat CHAR(3)

);

CREATE TABLE Notation (

NumEtu NUMBER(8),

NumEpreuve NUMBER(10),

Note NUMBER(4,2)

);

-- Ajouter les contraintes de cles etrangeres

REM ALTER TABLE Epreuve ADD CONSTRAINT FkEpreuve1 FOREIGN KEY(CodeMat) REFERENCES

REM Matiere(CodeMat);

-- Remplissage des tables

INSERT INTO Etudiant VALUES (110,'Dupont','Albert','01-06-1990','RuedeCrimée','69001','Lyon');

INSERT INTO Etudiant VALUES (222,'West','James','03-09-1993','Studio',NULL,'Hollywood');

INSERT INTO Etudiant VALUES (300,'Martin','Marie','05-06-1998','RuedesAcacias','69130','Ecully');

INSERT INTO Etudiant VALUES (421,'Durand','Gaston','15-11-1990','RuedelaMeuse','69008','Lyon');

INSERT INTO Etudiant VALUES (575,'Titgoutte','Justine','28-02-1995','CheminduChâteau','69630','Chaponost');

INSERT INTO Etudiant VALUES (667,'Dupond','Noémie','18-09-1997','RuedeDôle','69007','Lyon');

INSERT INTO Etudiant VALUES (999,'Phantom','Marcel','30-01-1990',NULL,NULL,NULL);

INSERT INTO Matiere VALUES ('STA','Statistique',0.4);

INSERT INTO Matiere VALUES ('INF','Informatique',0.4);

INSERT INTO Matiere VALUES ('ECO','Econométrie',0.2);

INSERT INTO Epreuve VALUES(11031,'15-12-2018','Salle191L','STA');

INSERT INTO Epreuve VALUES(11032,'01-04-2019','AmphiG','STA');

INSERT INTO Epreuve VALUES(21031,'30-10-2018','Salle191L','INF');

INSERT INTO Epreuve VALUES(21032,'01-06-2019','Salle192L','INF');

INSERT INTO Epreuve VALUES(31030,'02-07-2019','Salle05R','ECO');

INSERT INTO Notation VALUES (110,11031,10);

INSERT INTO Notation VALUES (110,11032,11.5);

INSERT INTO Notation VALUES (110,21031,8.5);

INSERT INTO Notation VALUES (110,21032, NULL);

INSERT INTO Notation VALUES (110,31030,13);

INSERT INTO Notation VALUES (222,11031,9);

INSERT INTO Notation VALUES (222,11032,14);

INSERT INTO Notation VALUES (222,21031,12);

INSERT INTO Notation VALUES (222,21032,16);

INSERT INTO Notation VALUES (222,31030,20);

INSERT INTO Notation VALUES (300,11031,14);

INSERT INTO Notation VALUES (300,11032,20);

INSERT INTO Notation VALUES (300,21031,20);

INSERT INTO Notation VALUES (300,21032,13.5);

INSERT INTO Notation VALUES (300,31030,16);

INSERT INTO Notation VALUES (421,11031,5.5);

INSERT INTO Notation VALUES (421,11032,17);

INSERT INTO Notation VALUES (421,21031,1.5);

INSERT INTO Notation VALUES (421,21032, NULL);

INSERT INTO Notation VALUES (421,31030,10);

INSERT INTO Notation VALUES (575,11031,13);

INSERT INTO Notation VALUES (575,11032,9);

INSERT INTO Notation VALUES (575,21031,12.5);

INSERT INTO Notation VALUES (575,21032,14);

INSERT INTO Notation VALUES (575,31030,7);

INSERT INTO Notation VALUES (667,11031,16);

INSERT INTO Notation VALUES (667,11032,20);

INSERT INTO Notation VALUES (667,21031,8.5);

INSERT INTO Notation VALUES (667,21032,9.5);

**Annexe-2**

Drop user User1 cascade;

create user Apprenti identified by M1appLSI

default tablespace USERS

temporary tablespace TEMP

quota 2 M on USERS

PROFILE DEFAULT

ACCOUNT UNLOCK;

GRANT ALTER SESSION TO Apprenti;

GRANT CREATE DATABASE LINK TO Apprenti;

GRANT CREATE SYNONYM TO Apprenti;

GRANT CREATE VIEW TO Apprenti;

GRANT CREATE TRIGGER TO Apprenti;

GRANT DEBUG CONNECT SESSION TO Apprenti;

GRANT UNLIMITED TABLESPACE TO Apprenti;

GRANT CONNECT TO Apprenti;

GRANT RESOURCE TO Apprenti;

GRANT SELECT\_CATALOG\_ROLE TO Apprenti;

**Manipulation de Base SQL\*PLUS :**

*Aide en ligne :*

**help <nom-commande>** : aide sur la commande citée (ex. : help exit)

*Dialogue avec le système hôte :*

**host <commande>** ou **! <commande>** : lance une commande du système d'exploitation.

*Travail sous sqlplus :*

**Sous l’interface SQLPLUS directement :** la plupart des commandes sql peuvent être tapées sur une ou plusieurs lignes et doivent être terminées par ; SQL ne fait pas la différence entre ses commandes minuscules et majuscules. Il est toutefois recommandé de mettre les mots SQL en majuscules, d'aller à la ligne et d'indenter les lignes de commande.

**A partir d'un script :** On écrit une suite de commandes (terminées par un ;) à l'aide d'un éditeur de texte (vi, emacs, xemacs, ...), bien qu'il y ait un buffer SQL et l'on sauvegarde le fichier dans le répertoire de travail, le nom du fichier devant avoir pour extension **.sql.** On exécute les commandes soit en faisant du "couper/coller" de l'éditeur vers SQL, soit en dialoguant avec le système hôte de la manière suivante :

**start** <*nomprog*> ou **@** <*nomprog*> : charge le fichier et lance les commandes contenues dans le fichier *nomprog.sql* (inutile de spécifier l’extension .sql)

*Memento SQL\*PLUS :*

**save filename.sql** sauve le contenu du buffer dans un fichier de nom filename.sql

**get filename.sql** charge le buffer avec le contenu du fichier de nom filename.sql

**start filename** charge le buffer et lance l'exécution du fichier script sql

**run**  lance l'exécution du contenu du buffer

**spool filename.txt** copie la sortie écran sur le fichier filename.txt

**spool off** suspend l'opération précédente

**help commande** pour obtenir de l'aide sur la commande donnée en argument

*Variables d'environnement :*

**set long 1024** pour voir la totalité des définitions de vues

**set pagesize 10** formate la sortie écran par blocs de 20 lignes

**set pause on** ne visualise la sortie qu'après un 2ième Return - bloc par bloc -

*Schéma d'information ( ou "méta-base") :*

**desc[ribe] tablename** donne le schéma de la relation tablename

**all\_catalog** relation donnant toutes les tables accessibles

**user\_catalog** relation donnant les seules tables du USER

**cat**  synonyme de la précédente

**user\_tables** relation donnant toutes les tables du USER

**tab** synonyme de la précédente

**all\_objects** relation donnant tous les objets accessibles

**user\_objects** relation donnant les seuls objets du USER

**obj**  synonyme de la précédente ..

**user\_sys\_privs** relation donnant les privilèges système du USER

**user\_tab\_privs** relation donnant les privilèges sur les objets accessibles