

# Rapport du TP n°1

Rougab Imene  
M1 SII Groupe 01  
Année scolaire : 2016/2017

## I. Création des TableSpaces et des utilisateurs

### 1- Création des TableSpaces :

```
SQL> CREATE TABLESPACE TP1_tbs DATAFILE 'C:\Users\im\Desktop\M1 SII\ASGBD\TP\tbs_tp1.dat' SIZE 100M AUTOEXTEND ON ONLINE;  
Tablespace cr  .
```

```
SQL> CREATE TEMPORARY TABLESPACE TP1_TempTbs TEMPFILE 'C:\Users\im\Desktop\M1 SII\ASGBD\TP\TempTbs_tp1.dat' SIZE 100M AUTOEXTEND ON;  
Tablespace cr  .
```

V  rification de la cr  ation des deux fichiers :

M1 SII > ASGBD > TP				
	Nom	Modifi�� le	Type	Taille
	TBS_TP1.DAT	03/10/2016 18:34	Fichier DAT	102 408 Ko
	TEMPTBS_TP1.DAT	03/10/2016 18:35	Fichier DAT	102 408 Ko

### 2- Cr  ation d'un nouvel utilisateur :

```
SQL> CREATE USER RougabImene IDENTIFIED BY IMENE DEFAULT TABLESPACE TP1_tbs TEMPORARY TABLESPACE TP1_TempTbs;  
Utilisateur cr  .
```

### 3- Attribution des droits au nouvel utilisateur :

```
SQL> GRANT ALL PRIVILEGES TO RougabImene;  
Autorisation de privil  ges (GRANT) accept  e.
```

On se connecte avec le nouvel utilisateur :

```
Entrez le nom utilisateur : RougabImene  
Entrez le mot de passe :  
  
Connect    :  
Oracle Database 10g Express Edition Release 10.2.0.1.0 - Production  
  
SQL>
```

## II. Langage de définition des données

### 4- Création des relations de la base avec toutes les contraintes :

#### 1. Création de la relation 'Hôtel' :

La relation 'Hôtel' est composée de 4 attributs parmi lesquels on a l'attribut '**NumHotel**' qui représente la clé primaire.

Le nombre d'étoiles doit être compris entre 1 et 5. Pour cela on utilise une contrainte de type 'Check' qui peut être définie de plusieurs façons :

Ex :

CONSTRAINT check\_etoiles CHECK (Etoiles BETWEEN 1 and 5)

CONSTRAINT check\_etoiles CHECK (Etoiles>0 AND Etoiles<6)

CONSTRAINT check\_etoiles CHECK (Etoiles>=1 AND Etoiles <=5)

Requête :

```
SQL> create table HOTEL (  
2 NumHotel Number(3),  
3 Nom varchar(20),  
4 Ville varchar(20),  
5 Etoiles number(1),  
6 constraint pk_hotel primary key (NumHotel),  
7 constraint check_etoiles check (Etoiles>0 AND Etoiles<6)  
8 );
```

Table créée.

Résultat :

```
SQL> Desc HOTEL;  
Nom  
-----  
NUMHOTEL          NOT NULL  NUMBER(3)  
NOM                VCHAR2(20)  
VILLE             VCHAR2(20)  
ETOILES            NUMBER(1)
```

#### 2. Création de la relation 'CHAMBRE'

La relation 'CHAMBRE' est composée de 5 attributs, et dont la clé primaire est composée de 2 attributs : l'attribut '**NumChambre**' ainsi que l'attribut '**NumHotel**' qui représente une clé étrangère référencée à la relation 'Hotel'.

Le type de la chambre doit appartenir à la liste { 'simple', 'double', 'triple', 'suite', 'autre'}, pour ce fait on définit une contrainte de type check qui vérifie que lors de l'insertion ou la modification que la valeur donnée à l'attribut 'TypeChambre' appartient bien à la liste imposée.

### Requête :

```
SQL> create table CHAMBRE (
  2  NumChambre Number(3),
  3  NumHotel number(3),
  4  Etage number(3),
  5  TypeChambre varchar(10),
  6  PrixNuit number(5),
  7  constraint pk_chambre primary key (NumHotel,NumChambre),
  8  constraint fk_hotel foreign key (NumHotel) references HOTEL(NumHotel) on delete cascade,
  9  constraint check_TypeChambre check (TypeChambre IN ('simple','double','triple','suite','autre'))
  10 );
```

Table cr   e.

### R  sultat :

```
SQL> Desc CHAMBRE;
Nom                                NULL ?    Type
-----
NUMCHAMBRE                        NOT NULL  NUMBER(3)
NUMHOTEL                          NOT NULL  NUMBER(3)
ETAGE                             NULL      NUMBER(3)
TYPECHAMBRE                       NULL      VARCHAR2(10)
PRIXNUIT                          NULL      NUMBER(5)
```

## 3. Cr  ation de la relation 'CLIENT'

La relation 'CLIENT' est compos   de 3 attributs dont '**NumClient**' repr  sente la cl   primaire.

### Requête :

```
SQL> create table CLIENT(
  2  NumClient number(5),
  3  Nom varchar(20),
  4  Prenom varchar(20),
  5  constraint prk_client primary key (NumClient)
  6 );
```

Table cr   e.

### R  sultat :

```
SQL> Desc CLIENT;
Nom                                NULL ?    Type
-----
NUMCLIENT                        NOT NULL  NUMBER(5)
NOM                              NULL      VARCHAR2(20)
PRENOM                           NULL      VARCHAR2(20)
```

#### 4. Création de la relation 'RESERVATION'

La relation 'RESERVATION' est composée de 5 attributs. Sa clé primaire est composée de 3 attributs.

Elle possède aussi 3 clés étrangères (dont deux font partie de sa clé primaire) qui sont :

- . 'NumClient' qui représente la clé primaire de la relation 'CLIENT'.
- . 'NumHotel' et 'NumChambre' qui représentent la clé primaire de la relation 'CHAMBRE'.

La date d'arrivée doit être antérieure à la date de départ. On définit une contrainte de type CHECK pour vérifier cela à chaque insertion ou modification.

#### Requête :

```
SQL> create table RESERVATION (  
2  NumClient number(5),  
3  NumHotel number(3),  
4  DateArrivee date,  
5  DateDepart date,  
6  NumChambre number(3),  
7  constraint pk_reservation primary key(NumHotel, NumClient, DateArrivee),  
8  constraint fk_chambre foreign key (NumHotel,NumChambre) references CHAMBRE(NumHotel,NumChambre) on delete cascade,  
9  constraint fk_client foreign key (NumClient) references CLIENT(NumClient) on delete cascade,  
10 constraint check_date check (DateArrivee<DateDepart)  
11 );
```

Table créée.

#### Résultat :

```
SQL> Desc RESERVATION;
```

Nom	NULL ?	Type
NUMCLIENT	NOT NULL	NUMBER(5)
NUMHOTEL	NOT NULL	NUMBER(3)
DATEARRIVEE	NOT NULL	DATE
DATEDEPART		DATE
NUMCHAMBRE		NUMBER(3)

5- Ajout de l'attribut 'ADRESSECLIENT' de type chaîne de caractère à la relation 'CLIENT' :

Requête :

```
SQL> Alter table CLIENT ADD AdresseClient varchar(50);
Table modifiée.
```

Résultat :

```
SQL> desc CLIENT;
Nom                               NULL ?   Type
-----
NUMCLIENT                        NOT NULL NUMBER(5)
NOM                               VCHAR2(20)
PRENOM                           VCHAR2(20)
ADRESSECLIENT                    VCHAR2(50)
```

6- Ajout de la contrainte 'NOT NULL' aux attributs 'ETOILES' et 'TYPECHAMBRE' :

Requêtes :

```
SQL> Alter table HOTEL MODIFY Etoiles number(1) NOT NULL;
Table modifiée.
```

```
SQL> Alter table CHAMBRE MODIFY TypeChambre varchar(10) NOT NULL;
Table modifiée.
```

Résultats :

```
SQL> desc HOTEL;
Nom                               NULL ?   Type
-----
NUMHOTEL                         NOT NULL NUMBER(3)
NOM                               VCHAR2(20)
VILLE                           VCHAR2(20)
ETOILES                          NOT NULL NUMBER(1)
```

```
SQL> desc CHAMBRE
Nom                               NULL ?   Type
-----
NUMCHAMBRE                       NOT NULL NUMBER(3)
NUMHOTEL                         NOT NULL NUMBER(3)
ETAGE                            NUMBER(3)
TYPECHAMBRE                      NOT NULL VCHAR2(10)
PRIXNUIT                         NUMBER(5)
```



#### 7- Modification de la longueur de l'attribut 'VILLE' :

Pour modifier la longueur d'un attribut il suffit d'utiliser la requête :

'ALTER TABLE Nom\_table MODIFY Nom\_attribut TYPE(TAILLE)'

En modifiant TAILLE selon le besoin.

. Agrandir : on augmente la longueur de l'attribut 'VILLE' à 50 au lieu de 20

```
SQL> Alter table HOTEL MODIFY Ville varchar(50);
Table modifiée.
SQL> desc HOTEL;
```

Nom	NULL ?	Type
NUMHOTEL	NOT NULL	NUMBER(3)
NOM		VARCHAR2(20)
VILLE		VARCHAR2(50)
ETOILES	NOT NULL	NUMBER(1)

. Réduire : on réduit la longueur augmentée à 20

```
SQL> Alter table HOTEL MODIFY Ville varchar(20);
Table modifiée.
SQL> desc HOTEL;
```

Nom	NULL ?	Type
NUMHOTEL	NOT NULL	NUMBER(3)
NOM		VARCHAR2(20)
VILLE		VARCHAR2(20)
ETOILES	NOT NULL	NUMBER(1)

#### 8- Suppression de la colonne 'PRIXNUIT' de la table 'CHAMBRE' :

Requête :

```
SQL> Alter table CHAMBRE DROP COLUMN PrixNuit;
Table modifiée.
```

Résultat : La colonne 'PRIXNUIT' a été supprimée

```
SQL> desc CHAMBRE;
```

Nom	NULL ?	Type
NUMCHAMBRE	NOT NULL	NUMBER(3)
NUMHOTEL	NOT NULL	NUMBER(3)
ETAGE		NUMBER(3)
TYPECHAMBRE	NOT NULL	VARCHAR2(10)

On recrée la colonne 'PRIXNUIT' :

```
SQL> Alter table CHAMBRE ADD PrixNuit number(5);
Table modifiée.
SQL> desc CHAMBRE;
```

Nom	NULL ?	Type
NUMCHAMBRE	NOT NULL	NUMBER(3)
NUMHOTEL	NOT NULL	NUMBER(3)
ETAGE		NUMBER(3)
TYPECHAMBRE	NOT NULL	VARCHAR2(10)
PRIXNUIT		NUMBER(5)

9- Renommage de l'attribut 'ETAGES' dans la relation 'CHAMBRE' par 'ETAGECHAMBRE' :

Requête :

```
SQL> Alter table CHAMBRE rename column Etage to EtageChambre;  
Table modifiée.
```

Résultat :

```
SQL> desc CHAMBRE;  
Nom                                NULL ?    Type  
-----  
NUMCHAMBRE                        NOT NULL  NUMBER(3)  
NUMHOTEL                          NOT NULL  NUMBER(3)  
ETAGECHAMBRE                      NUMBER(3)  
TYPECHAMBRE                      NOT NULL  VARCHAR2(10)  
PRIXNUIT                          NUMBER(5)
```

10- Ajout de la contrainte : 2000 DA < PRIXNUIT < 20000 DA :

```
SQL> Alter table CHAMBRE add constraint check_prix check( PrixNuit>2000 and PrixNuit<20000);  
Table modifiée.
```



### III. Langage des manipulations des données

#### 11- Insertion des données :

Etant donné le grand nombre de données à insérer, on va prendre comme exemple de l'insertion d'une seule donnée pour chaque table

**Les requêtes :**

Insert into HOTEL values (1,'Renaissance','Tlemcen', 5);

Insert into CLIENT values (1,'BOUROUBI','Taous', NULL);

Insert into CHAMBRE values (1, 2, 1, 'simple', 4500);

Insert into RESERVATION values (1, 5, to\_date ('2016-05-11','yyyy-mm-dd'), to\_date ('2016-05-05','yyyy-mm-dd'), 1);

#### 12- Si L'hôtel 'EL Mountazeh Annaba' change de classement et passe à 5 étoiles il faut mettre à jour le nombre d'étoile de ce dernier dans notre base de données, et pour cela on utilise la requête 'UPDATE Nom\_Table'

```
SQL> update hotel set Etoiles=5 where nom='El Mountazah Annaba';  
1 ligne mise à jour.
```

#### 13- Augmentation des prix des chambres 4 étoiles de 1000 DA :

- On désactive la contrainte imposée sur l'attribut 'PRIXNUIT' :

**Requête :** ALTER TABLE CHAMBRE DISABLE CONSTRAINT check\_prix;

- On ajoute 1000 DA au prix des chambres de 4 étoiles :

```
SQL> update chambre set prixnuit = prixnuit+1000 where numhotel in (select numhotel from hotel where etoiles=4);  
16 ligne(s) mise(s) à jour.
```

- On réactive la contrainte :

**Requête :** ALTER TABLE CHAMBRE ENABLE CONSTRAINT check\_prix ;

#### 14- Suppression des chambres de l'hôtel 'Renaissance' :

```
SQL> delete from chambre where numhotel in (select numhotel from hotel where nom='Renaissance');  
8 ligne(s) supprimé(s).
```

On ne rencontre aucun problème et cela est dû au fait d'avoir ajouté la contrainte 'ON DELETE CASCADE' lors de la création des relations. En l'absence de cette contrainte, on ne peut pas supprimer les tuples d'une table dont la clé primaire est clé étrangère dans une autre table.

Ici la clé primaire de la relation 'chambre', 'NUMCHAMBRE', est clé étrangère dans la relation 'RESERVATION', et si on n'avait pas imposé la contrainte 'ON DELETE CASCADE' lors de la création des tables, on n'aurait pas pu supprimer les tuples de la table chambre avant de les supprimer de la table RESERVATION.

#### IV. Langage d'interrogation de données

##### 15- Liste des hôtels et de leurs villes :

```
SQL> Select nom, ville from HOTEL;
```

NOM	VILLE
Ibis Alger	Alger
Renaissance	Tlemcen
Seybouse	Annaba
Hôtel Novotel	Constantine
Saint George d'Alger	Alger
El Mountazah Annaba	Annaba
Hôtel Albert 1er	Alger
Chems	Oran
Colombe	Oran
Mercure	Alger
Le Méridien	Oran
Hôtel Sofitel	Alger

12 ligne(s) sélectionnée(s).

##### 16- Liste des hôtels sur lesquels porte au moins une réservation :

```
SQL> Select distinct Nom from HOTEL h, RESERVATION r where h.NUMHOTEL=r.NUMHOTEL;
```

NOM
Ibis Alger
Renaissance
Le Méridien
Saint George d'Alger
Mercure
Hôtel Sofitel
Seybouse
Hôtel Albert 1er
Chems
Colombe
El Mountazah Annaba

11 ligne(s) sélectionnée(s).

17- Liste des clients qui ont toujours séjournés au premier étage :

```
SQL> select NOM from CLIENT c1 where
2   c1.NUMCLIENT not in (
3     select NUMCLIENT from
4       RESERVATION r, CHAMBRE c where
5       r.NUMCHAMBRE = c.NUMCHAMBRE
6       and c.ETAGECHAMBRE <> 1 )
7     MINUS
8     select NOM from CLIENT c1 where
9     c1.NUMCLIENT not in (
10    select NUMCLIENT from
11      RESERVATION r where
12      r.NUMCLIENT = c1.NUMCLIENT);
```

NOM  
 -----  
 ADDAD  
 AROUEL

18- Liste des hôtels qui offrent des suites avec le prix de chaque suite :

```
SQL> Select nom, ville, numchambre, prixnuit
2   from HOTEL h, CHAMBRE c
3   where h.numhotel= c.numhotel
4   and c.typechambre='suite';
```

NOM	VILLE	NUMCHAMBRE	PRIXNUIT
-----	-----	-----	-----
Seybouse	Annaba	9	16000
Seybouse	Annaba	10	16500
Hôtel Sofitel	Alger	301	19500
Hôtel Sofitel	Alger	302	19500
Hôtel Sofitel	Alger	303	19500