

Rapport TP02

BD1

NAAJI Dorian & ARMANET Nathan – 3A INFO GROUPE 2
POLYTECH LYON
DORIAN.NAAJI@ETU.UNIV-LYON1.FR
NATHAN.ARMANET@ETU.UNIV-LYON1.FR

1. Partie 1 : Ajout des contraintes d'intégrité & tests

1.1. Ajout des clés primaires

On exécute des commandes de la forme *ALTER TABLE ... AD CONSTRAINT ... PRIMARY KEY*

1.1.1. Ajout de clé primaire sur la table FOURNISSEUR

On exécute une commande « *ALTER TABLE FOURNISSEUR ADD CONSTRAINT pk_fournisseur PRIMARY KEY (NOMFOUR);* »

1.1.2. Ajout de la clé primaire sur la table PRODUIT

On exécute une commande « *ALTER TABLE PRODUIT ADD CONSTRAINT pk_produit PRIMARY KEY (NOMFOUR, APPELLATION);* »

1.1.3. Ajout de la clé primaire sur la table COMMANDE

On exécute une commande « *ALTER TABLE COMMANDE ADD CONSTRAINT pk_commande PRIMARY KEY (NOCOM);* »

1.2. Ajout des clés étrangères

On exécute des commandes de la forme *ALTER TABLE ... ADD CONSTRAINT ... FOREIGN KEY ... REFERENCES ...*

1.2.1. Ajout de clé étrangère entre PRODUIT & FOURNISSEUR

On exécute une commande « *ALTER TABLE PRODUIT ADD CONSTRAINT fk_produit FOREIGN KEY (NOMFOUR) REFERENCES FOURNISSEUR(NOMFOUR);* »

1.2.2. Ajout de clé étrangère entre COMMANDE & PRODUIT

On exécute une commande « *ALTER TABLE COMMANDE ADD CONSTRAINT fk_commande FOREIGN KEY (APPELLATION, NOMFOUR) REFERENCES PRODUIT(APPELLATION, NOMFOUR);* »

1.3. Vérification

On effectue une requête de sélection sur le dictionnaire des contraintes utilisateur : « *SELECT * FROM USER_CONSTRAINTS;* »

1.4. Tests des contraintes d'intégrité

1.4.1. Test 1

On exécute la commande : « *INSERT INTO FOURNISSEUR(nomfour, statut, ville) VALUES ('BonVin', 'EARL', 'Bordeaux');* » On remarque que cette commande lève une exception : **ORA-00001: violation de contrainte unique (INI3A03.PK_FOURNISSEUR)**. Cette erreur est liée au fait que **le fournisseur BonVin existe déjà**.

1.4.2. Test 2

On exécute la commande : « *INSERT INTO FOURNISSEUR(nomfour, statut, ville) VALUES ('BonVin', 'EARL', 'Bordeaux');* » On remarque que **l'insertion fonctionne sans aucun problème**.

1.4.3. Test 3

On exécute la commande : « *DELETE FROM FOURNISSEUR WHERE NOMFOUR= 'BonVin' AND STATUT = 'SARL' AND VILLE = 'Dijon';* » On remarque que cette commande de suppression lève une exception : **ORA-02292: violation de contrainte (INI3A03.FK_PRODUIT) d'intégrité - enregistrement fils existant**. Cette exception est générée car **le fournisseur BonVin avait des produits encore enregistrés dans la table PRODUIT**.

1.4.4. Test 4

On exécute la commande : « *INSERT INTO PRODUIT(appellation, nomfour, prix) VALUES ('Bordeaux' 'Chapoutier', '20');* » Cette commande d'insertion lève une exception : **ORA-00001: violation de contrainte unique (INI3A03.PK_PRODUIT)** Cette exception est générée car le **couple de clé primaire (Bordeaux, Chapoutier) existe déjà dans la table PRODUIT**.

1.4.5. Test 5

On exécute la commande « *INSERT INTO PRODUIT(appellation, nomfour, prix) VALUES ('Bordeaux', 'LesVignesDeToto', '15');* » Cette commande lève une exception car le « **nomfour** » **'LesVignesDeToto'** **n'existe pas dans la table FOURNISSEUR**.

1.4.6. Test 6

On exécute la commande « *INSERT INTO COMMANDE(NoCom, Client, Appellation, NomFour) VALUES(2345, 'Florent', 'Cornas', 'Vini');* ». Cette commande lève une exception car **le produit décrit par le couple de clés primaires ('Cornas' 'Vini') n'existe pas dans la table PRODUIT**.

1.5. Stratégie d'insertion de données

Pour insérer les données, il faut entrer les données dans un ordre précis afin de respecter les contraintes d'intégrité des différentes tables. En effet, lors d'une insertion de données, il faut que les données insérées n'aient aucune référence vers un élément qui n'a pas encore été créé. Dès lors, on va d'abord entrer les données correspondant au **FOURNISSEUR**, puis les données du **PRODUIT** et finalement les données de la **COMMANDE**.

2. Partie 2

2.1. Nettoyage de l'espace de travail

On effectue des commandes de type « DROP » pour supprimer les vues ainsi que les tables. Pour les tables possédant des contraintes d'intégrité, on veillera à supprimer en premier la table n'ayant aucune table faisant référence à une colonne de cette dernière qu'on supprime.

On vérifie ensuite cela en consultant les dictionnaires **USER_TABLES**, **USER_VIEWS** et **USER_OBJECTS** :

- *SELECT * FROM USER_TABLES;*
- *SELECT * FROM USER_VIEWS;*
- *SELECT * FROM USER_OBJECTS;*

On remarque que ces derniers sont vides, notre nettoyage a donc été effectif

2.2. Visualisation d'objets accessibles à notre compte

On effectue les différentes requêtes qui nous permettent de visualiser des objets qui nous sont accessibles :

« *SELECT * FROM ALL_OBJECTS* » : on remarque que 9180 objets nous sont accessibles

« *SELECT DISTINCT OBJECT_TYPE, count(*) AS nbObject FROM ALL_OBJECTS GROUP BY OBJECT_TYPE;* »

	OBJECT_TYPE	NBOBJECT
1	EDITION	1
2	CONSUMER GROUP	2
3	SEQUENCE	5
4	SCHEDULE	4
5	PROCEDURE	18
6	OPERATOR	14
7	DESTINATION	2
8	WINDOW	9
9	SCHEDULER GROUP	4
10	PACKAGE	258
11	PROGRAM	1
12	XML SCHEMA	19
13	JOB CLASS	2
14	TABLE	75
15	SYNONYM	5780
16	INDEX	53
17	VIEW	1690
18	FUNCTION	94
19	INDEXTYPE	2
20	TYPE	1146
21	EVALUATION CONTEXT	1

FIGURE 1 : RESULTAT DE LA REQUETE

« *SELECT COUNT(*) as nbTableSys FROM ALL_OBJECTS WHERE OWNER = 'SYS';* »

«SELECT DISTINCT OBJECT_TYPE, count(*) as nbObject FROM ALL_OBJECTS where OWNER = 'SYS' GROUP BY OBJECT_TYPE; »

	OBJECT_TYPE	NBOBJECT
1	EDITION	1
2	CONSUMER GROUP	2
3	SEQUENCE	4
4	SCHEDULE	4
5	PROCEDURE	17
6	OPERATOR	7
7	DESTINATION	2
8	WINDOW	9
9	SCHEDULER GROUP	4
10	PACKAGE	218
11	PROGRAM	1
12	JOB CLASS	2
13	TABLE	35
14	INDEX	13
15	VIEW	1678
16	FUNCTION	89
17	TYPE	1047
18	EVALUATION CONTEXT	1

FIGURE 2 : RESULTAT DE LA REQUETE

« SELECT DISTINCT OBJECT_TYPE, count(*) as nbObject from ALL_OBJECTS WHERE OWNER = 'CIRQUE' GROUP BY OBJECT_TYPE; »

	OBJECT_TYPE	NBOBJECT
1	INDEX	3
2	TABLE	3
3	VIEW	1

FIGURE 3 : RESULTAT DE LA REQUETE

« SELECT * FROM ALL_TABLES WHERE OWNER = 'CIRQUE'; »

« SELECT * FROM ALL_VIEWS WHERE OWNER = 'CIRQUE'; »

2.3. Listage des contraintes d'intégrités de la BD Cirque

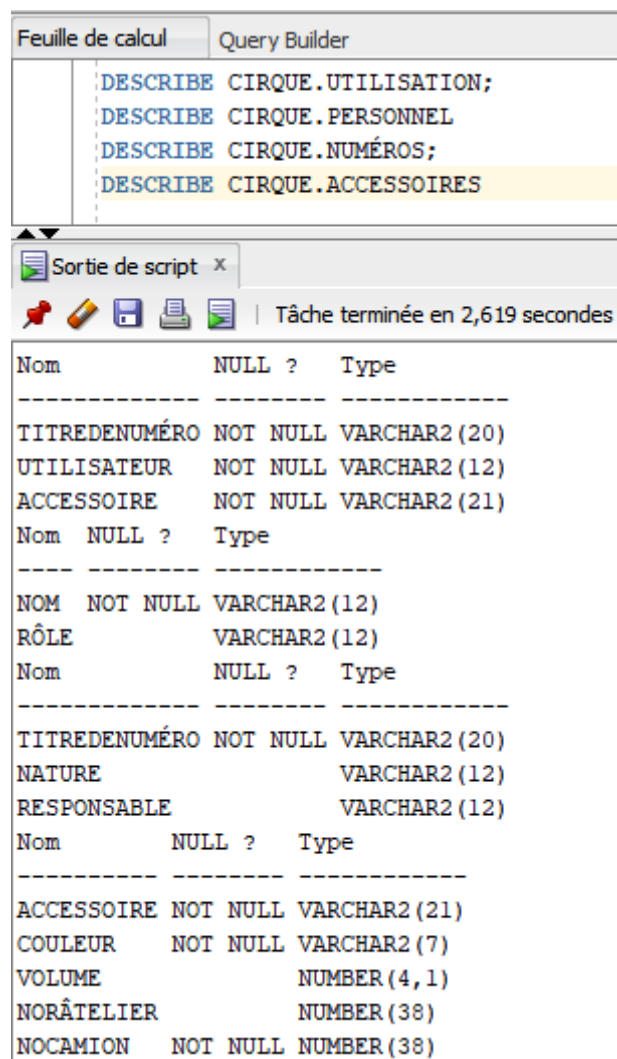
On effectue les requêtes qui permettent de visualiser les contraintes d'intégrités de la BD cirque :

« `select * from ALL_CONSTRAINTS where OWNER = 'CIRQUE';` »

« `select * from ALL_CONS_COLUMNS where OWNER = 'CIRQUE';` »

L'attribut POSITION de la vue ALL_CONS_COLUMNS est utile car il permet de savoir dans quel ordre ont été créés les contraintes. Dans le cas d'une insertion de données, cela nous aidera alors grandement à savoir dans quel ordre insérer les données, comme par exemple dans le cas d'une question précédente où on nous demandait d'élaborer une stratégie pour pouvoir stocker des informations dans la base de données

2.4. Listage des contraintes d'intégrité NOT NULL



Nom	NULL ?	Type

TITREDENUMÉRO	NOT NULL	VARCHAR2(20)
UTILISATEUR	NOT NULL	VARCHAR2(12)
ACCESSOIRE	NOT NULL	VARCHAR2(21)

Nom	NULL ?	Type

NOM	NOT NULL	VARCHAR2(12)
RÔLE		VARCHAR2(12)

Nom	NULL ?	Type

TITREDENUMÉRO	NOT NULL	VARCHAR2(20)
NATURE		VARCHAR2(12)
RESPONSABLE		VARCHAR2(12)

Nom	NULL ?	Type

ACCESSOIRE	NOT NULL	VARCHAR2(21)
COULEUR	NOT NULL	VARCHAR2(7)
VOLUME		NUMBER(4,1)
NORÂTELIER		NUMBER(38)
NOCAMION	NOT NULL	NUMBER(38)

FIGURE 4 : INSERTION DES CI "NOT NULL"

2.5. On ne voit pas beaucoup de CI concernant la vue CirqueAccessoires et les tables CirqueAccessoire1 et CirqueRangement. Pourquoi ?

Car les vues sont des résultats de requêtes qui s'actualisent automatiquement.

2.6. Création des tables PERSONNEL, NUMÉROS, ACCESSOIRES, UTILISATION à partir des tables ou vues du compte CIRQUE.

On effectue les requêtes suivantes :

- « *create table Personnel as (SELECT * FROM CIRQUE.Personnel);* »
- « *create table Numeros as (SELECT * FROM CIRQUE.Numeros);* »
- « *create table Utilisation as (SELECT * FROM CIRQUE.Utilisation);* »
- « *create table Accessoires as (SELECT * FROM CIRQUE.ACCESSOIRES);* »

2.7. Que constater sur USER_CONSTRAINTS ?

Les contraintes d'intégrité (PRIMARY et FOREIGN KEY) n'ont pas été dupliquées sur nos tables. En revanche, les attributs de type NOT NULL ont été dupliqués sur nos tables.

2.8. Création des contraintes et tests

« *alter table ACCESSOIRES modify NOCAMION not null;* »

Cette commande génère une exception : **ORA-01442: colonne à modifier en NOT NULL est déjà NOT NULL.**
On ne peut en effet pas ajouter cette contrainte sur l'attribut car elle existe déjà.

« *alter table ACCESSOIRES add check (NOCAMION is not null);* »

Cette commande fonctionne.

« *alter table ACCESSOIRES add check (NOCAMION is not null);* »

Cette commande fonctionne encore.

« *alter table ACCESSOIRES add check (NORÂTELIER is not null);* »

Cette commande fonctionne.

On peut en **conclure** que l'on peut créer plusieurs CHECK sur une table même s'ils sont identiques !
En revanche, on ne peut pas ajouter deux fois la même contrainte sur un attribut.

Finalement, on supprime les contraintes :

« *alter table ACCESSOIRES drop constraint nomDeLaContrainte;* »

2.9. Reconstruction des CI

- « alter table UTILISATION add constraint fk_utilisateur foreign key (UTILISATEUR) references PERSONNEL(NOM); »
- « alter table UTILISATION add constraint fk_titreNum foreign key (TITREDENUMÉRO) references NUMEROS(TITREDENUMÉRO); »
- « alter table NUMEROS add constraint fk_responsable foreign key (RESPONSABLE) REFERENCES PERSONNEL (NOM); »
- « alter table NUMEROS add constraint pk_numero primary key (TITREDENUMÉRO); »
- « select * from USER_INDEXES; »
(Permet d'afficher les 3 clés primaires créées)
- « select * from USER_IND_COLUMNS; »
(Affiche les clés primaires créées avec tous les détails)

2.10. Création de la contrainte d'intégrité référentielle

Il est impossible de créer cette contrainte d'intégrité référentielle. La commande :
« ALTER TABLE UTILISATION ADD CONSTRAINT fk_UtilAccess FOREIGN KEY (ACCESSOIRE) REFERENCES ACCESSOIRES(ACCESSOIRE); » génère une exception :

ORA-02270: pas de correspondance de clé primaire ou unique pour cette liste de colonnes 02270)

Il est ensuite demandé de créer une clé UNIQUE sur Accessoires.Accessoire, ce qui est impossible car il y a des doublons dans les données :

« alter table ACCESSOIRES add constraint uk_accessoires unique (ACCESSOIRE); »

ORA-02299: impossible de valider (INI3A03.UK_ACCESSOIRES) - clés en double trouvées

2.11. Nouvel essai de création de la contrainte d'intégrité référentielle

Ce qui est demandé est impossible car on ne peut pas passer ACCESSOIRES.ACCESSOIRE en clé unique. Impossible donc de créer à nouveau la contrainte et de l'implanter avec la clause DISABLE.

2.12. Activation de la contrainte

On crée la table EXCEPTION comme demandé :

```
« create table EXCEPTIONS
(
  ROW_ID    NUMBER generated as identity
  constraint EXCEPTIONS_PK
  primary key,
  OWNER     VARCHAR2(30),
  TABLE_NAME VARCHAR2(30),
  CONSTRAINT VARCHAR2(30)
) »
```

Néanmoins, la commande ALTER TABLE qui doit être exécutée dans la suite du sujet ne peut pas fonctionner car nous n'avons toujours pas pu ajouter de clé primaire dans la table Accessoires, cette dernière contenant en effet des doublons.

```
« ALTER TABLE Utilisation ADD CONSTRAINT rUtilisation_Access FOREIGN KEY(Accessoire) REFERENCES
Accessoires(Accessoire) EXCEPTIONS INTO Exceptions »
```

2.13. Insertion de nouvelle ligne.

On essaye d'insérer la 1^{ère} ligne : « *insert into ACCESSOIRES(accessoire, couleur, volume, norâtelier, nocamion) values ('étrier', NULL, NULL, 0.2, NULL);* ». C'est impossible car on essaye d'insérer des données NULL dans une colonne NOT NULL. En l'occurrence, la colonne couleur. Il est donc impossible d'insérer la valeur « NULL » dans la colonne « COULEUR ».

On essaye d'insérer la 2^{ème} ligne : « *insert into ACCESSOIRES(accessoire, couleur, volume, norâtelier, nocamion) values ('étrier', ' ', NULL, 0.2, NULL)* ». C'est impossible car on essaye d'insérer des données NULL dans une colonne NOT NULL. En l'occurrence, la colonne noCamion. Il est donc impossible d'insérer la valeur « NULL » dans la colonne « NOCAMION ».

On essaye d'insérer la 3^{ème} ligne : « *insert into ACCESSOIRES(accessoire, couleur, volume, norâtelier, nocamion) values ('étrier', ' ', NULL, 0.2, 0);* ». Cela fonctionne.

On essaye de créer la contrainte d'intégrité référentielle C4, cela ne fonctionne pas toujours pas à cause des doublons de la table.