

L3 INFORMATIQUE-L3 MIASHS – M1 MSID

BASES DE DONNEES

LE SGBD RELATIONNEL ORACLE®

L'accès à la base de données d'oracle nommée etud peut se faire à partir de l'éditeur sqlplus. Pour lancer ce module d'édition et d'exécution de requêtes SQL la commande ci-dessous est utilisée : sqlplus **NomUtilisateur**@etud (le mot de passe qui vous est ensuite demandé est votre **NomUtilisateur**).

Vous pouvez également utiliser **sqldeveloper** afin d'avoir accès à la base.

DESCRIPTION DES DONNEES

Commandes CREATE, ALTER, DROP

Création de tables :

```
CREATE TABLE Nom_de_table
(Nom_attribut1 {char|varchar|number....} [NOT NULL]
[, Nom_attribut2 {char|varchar|number....} [NOT NULL]]
.....
[, Nom_attributn {char|varchar|number....} [NOT NULL]]
[[CONSTRAINT nomcontrainte1] Typecontrainte1]
[, [CONSTRAINT nomcontrainte2] Typecontrainte2]
.....
);
```

TYPE DE DONNEES :

CHAR (n [BYTE | CHAR]), **VARCHAR2** (n [BYTE | CHAR]), **CLOB**, **NLOB**, **LONG NUMBER** (n[,d])

DATE, **INTERVAL YEAR** (an) **TO MONTH**, **INTERVAL DAY** (jo) **TO SECOND** (fsec)

BLOB, **BFILE**, **RAW**(taille), **LONG RAW**

Définition des contraintes d'intégrité :

[**CONSTRAINT** nomcontrainte] **PRIMARY KEY** (Nom_attribut1[, Nom_attribut2]...)

[**CONSTRAINT** nomcontrainte] **FOREIGN KEY** (Nom_attribut1[,Nom_attribut2]...) **REFERENCES** Nom_table [(nom_attribut)]

[**CONSTRAINT** nomcontrainte] **CHECK** (Condition)

[**CONSTRAINT** nomcontrainte] **UNIQUE** (Nom_attribut1[, Nom_attribut2]...)

Remarque : Le nommage d'une contrainte est optionnel cependant il est vivement recommandé. Si un nom n'est pas attribué le SGBD affecte un nom par défaut.

Création d'une table à partir du résultat d'une requête SQL :

```
CREATE TABLE Nom_de_table [(Nom_col1, Nom_col2, ....)]
AS requête ;
```

Dans ce cas la table est créée et les données résultat de la requête sont insérées dans la table. Attention aucune contrainte n'est définie sur la table obtenue, l'ajout de contraintes doit être réalisé avec la commande ALTER TABLE.

Création de séquence : Séquence, possibilité de générer des valeurs automatiquement à partir d'une valeur initiale jusqu'à une valeur finale.

CREATE SEQUENCE nom_seq

[INCREMENT BY entier] [START WITH entier] [{MAXVALUE entier | NOMAXVALUE}] [{MINVALUE entier | NOMINVALUE}] [{CYCLE | NOCYCLE}]

Utilisation de la séquence :

nom_seq.CURRVAL retourne la valeur courante de la séquence

nom_seq.NEXTVAL incrémente la séquence et retourne la valeur.

Modification de la structure des tables et des contraintes :

Ajout, modification, suppression de colonnes ou de contraintes:

ALTER TABLE Nomtable **ADD** (Nomattribut {char|varchar|number....} [NOT NULL]);

ALTER TABLE Nomtable **MODIFY** (Nomattribut {char|varchar|number....} [NOT NULL]);

ALTER TABLE Nomtable **DROP COLUMN** Nomattribut ;

ALTER TABLE Nomtable **RENAME COLUMN** Nomattribut **TO** NouveauNom ;

ALTER TABLE Nomtable **ADD** [CONSTRAINT Nomcontrainte] typecontrainte;

ALTER TABLE Nomtable **DROP CONSTRAINT** Nomcontrainte ;

Suppression de tables :

DROP TABLE Nom_table;

MANIPULATION DES DONNEES

INSERT, DELETE, UPDATE

Insertion de lignes dans une table :

INSERT INTO Nom_de_table [(Nom1,Nomn)]
 {VALUES (valeur1,..., valeurn)|requête};

Suppression de lignes :

DELETE FROM Nom_table [WHERE condition];

Modification de valeurs contenues dans des lignes :

UPDATE Nom_table **SET** col1=expr1, col2=expr2, ..[WHERE condition];

UPDATE Nom_table **SET** (col1, col2,...)=(requête) [WHERE condition];

Modification ou Insertion de lignes dans une table : MERGE

MERGE INTO Nom_table

USING {Nom_tab_source | Nom_Vue | Requete} [Alias] **ON** Condition
WHEN MATCHED THEN

UPDATE SET Nom_Col1 = expression1 [, Nom_Col2 = expression2]

WHEN NOT MATCHED THEN

INSERT (Nom_Col1 [, Nom_Col2]....)

VALUES (Expression1 [, Expression2]....);

APPLICATION :

Une base de données contenant des informations sur les productions cinématographiques a été créée. L'utilisateur « tpbd » qui est propriétaire des tables de cette base a donné l'autorisation d'accès à ces tables pour tous les utilisateurs.

QUESTION 1 : Consultez la structure et le contenu de ces tables à partir de votre compte utilisateur.

QUESTION 2 : A partir de cette base vous devez créer sous votre compte la base correspondant au schéma relationnel ci-dessous (vu en TD) :

CATEGORIE (COD_CAT, NOM_CAT)

PRODUCT_CINE (COD_PROD, COD_CAT, TITRE, DUREE, DATE_SORTIE, BUDGET)

→ **CATEGORIE**

FILM_ANIM (COD_PROD, COD_CAT, TITRE, DUREE, DATE_SORTIE, BUDGET, SOC_ANIM, TYPE_ANIM)

→ **CATEGORIE**

→ **PRODUCT_CINE**

FILMS (COD_PROD, COD_CAT, TITRE, DUREE, DATE_SORTIE, BUDGET, DATE_DEB_TOUR, DATE_FIN_TOUR)

→ **CATEGORIE**

→ **PRODUCT_CINE**

PERSONNAGES (COD_PERS, NOM_PERSO)

SAGAS (COD_SAGA, NOM_SAGA)

ACT_REAL (COD_AR, NOM_PERS, PRENOM_PERS, DAT_NAISS, DATE_DEC, SEXE, NATIONALITE)

REALISE (COD_PROD, COD_AR)

→ **ACT_REAL**

→ **PRODUCT_CINE**

JOUEUR (COD_PROD, COD_AR, COD_PERS)

→ **FILMS**

→ **ACT_REAL**

→ **PERSONNAGES**

CONTIENT (COD_PROD, COD_PERS)

→ **FILM_ANIM**

→ **PERSONNAGES**

EPISODES (COD_PROD, COD_SAGA, NUM_EPISODE)

→ **PRODUCT_CINE**

→ **SAGAS**

Pour cela vous devrez reprendre (lorsque ceci est possible) les tables existantes et les données déjà contenues dans la base :

Les tables **PRODUCT_CINE**, **FILM_ANIM**, **FILMS** seront directement créées avec les commandes de création de table. Les données de la table **FILMS** seront ensuite rapatriées dans les tables correspondantes.

La table **PERSONNAGES** sera créée à partir de la table **JOUEUR**, la clé primaire sera constituée par un numéro séquentiel (de 1 à N) attribué lors de l'insertion d'une ligne dans la table **PERSONNAGES**.

La table **JOUEUR** est ensuite restructurée afin de remplacer le nom du personnage par son identifiant (clé primaire de la table **PERSONNAGES**).

La table **ACT_REAL** sera créée à partir des tables **ACTEURS** et **REALISATEURS**.