BD1-TP 2: Contraintes d'intégrité (CI)

L'objectif de ce TP est d'appliquer concrètement les notions théoriques vues en cours et TD sur les contraintes d'intégrités.

PARTIE 1

Considérons la base de données (Fournisseurs, Produits, commandes) du TP1.

Ajout des contraintes d'intégrité

On souhaite modifier la base pour garantir le fait que :

- deux fournisseurs différents ne peuvent pas avoir le même nom,
- deux commandes différentes ne peuvent pas avoir le même numéro,
- deux produits différents ne peuvent pas avoir la même appellation et le même fournisseur à la fois
- un produit fait forcément référence à un fournisseur existant
- une commande fait forcément référence à un produit existant

a) Ajouter les clés primaires

Rappel : ALTER TABLE … ADD CONSTRAINST …PRIMARY KEY ..

i) Modifier la table fournisseur pour ajouter la contrainte de clé primaire sur le nom du fournisseur

ii) Modifier la table produit pour ajouter la contrainte de clé primaire sur le l'appellation et le nom du fournisseur

iii) Modifier la table commande pour ajouter la contrainte de clé primaire sur le numéro de commande

b) Ajouter les clés étrangères

Rappel : ALTER TABLE … ADD CONSTRAINST …FOREIGN KEY ... REFERENCES …

i) Modifier la table produit pour ajouter la contrainte de clé étrangère sur la table fournisseur

ii) Modifier la table commande pour ajouter la contrainte de clé étrangère sur la table produit

c) Vérification

Assurez-vous que les contraintes d'intégrité de clé primaire et de clé étrangère sont bien créées. Pour cela, consultez le dictionnaire pour voir vos contraintes (USER_CONSTRAINTS)

Test des contraintes d'intégrité

1. Faites les tests et répondez aux questions :

a) Test 1

Ajouter le fournisseur : ('BonVin' , 'EARL', 'Bordeaux')
Que se passe t-il ? Pourquoi ?

b) Test 2

Ajouter le fournisseur : ('VinBon' , 'SARL', 'Dijon')
Que se passe t-il ? Pourquoi ?

c) Test 3

Supprimer le fournisseur : ('BonVin' , 'SARL', 'Dijon')
Que se passe t-il ? Pourquoi ?

d) Test 4

Ajouter le produit : ('Bordeaux' , 'Chapoutier', '20')
Que se passe t-il ? Pourquoi ?

e) Test 5

Ajouter le produit : ('Bordeaux' , 'LesVignesDeToto', '15')
Que se passe t-il ? Pourquoi ?

f) Test 6

Ajouter la commande : (2345, 'Florent', 'Cornas', 'Vini')
Que se passe t-il ? Pourquoi ?

2. A partir des observations précédentes, proposez une stratégie pour pouvoir stocker dans la base l'information suivante : La commande 9876 a été passée par le client 'Olivier' qui a acheté 12 bouteilles de Sauterne à 9euros l'unité au fournisseur 'AuxBonnesVignes' qui est une SARL se trouvant à 'Royan'

PARTIE 2

Nettoyage de son espace de travail ou son compte

1° Supprimer les tables et les vues de votre compte, si par mégarde il en restait encore. Vérifier le résultat en utilisant les éléments du dictionnaire de données **USER_TABLES**, **USER_VIEWS** et plus généralement **USER_OBJECTS** définissant respectivement les tables, les vues et les objets de votre compte.

Nombre d'objets accessibles à son compte, notamment ceux du dictionnaire de données et du compte (ou de la base de données) *Cirque*

2° Visualiser le nombre des objets qui vous sont accessibles en utilisant la vue **ALL_OBJECTS**, puis le nombre des objets pour chaque type d'objet (les valeurs de l'attribut OBJECT_TYPE), puis sélectionner le nombre des objets du dictionnaire de données (de propriétaire SYS) et le nombre des objets par type d'objet de ce dictionnaire des données, enfin le nombre des objets par type du compte *Cirque* (ou CIRQUE).

Visualiser les tables et vues de *Cirque* à partir de **ALL_TABLES** et **ALL_VIEWS**.

Lister les contraintes d'intégrité visibles de la base de données *Cirque*

3° Pour cela on utilisera les vues du dictionnaire de données

> **ALL_CONSTRAINTS**(OWNER TABLE_NAME CONSTRAINT_NAME CONSTRAINT_TYPE[1] R_CONSTRAINT_NAME[2] DELETE_RULE[3] SEARCH_CONDITION[4]) et
> **ALL_CONS_COLUMNS**(OWNER TABLE_NAME CONSTRAINT_NAME COLUMN_NAME POSITION[5]).

En quoi l'attribut POSITION de la vue ALL_CONS_COLUMNS est utile ?

4° Lister les contraintes d'intégrité NOT NULL obtenues par la commande "DESC[RIBE] *Table/Vue*". Retrouve-t-on les contraintes affichées dans la question précédente ?

5° On ne voit pas beaucoup de contraintes d'intégrité concernant la vue *Cirque*.***Accessoires*** et les tables *Cirque*.***Accessoires1*** et *Cirque*.***Rangement***. Pourquoi ?

Copie de la base de données *Cirque*

6° Créer dans votre compte les tables ***Personnel***, ***Numéros***, ***Accessoires***, ***Utilisation*** à partir respectivement des tables ou vue ***Personnel***, ***Numéros***, ***Accessoires***, ***Utilisation*** du compte *Cirque*.

7° Que constater sur la vue **USER_CONSTRAINTS**, sachant notamment que les tables *Cirque*.***Accessoires1*** sur *Couleur* et *Cirque*.***Rangement*** sur *NoCamion* avaient des contraintes de type NOT NULL ?

Contraintes d'intégrité de type CHECK implantant des contraintes d'intégrité NOT NULL

[1] Définition du type de contrainte : C pour CHECK, P pour PRIMARY, U pour UNIQUE, R pour une contrainte d'intégrité référentielle.
[2] Nom de la contrainte UNIQUE ou PRIMARY KEY pour la table référencée d'une contrainte d'intégrité référentielle.
[3] Règle de suppression pour une contrainte référentielle : CASCADE ou NO ACTION (voir question 17).
[4] Condition d'une contrainte de type CHECK.
[5] Position de la colonne dans la définition de la table.

8° Essayer de créer une nouvelle contrainte d'intégrité de type NOT NULL sur l'attribut *NoCamion* de votre table ***Accessoires***, puis la contrainte d'intégrité CHECK(NoCamion Is NoT NULL), puis une autre contrainte d'intégrité CHECK(NoCamion Is NoT NULL), enfin la contrainte d'intégrité CHECK(NoRatelier IS NOT NULL). Conclure et supprimer les trois contraintes qui viennent d'être créées.

Reconstruction des contraintes d'intégrité non recopiées

9° Recréer (avec des noms significatifs bien sûr) dans votre schéma, les contraintes d'intégrité trouvées dans la question 3 dans le schéma *Cirque* :

PRIMARY KEY de ***Personnel***, ***Numéros***, ***Utilisation***, et contraintes d'intégrité référentielles *Utilisateur* $\subseteq_1$ *Nom*, *Responsable* $\subseteq_2$ *Nom* et ***Utilisation***.*TitreDeNuméro* $\subseteq_3$ ***Numéros***.*TitreDeNuméro* sans qu'il soit nécessaire d'expliciter les membres droits puisque la PRIMARY KEY est spécifiée par défaut.

Que constate-t-on à partir des vues USER_INDEXES et USER_IND_COLUMNS ?

Création d'une contrainte d'intégrité référentielle supplémentaire

10° Essayer de créer la contrainte d'intégrité référentielle ***Utilisation***.*Accessoire* $\subseteq_4$ ***Accessoires***.*Accessoire*. Créer une clé UNIQUE sur ***Accessoires***.*Accessoire*.

11° Essayer à nouveau de créer la contrainte d'intégrité référentielle$\subseteq_4$.

Implanter avec la clause DISABLE infirmant $\subseteq_4$ (elle pourra être activée ultérieurement par la clause ENABLE). Une autre façon de l'implanter est de remplacer DISABLE par ENABLE NOVALIDATE.

Que conclure à partir des questions 10 et 11 ?

12° Pour activer la contrainte et pour qu'elle soit complètement vérifiée, on opérera ainsi :
d'abord construire pour lister les violations de la contrainte la table ***Exceptions***(ROW_ID ROWID,OWNER VARCHAR2 (30), TABLE_NAME VARCHAR2(30), CONSTRAINT VARCHAR2(30)),
puis utiliser la commande ALTER TABLE ***Utilisation*** ADD CONSTRAINT *rUtilisation_Access* FOREIGN KEY(*Accessoire*) REFERENCES ***Accessoires***(Accessoire) EXCEPTIONS INTO ***Exceptions***,
enfin déduire de la table ***Exceptions*** les lignes de la table référençante violant la contrainte désirée (on n'en trouvera qu'une).

13° Insérer dans ***Accessoires*** la nouvelle ligne (*'étrier'*, NULL, NULL,*0.2*, NULL). Est-ce possible? Insérer dans ***Accessoires*** (*'étrier'*, ' ', NULL, *0.2*, NULL). Est-ce possible? Finalement insérer (*'étrier'*, ' ', NULL, *0.2*, *0)* et créer la contrainte d'intégrité référentielle $\subseteq_4$.