Université des Sciences et de la Technologie Houari Boumediene

Architectures et Systèmes des Base de Données

Rapport de Travaux Pratiques N°3
Dictionnaire Oracle

Binôme
MOHAMMEDI Haroune
HOUACINE Naila Aziza

Professeur
Pr. BOUKHALFA Kamel

21 octobre 2017

1 Connexion en tant que « System ». Lister le catalogue « DICT » et affichage du nombre d'instances qu'il contient ainsi que sa structure.

1.1 Requête

```
CONNECT SYSTEM/oracle
SELECT COUNT(*) AS NBR FROM DICT;
DESC DICT;
```

1.2 Résultat

```
       NBR
----------
      2551

Name                                      Null?    Type
----------------------------------------- -------- ------------------
TABLE_NAME                                         VARCHAR2(30)
COMMENTS                                           VARCHAR2(4000)
```

On remarque que la table «DICT» est composée de deux (2) colonnes seulement qui sont le nom des tables qui est une chaine de caractère de taille = 30 et de leur description qui est aussi une chaine de caractère de taille = 4000.

2 Rôle et structure des tables (ou vues) suivantes : ALL_TAB_COLUMNS, USER_USERS, ALL_CONSTRAINTS et USER_TAB_PRIVS

Afin de connaitre le rôle d'une table on consulte le dictionnaire DICT, plus précisément la colonne COMMENTS

2.1 Requête

```
SELECT *
FROM DICT
WHERE TABLE\_NAME IN ('ALL\_TAB\_COLUMNS','USER\_USERS', 'ALL\_CONSTRAINTS', 'USER\_TAB\
    ↪ _PRIVS');
```

2.2 Résultat

```
TABLE_NAME                     COMMENTS
------------------------------ --------------------------------------------------
ALL_CONSTRAINTS                Constraint definitions on accessible tables
ALL_TAB_COLUMNS                Columns of user s tables, views and clusters
USER_TAB_PRIVS                 Grants on objects for which the user is the owner
USER_USERS                     Information about the current user
```

Puis la description de la structure de chaque table se fait via la commande DESC.

2.3 Requête

```
DESC ALL_TAB_COLUMNS;
DESC USER_USERS;
DESC ALL_CONSTRAINTS;
DESC USER_TAB_PRIVS;
```

2.4 Résultat

```
SQL> DESC ALL_TAB_COLUMNS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 OWNER                                     NOT NULL VARCHAR2(30)
 TABLE_NAME                                NOT NULL VARCHAR2(30)
 COLUMN_NAME                               NOT NULL VARCHAR2(30)
 DATA_TYPE                                          VARCHAR2(106)
 DATA_TYPE_MOD                                      VARCHAR2(3)
 DATA_TYPE_OWNER                                    VARCHAR2(120)
 DATA_LENGTH                               NOT NULL NUMBER
 DATA_PRECISION                                     NUMBER
 DATA_SCALE                                         NUMBER
 NULLABLE                                           VARCHAR2(1)
 COLUMN_ID                                          NUMBER
 DEFAULT_LENGTH                                     NUMBER
 DATA_DEFAULT                                       LONG
 NUM_DISTINCT                                       NUMBER
 LOW_VALUE                                          RAW(32)
 HIGH_VALUE                                         RAW(32)
 DENSITY                                            NUMBER
 NUM_NULLS                                          NUMBER
 NUM_BUCKETS                                        NUMBER
 LAST_ANALYZED                                      DATE
 SAMPLE_SIZE                                        NUMBER
 CHARACTER_SET_NAME                                 VARCHAR2(44)
 CHAR_COL_DECL_LENGTH                               NUMBER
 GLOBAL_STATS                                       VARCHAR2(3)
 USER_STATS                                         VARCHAR2(3)
 AVG_COL_LEN                                        NUMBER
 CHAR_LENGTH                                        NUMBER
 CHAR_USED                                          VARCHAR2(1)
 V80_FMT_IMAGE                                      VARCHAR2(3)
 DATA_UPGRADED                                      VARCHAR2(3)
 HISTOGRAM                                          VARCHAR2(15)

SQL> DESC USER_USERS;
 Name                                      Null?    Type
 ----------------------------------------- -------- -----------------
 USERNAME                                  NOT NULL VARCHAR2(30)
 USER_ID                                   NOT NULL NUMBER
 ACCOUNT_STATUS                            NOT NULL VARCHAR2(32)
 LOCK_DATE                                          DATE
 EXPIRY_DATE                                        DATE
 DEFAULT_TABLESPACE                        NOT NULL VARCHAR2(30)
 TEMPORARY_TABLESPACE                      NOT NULL VARCHAR2(30)
 CREATED                                   NOT NULL DATE
 INITIAL_RSRC_CONSUMER_GROUP                        VARCHAR2(30)
 EXTERNAL_NAME                                      VARCHAR2(4000)


SQL> DESC ALL_CONSTRAINTS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 OWNER                                              VARCHAR2(120)
 CONSTRAINT_NAME                           NOT NULL VARCHAR2(30)
 CONSTRAINT_TYPE                                    VARCHAR2(1)
 TABLE_NAME                                NOT NULL VARCHAR2(30)
```

```
 SEARCH_CONDITION                                      LONG
 R_OWNER                                               VARCHAR2(120)
 R_CONSTRAINT_NAME                                     VARCHAR2(30)
 DELETE_RULE                                           VARCHAR2(9)
 STATUS                                                VARCHAR2(8)
 DEFERRABLE                                            VARCHAR2(14)
 DEFERRED                                              VARCHAR2(9)
 VALIDATED                                             VARCHAR2(13)
 GENERATED                                             VARCHAR2(14)
 BAD                                                   VARCHAR2(3)
 RELY                                                  VARCHAR2(4)
 LAST_CHANGE                                           DATE
 INDEX_OWNER                                           VARCHAR2(30)
 INDEX_NAME                                            VARCHAR2(30)
 INVALID                                               VARCHAR2(7)
 VIEW_RELATED                                          VARCHAR2(14)


SQL> DESC USER_TAB_PRIVS;
 Name                                      Null?    Type
 ----------------------------------------- -------- --------------------
 GRANTEE                                   NOT NULL VARCHAR2(30)
 OWNER                                     NOT NULL VARCHAR2(30)
 TABLE_NAME                                NOT NULL VARCHAR2(30)
 GRANTOR                                   NOT NULL VARCHAR2(30)
 PRIVILEGE                                 NOT NULL VARCHAR2(40)
 GRANTABLE                                          VARCHAR2(3)
 HIERARCHY                                          VARCHAR2(3)
```

3 Afficher le nom d'utilisateur avec lequel nous sommes connectés

Cette information peut être récupérée à partir de la table `USER_USERS` dont nous avons précédemment affiché la structure ; Ainsi nous constatons que la colonne qui nous intéresse est `USERNAME`, se qui nous permet de l'afficher maintenant :

3.1 Requête

```
SELECT USERNAME FROM USER_USERS;
```

3.2 Résultat

```
 USERNAME
 ------------------------------
 SYSTEM
```

Effectivement nous sommes connectés en tant que `SYSTEM`.

4 Comparaison de la structure et le contenu des tables `ALL_TAB_COLUMNS` et `USER_TAB_COLUMNS`

4.1 Requête

```
DESC ALL_TAB_COLUMNS;
DESC USER_TAB_COLUMNS;

SELECT COUNT(*) AS NBR_ALL_TAB_COLUMNS
FROM ALL_TAB_COLUMNS;

SELECT COUNT(*) AS NBR_USER_TAB_COLUMNS
FROM USER_TAB_COLUMNS;
```

4.2 Résultat

```
SQL> DESC ALL_TAB_COLUMNS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 OWNER                                     NOT NULL VARCHAR2(30)
 TABLE_NAME                                NOT NULL VARCHAR2(30)
 COLUMN_NAME                               NOT NULL VARCHAR2(30)
 DATA_TYPE                                          VARCHAR2(106)
 DATA_TYPE_MOD                                      VARCHAR2(3)
 DATA_TYPE_OWNER                                    VARCHAR2(120)
 DATA_LENGTH                               NOT NULL NUMBER
 DATA_PRECISION                                     NUMBER
 DATA_SCALE                                         NUMBER
 NULLABLE                                           VARCHAR2(1)
 COLUMN_ID                                          NUMBER
 DEFAULT_LENGTH                                     NUMBER
 DATA_DEFAULT                                       LONG
 NUM_DISTINCT                                       NUMBER
 LOW_VALUE                                          RAW(32)
 HIGH_VALUE                                         RAW(32)
 DENSITY                                            NUMBER
 NUM_NULLS                                          NUMBER
 NUM_BUCKETS                                        NUMBER
 LAST_ANALYZED                                      DATE
 SAMPLE_SIZE                                        NUMBER
 CHARACTER_SET_NAME                                 VARCHAR2(44)
 CHAR_COL_DECL_LENGTH                               NUMBER
 GLOBAL_STATS                                       VARCHAR2(3)
 USER_STATS                                         VARCHAR2(3)
 AVG_COL_LEN                                        NUMBER
 CHAR_LENGTH                                        NUMBER
 CHAR_USED                                          VARCHAR2(1)
 V80_FMT_IMAGE                                      VARCHAR2(3)
 DATA_UPGRADED                                      VARCHAR2(3)
 HISTOGRAM                                          VARCHAR2(15)

SQL> DESC USER_TAB_COLUMNS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TABLE_NAME                                NOT NULL VARCHAR2(30)
 COLUMN_NAME                               NOT NULL VARCHAR2(30)
 DATA_TYPE                                          VARCHAR2(106)
 DATA_TYPE_MOD                                      VARCHAR2(3)
 DATA_TYPE_OWNER                                    VARCHAR2(120)
 DATA_LENGTH                               NOT NULL NUMBER
 DATA_PRECISION                                     NUMBER
 DATA_SCALE                                         NUMBER
 NULLABLE                                           VARCHAR2(1)
 COLUMN_ID                                          NUMBER
 DEFAULT_LENGTH                                     NUMBER
 DATA_DEFAULT                                       LONG
 NUM_DISTINCT                                       NUMBER
 LOW_VALUE                                          RAW(32)
 HIGH_VALUE                                         RAW(32)
 DENSITY                                            NUMBER
 NUM_NULLS                                          NUMBER
 NUM_BUCKETS                                        NUMBER
 LAST_ANALYZED                                      DATE
 SAMPLE_SIZE                                        NUMBER
 CHARACTER_SET_NAME                                 VARCHAR2(44)
```

```
 CHAR_COL_DECL_LENGTH                                    NUMBER
 GLOBAL_STATS                                            VARCHAR2(3)
 USER_STATS                                              VARCHAR2(3)
 AVG_COL_LEN                                             NUMBER
 CHAR_LENGTH                                             NUMBER
 CHAR_USED                                               VARCHAR2(1)
 V80_FMT_IMAGE                                           VARCHAR2(3)
 DATA_UPGRADED                                           VARCHAR2(3)
 HISTOGRAM                                               VARCHAR2(15)


NBR_ALL_TAB_COLUMNS
-------------------
              73142

NBR_USER_TAB_COLUMNS
--------------------
                1684
```

La structure des tables est presque la même, la seule différence c'est l'attribut `OWNER` présent dans la table `ALL_TAB_COLUMNS` en plus.

Le nombre d'instance contenue dans la table `ALL_TAB_COLUMNS` est beaucoup plus important que celui dans la table `USER_TAB_COLUMNS`

Sachant que la table `USER_TAB_COLUMNS` ne contient que les informations sur les colonnes des tables de l'utilisateur courant tandis que la table `ALL_TAB_COLUMNS` contient toutes les informations sur toutes les colonnes de tous les utilisateurs ayant un quota dans la tablespace (aux quelles l'utilisateur courant a accès).

Ainsi le contenu de la table `USER_TAB_COLUMNS` est inclus dans la table `ALL_TAB_COLUMNS`.

5 Vérification que les tables du TP1 ont été réellement créées puis affichage de toutes les informations sur ces tables.

Sachant que les tables du TP1 ont été crées par l'utilisateur `DBAINTERVENTION`, il nous suffit de récupérer les nom des tables crées par cet utilisateur.

5.1 Requête

```
SELECT TABLE_NAME
FROM ALL_TABLES
WHERE OWNER = 'DBAINTERVENTION' ;
```

5.2 Résultat

```
TABLE_NAME
------------------------------
CLIENT
EMPLOYE
MARQUE
MODELE
VEHICULE
INTERVENTIONS
INTERVENANT

7 rows selected.
```

Effectivement les tables ont toutes bien été crées lors du TP1.

5.3 Requête

```
SELECT *
FROM ALL_TABLES
WHERE OWNER = 'DBAINTERVENTION';
```

Vu que le nombre d'informations est très important nous allons afficher que 3 colonnes qui sont : le nom de la table , le nom de la tablespace dans la quelle les tables ont été crées et le nom du propriétaire de la table.

```
SELECT TABLE_NAME, TABLESPACE_NAME, OWNER
FROM ALL_TABLES
WHERE OWNER = 'DBAINTERVENTION';
```

5.4 Résultat

```
TABLE_NAME                     TABLESPACE_NAME                OWNER
------------------------------------------------------------------
CLIENT                         INTERVENTION_TBS               DBAINTERVENTION
EMPLOYE                        INTERVENTION_TBS               DBAINTERVENTION
MARQUE                         INTERVENTION_TBS               DBAINTERVENTION
MODELE                         INTERVENTION_TBS               DBAINTERVENTION
VEHICULE                       INTERVENTION_TBS               DBAINTERVENTION
INTERVENTIONS                  INTERVENTION_TBS               DBAINTERVENTION
INTERVENANT                    INTERVENTION_TBS               DBAINTERVENTION

7 rows selected.
```

6 La liste des tables de l'utilisateur SYSTEM et celles de l'utilisateur DBAINTERVENTION

6.1 Requête

```
SELECT TABLE_NAME  FROM ALL_TABLES WHERE OWNER = 'SYSTEM' ;
```

Le nombre de table étant très important (162) pour l'utilisateur SYSTEM nous allons afficher un extrait de 15 tables seulement.

6.2 Résultat

```
LOGSTDBY\$EDS_TABLES
INTERVENTIONS
INTERVENANT
SQLPLUS_PRODUCT_PROFILE
HELP
CLIENT
TABLEERREURS
EMPLOYE
MARQUE
MODELE
VEHICULE
LOGMNR_GT_TAB_INCLUDE\$
LOGMNR_GT_USER_INCLUDE\$
LOGMNR_GT_XID_INCLUDE\$
LOGMNRT_MDDL\$
...

162 rows selected.
```

puis les tables de l'utilisateur DBAINTERVENTION.

6.3 Requête

```
SELECT TABLE_NAME
FROM ALL_TABLES
WHERE OWNER = 'DBAINTERVENTION' ;
```

6.4 Résultat

```
TABLE_NAME
------------------------------
CLIENT
EMPLOYE
MARQUE
MODELE
VEHICULE
INTERVENTIONS
INTERVENANT

7 rows selected.
```

L'attribut `OWNER` dans la table `ALL_TABLES` contient le nom du créateur de la table en question.

On remarque que les tables de l'utilisateur `DBAINTERVENTION` sont incluses dans la liste des tables de l'utilisateur `SYSTEM`.

7 Affichage de la description des attributs des tables `VEHICULE` et `INTERVENTIONS`.

Nous devant d'abord prendre connaissance de la structure de la table .

7.1 Requête

```
DESC USER_TAB_COLUMNS;
```

7.2 Résultat

```
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------

 TABLE_NAME                                NOT NULL VARCHAR2(30)
 COLUMN_NAME                               NOT NULL VARCHAR2(30)
 DATA_TYPE                                          VARCHAR2(106)
 DATA_TYPE_MOD                                      VARCHAR2(3)
 DATA_TYPE_OWNER                                    VARCHAR2(120)
 DATA_LENGTH                               NOT NULL NUMBER
 DATA_PRECISION                                     NUMBER
 DATA_SCALE                                         NUMBER
 NULLABLE                                           VARCHAR2(1)
 COLUMN_ID                                          NUMBER
 DEFAULT_LENGTH                                     NUMBER
 DATA_DEFAULT                                       LONG
 NUM_DISTINCT                                       NUMBER
 LOW_VALUE                                          RAW(32)
 HIGH_VALUE                                         RAW(32)
 DENSITY                                            NUMBER
 NUM_NULLS                                          NUMBER
 NUM_BUCKETS                                        NUMBER
 LAST_ANALYZED                                      DATE
 SAMPLE_SIZE                                        NUMBER
 CHARACTER_SET_NAME                                 VARCHAR2(44)
```

```
CHAR_COL_DECL_LENGTH                                    NUMBER
GLOBAL_STATS                                            VARCHAR2(3)
USER_STATS                                              VARCHAR2(3)
AVG_COL_LEN                                             NUMBER
CHAR_LENGTH                                             NUMBER
CHAR_USED                                               VARCHAR2(1)
V80_FMT_IMAGE                                           VARCHAR2(3)
DATA_UPGRADED                                           VARCHAR2(3)
HISTOGRAM                                               VARCHAR2(15)
```

Vu le grand nombre de colonne nous nous contenterons d'afficher uniquement cinq (5) d'entre elles, qui sont : le nom de la table, le nom de la colonne, si la valeur peut être NULL le type et la taille (en nombre de caractère).

7.3 Requête

```
SELECT TABLE_NAME, COLUMN_NAME, NULLABLE, DATA_TYPE, DATA_LENGTH
FROM USER_TAB_COLUMNS
WHERE (TABLE_NAME = 'VEHICULE' OR TABLE_NAME = 'INTERVENTIONS');
```

7.4 Résultat

```
 TABLE_NAME             COLUMN_NAME                N      DATA_TYPE       DATA_LENGTH
-------------------------------------------------------------------------------------
INTERVENTIONS           NUMINTERVENTION            N      NUMBER            22
INTERVENTIONS           NUMVEHICULE                Y      NUMBER            22
INTERVENTIONS           TYPEINTERVENTION           Y      VARCHAR2          40
INTERVENTIONS           DATEDEBINTERV              Y      DATE               7
INTERVENTIONS           DATEFININTERV              Y      DATE               7
INTERVENTIONS           COUTINTERV                 Y      FLOAT             22
VEHICULE                NUMVEHICULE                N      NUMBER            22
VEHICULE                NUMCLIENT                  Y      NUMBER            22
VEHICULE                NUMMODELE                  Y      NUMBER            22
VEHICULE                NUMIMMAT                   Y      NUMBER            22
VEHICULE                ANNEE                      Y      VARCHAR2           4
11 rows selected.
```

8 Vérification de l'existance d'une référence de clé étrangère entre les tables VEHICULE et INTERVENTIONS

1. Chercher dans le dictionnaire toutes les tables qui contiennent le mot CONSTRAINTS

```
SELECT *
FROM DICT
WHERE TABLE_NAME LIKE '%CONSTRAINTS%';

TABLE_NAME                COMMENTS
------------------------------ --------------------------------------------------
DBA_CONSTRAINTS               Constraint definitions on all tables
ALL_CONSTRAINTS               Constraint definitions on accessible tables
USER_CONSTRAINTS              Constraint definitions on user s own tables
```

2. A partir de la description des tables : Extraire les noms des tables qui peuvent contenir l'information concernant la contrainte demandée, dans ce cas : USER_CONSTRAINTS puis afficher sa structure.

```
DESC USER_CONSTRAINTS;

 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 OWNER                                               VARCHAR2(120)
```

```
CONSTRAINT_NAME                                NOT NULL VARCHAR2(30)
CONSTRAINT_TYPE                                         VARCHAR2(1)
TABLE_NAME                                     NOT NULL VARCHAR2(30)
SEARCH_CONDITION                                        LONG
R_OWNER                                                 VARCHAR2(120)
R_CONSTRAINT_NAME                                       VARCHAR2(30)
DELETE_RULE                                             VARCHAR2(9)
STATUS                                                  VARCHAR2(8)
DEFERRABLE                                              VARCHAR2(14)
DEFERRED                                                VARCHAR2(9)
VALIDATED                                               VARCHAR2(13)
GENERATED                                               VARCHAR2(14)
BAD                                                     VARCHAR2(3)
RELY                                                    VARCHAR2(4)
LAST_CHANGE                                             DATE
INDEX_OWNER                                             VARCHAR2(30)
INDEX_NAME                                              VARCHAR2(30)
INVALID                                                 VARCHAR2(7)
VIEW_RELATED                                            VARCHAR2(14)
```

3. Écriture de la requête qui vérifie l'existence de la contrainte.

```
SELECT CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE CONSTRAINT_TYPE = 'R' AND TABLE_NAME = 'VEHICULE' OR TABLE_NAME = '
    ↪ INTERVENTIONS';

CONSTRAINT_NAME
------------------------------
FK_VEHICULE_CLIENT
FK_VEHICULE_MODELE
PK_INTERVENTIONS
FK_INTERVENTIONS_VEHICULE
CHK_DATEINTERV
```

Effectivement il existe une référence de clé étrangère nommé : FK_INTERVENTIONS_VEHICULE entre les tables : VEHICULE et INTERVENTIONS.

9 Affichage de toutes les contraintes créées lors du TP1 et les informations qui les caractérisent.

La structure de la table USER_CONSTRAINTS ayant été déjà étudié nous pouvant directement passé a l'écriture de la requête :

9.1 Requête

```
SELECT  *
FROM USER_CONSTRAINTS
WHERE TABLE_NAME IN ('EMPLOYE','VEHICULE','MARQUE','MODELE','INTERVENTIONS','INTERVENANT
    ↪ ','CLIENT');
```

En raison du grand volume d'information nous n'afficherons que les colonnes suivantes : CONSTRAINT_NAME, CONSTRAINT_TYPE, TABLE_NAME et STATUS.

```
SELECT  CONSTRAINT_NAME, CONSTRAINT_TYPE, TABLE_NAME, STATUS  FROM USER_CONSTRAINTS
    ↪ WHERE TABLE_NAME IN ('EMPLOYE','VEHICULE','MARQUE','MODELE','INTERVENTIONS','
    ↪ INTERVENANT','CLIENT');
```

9.2 Résultat

```
CONSTRAINT_NAME                 C TABLE_NAME                     STATUS
------------------------------ - ------------------------------ --------
CH_EMPLOYE                      C EMPLOYE                        ENABLED
FK_VEHICULE_MODELE              R VEHICULE                       ENABLED
FK_VEHICULE_CLIENT              R VEHICULE                       ENABLED
FK_MODELE_MARQUE                R MODELE                         ENABLED
FK_INTERVENTIONS_VEHICULE       R INTERVENTIONS                  ENABLED
FK_INTERVENANT_INTERVENTIONS    R INTERVENANT                    ENABLED
FK_INTERVENANT_EMPLOYE          R INTERVENANT                    ENABLED
PK_VEHICULE                     P VEHICULE                       ENABLED
PK_MODELE                       P MODELE                         ENABLED
PK_MARQUE                       P MARQUE                         ENABLED
PK_INTERVENTIONS                P INTERVENTIONS                  ENABLED
PK_INTERVENANT                  P INTERVENANT                    ENABLED
PK_EMPLOYE                      P EMPLOYE                        ENABLED
PK_CLIENT                       P CLIENT                         ENABLED
CHK_DATEINTERV                  C INTERVENTIONS                  DISABLED

15 rows selected.
```

10 Retrouver toutes les informations permettant de recréer la table INTERVENTIONS.

Pour recréer la table INTERVENTIONS nous devons avoir connaissance des informations suivantes :

1. Nom, Type, Taille des colonnes et Acceptation de valeur NULL.
2. Contraintes associés. Mais il y a possibilité que des indexes aillent été créé sur cette table donc nous aurons aussi besoin des informations suivantes :
3. Nom des indexes sur cette table.
4. Colonne Sur la quelle il existe un indexe. En plus de la possibilité que des privilèges sur cette table aillent été accordé a des utilisateur, donc :
5. Privilège et utilisateur.

10.1 Requête

```
DESC INTERVENTIONS;
```

10.2 Résultat

```
Name                                      Null?    Type
----------------------------------------- -------- ---------------

NUMINTERVENTION                           NOT NULL NUMBER(38)
NUMVEHICULE                                        NUMBER(38)
TYPEINTERVENTION                                   VARCHAR2(40)
DATEDEBINTERV                                      DATE
DATEFININTERV                                      DATE
COUTINTERV                                         FLOAT(63)
```

10.3 Requête

```
SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, TABLE_NAME, STATUS, R_OWNER ,
    ↪ R_CONSTRAINT_NAME
FROM ALL_CONSTRAINTS
WHERE TABLE_NAME = 'INTERVENTIONS';
```

10.4 Résultat

```
CONSTRAINT_NAME                C TABLE_NAME           STATUS     R_OWNER
    ↪ R_CONSTRAINT_NAME
------------------------------ - -------------------- --------- -----------------
    ↪ -----------------
FK_INTERVENTIONS_VEHICULE      R INTERVENTIONS        ENABLED    SYSTEM
    ↪ PK_VEHICULE
FK_INTERVENTIONS_VEHICULE      R INTERVENTIONS        ENABLED    DBAINTERVENTION
    ↪ PK_VEHICULE
PK_INTERVENTIONS               P INTERVENTIONS        ENABLED
PK_INTERVENTIONS               P INTERVENTIONS        ENABLED
CHK_DATEINTERV                 C INTERVENTIONS        DISABLED

7 rows selected.
```

10.5 Requête

```
SELECT  OWNER ,INDEX_NAME ,INDEX_TYPE ,TABLE_OWNER ,TABLE_TYPE
FROM ALL_INDEXES
WHERE TABLE_NAME = 'INTERVENTIONS';
```

10.6 Résultat

```
OWNER                INDEX_NAME         INDEX_TYPE      TABLE_OWNER         TABLE_TYPE
-------------------- ------------------ --------------- ------------------- -----------
SYSTEM               PK_INTERVENTIONS   NORMAL          SYSTEM              TABLE
DBAINTERVENTION      PK_INTERVENTIONS   NORMAL          DBAINTERVENTION     TABLE
```

10.7 Requête

```
SELECT *
FROM ALL_IND_COLUMNS
WHERE TABLE_NAME = 'INTERVENTIONS';
```

10.8 Résultat

```
INDEX_OWNER     INDEX_NAME         TABLE_OWNER       TABLE_NAME     COLUMN_NAME       C_P
    ↪ C_L CHAR_L DESC
--------------- ------------------ ----------------- -------------- ----------------- ---
    ↪ --- ------ ----
SYSTEM          PK_INTERVENTIONS   SYSTEM            INTERVENTIONS  NUMINTERVENTION   1
    ↪ 22  0      ASC
DBAINTERVENTION PK_INTERVENTIONS   DBAINTERVENTION   INTERVENTIONS  NUMINTERVENTION   1
    ↪ 22  0      ASC
```

10.9 Requête

```
SELECT *
FROM ALL_TAB_PRIVS
WHERE TABLE_NAME = 'INTERVENTIONS' ;
```

10.10 Résultat

```
GRANTOR GRANTEE                        TABLE_SCHEMA   TABLE_NAME      PRIVILEGE  GRA  HIE
------- ------------------------------ ------------- -------------- ---------- ---- ---
SYSTEM  GESTIONNAIRE_DES_INTERVENTIONS  SYSTEM         INTERVENTIONS  UPDATE     NO   NO
```

11 Trouver tous les privilèges accordés à Admin.

11.1 Requête

```
SELECT PRIVILEGE, TABLE_NAME
FROM ALL_TAB_PRIVS
WHERE GRANTEE = 'Admin';
```

11.2 Résultat

```
no rows selected
```

Il y a pas de privilèges pour l'utilisateur `Admin`, car nous avions retiré tous les privilèges à l'Admin lors du TP précédent.

12 Trouver les rôles donnés à l'utilisateur Admin.

Afin de prendre connaissance des colonnes de la table `DBA_ROLE_PRIVS` nous affichons d'abord sa structure.

12.1 Requête

```
DESC DBA_ROLE_PRIVS;

SELECT GRANTED_ROLE
FROM DBA_ROLE_PRIVS
WHERE GRANTEE = 'ADMIN';
```

12.2 Résultat

```
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 GRANTEE                                            VARCHAR2(30)
 GRANTED_ROLE                              NOT NULL VARCHAR2(30)
 ADMIN_OPTION                                       VARCHAR2(3)
 DEFAULT_ROLE                                       VARCHAR2(3)


GRANTED_ROLE
------------------------------
GESTIONNAIRE_DES_INTERVENTIONS
```

Nous retrouvons le rôle `GESTIONNAIRE_DES_INTERVENTIONS` crée et assigné à l'Admin lors du TP2.

13 Trouver tous les objets appartenant à Admin.

Nous nous orientons vers la table `ALL_OBJECTS` pour y récupérer les objets appartenant à Admin.

13.1 Requête

```
DESC ALL_OBJECTS;

SELECT OBJECT_NAME
FROM ALL_OBJECTS
WHERE OWNER = 'ADMIN' ;
```

13.2 Résultat

```
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 OWNER                                     NOT NULL VARCHAR2(30)
 OBJECT_NAME                               NOT NULL VARCHAR2(30)
 SUBOBJECT_NAME                                     VARCHAR2(30)
 OBJECT_ID                                 NOT NULL NUMBER
 DATA_OBJECT_ID                                     NUMBER
 OBJECT_TYPE                                        VARCHAR2(19)
 CREATED                                   NOT NULL DATE
 LAST_DDL_TIME                             NOT NULL DATE
 TIMESTAMP                                          VARCHAR2(19)
 STATUS                                             VARCHAR2(7)
 TEMPORARY                                          VARCHAR2(1)
 GENERATED                                          VARCHAR2(1)
 SECONDARY                                          VARCHAR2(1)
 NAMESPACE                                 NOT NULL NUMBER
 EDITION_NAME                                       VARCHAR2(30)


OBJECT_NAME
--------------------------------------------
NOMEMP_IX
SYS_C007170
MY_TABLE
```

Ainsi nous savons que l'Admin possède trois (3) objets dont une table : `MY_TABLE` et un index : `NOMEMP_IX`.

14 Recherche du propriétaire de la table `INTERVENTIONS` par l'administrateur.

14.1 Requête

```
SELECT OWNER
FROM ALL_TABLES
WHERE TABLE_NAME = 'INTERVENTIONS';
```

14.2 Résultat

```
OWNER
------------------------------
DBAINTERVENTION
```

15 Taille en Ko de la table `INTERVENTIONS`.

C'est à travers la table `USER_EXTENTS` que l'on pourra extraire la taille d'une table en BYTES, le résultat devra être divisé par 1024 afin de le convertir en Ko.

15.1 Requête

```
DESC USER_EXTENTS;

SELECT BYTES/1024 AS Taille_Ko
FROM USER_EXTENTS
WHERE SEGMENT_NAME = 'INTERVENTIONS';
```

15.2 Résultat

```
 Name                                      Null?    Type
 ----------------------------------------- -------- --------------
 SEGMENT_NAME                                       VARCHAR2(81)
 PARTITION_NAME                                     VARCHAR2(30)
 SEGMENT_TYPE                                       VARCHAR2(18)
 TABLESPACE_NAME                                    VARCHAR2(30)
 EXTENT_ID                                          NUMBER
 BYTES                                              NUMBER
 BLOCKS                                             NUMBER

 TAILLE_KO
----------
        64
```

16 Vérification de l'effet produit par chacune des commandes de définition de données du TP1 sur le dictionnaire.

Les commandes de définition de données du TP1 se résument en :

1. Création d'un utilisateur
2. Création d'une table
3. Ajout d'une colonne à une table
4. Ajout d'un contrainte à une table
5. Accorder des privilèges à un utilisateur
6. Retirer des privilèges à un utilisateur

Création d'un utilisateur :

16.1 Requête/Résultat avant

```
SQL> SELECT USERNAME FROM ALL_USERS ;

USERNAME
------------------------------
XS\$NULL
ADMIN
USER0
USER1
APEX_040000
APEX_PUBLIC_USER
FLOWS_FILES
HR
MDSYS
ANONYMOUS
XDB
CTXSYS
```

```
OUTLN
SYSTEM
SYS
ME
DBAINTERVENTION

17 rows selected.
```

16.2 Requête/Résulta commande de définition de donnée

```
SQL> CREATE USER TESTEUR IDENTIFIED BY TESTEUR;

User created.
```

16.3 Requête/Résultat après

```
SQL> SELECT USERNAME FROM ALL_USERS ;

USERNAME
------------------------------
XS\$NULL
TESTEUR
ADMIN
USER0
USER1
APEX_040000
APEX_PUBLIC_USER
FLOWS_FILES
HR
MDSYS
ANONYMOUS
XDB
CTXSYS
OUTLN
SYSTEM
SYS
ME
DBAINTERVENTION

18 rows selected.
```

Création d'une table :

16.4 Requête/Résultat avant

```
 SQL> SELECT * FROM USER_TABLES WHERE TABLE_NAME = 'TABLE_TESTEUR' ;

no rows selected
```

16.5 Requête/Résulta commande de définition de donnée

```
SQL> CREATE TABLE TABLE_TESTEUR (ID_TESTEUR INTEGER PRIMARY KEY, NOM_TESTEUR VAR
CHAR2(30), NBR INTEGER DEFAULT(0));

Table created.
```

16.6 Requête/Résultat après

```
SQL> SELECT TABLE_NAME,TABLESPACE_NAME, STATUS FROM USER_TABLES WHERE TABLE_NAME
= 'TABLE_TESTEUR' ;

TABLE_NAME                     TABLESPACE_NAME                STATUS
------------------------------ ------------------------------ --------
TABLE_TESTEUR                  SYSTEM                         VALID
```

Ajout d'une colonne à une table :

16.7 Requête/Résultat avant

```
SQL> SELECT TABLE_NAME ,COLUMN_NAME , DATA_LENGTH FROM USER_TAB_COLUMNS WHERE
    ↪ TABLE_NAME = 'TABLE_TESTEUR';

 TABLE_NAME                     COLUMN_NAME                    DATA_LENGTH
------------------------------ ------------------------------ -----------
TABLE_TESTEUR                  ID_TESTEUR                              22
TABLE_TESTEUR                  NOM_TESTEUR                             30
TABLE_TESTEUR                  NBR                                     22
```

16.8 Requête/Résulta commande de définition de donnée

```
SQL>  ALTER TABLE TABLE_TESTEUR ADD PRENOM_TESTEUR VARCHAR2(30) ;

Table altered.
```

16.9 Requête/Résultat après

```
SQL>  SELECT TABLE_NAME ,COLUMN_NAME , DATA_LENGTH FROM USER_TAB_COLUMNS WHERE T
ABLE_NAME = 'TABLE_TESTEUR';

TABLE_NAME                     COLUMN_NAME                    DATA_LENGTH
------------------------------ ------------------------------ -----------
TABLE_TESTEUR                  ID_TESTEUR                              22
TABLE_TESTEUR                  NOM_TESTEUR                             30
TABLE_TESTEUR                  NBR                                     22
TABLE_TESTEUR                  PRENOM_TESTEUR                          30
```

Ajout d'un contrainte à une table :

16.10 Requête/Résultat avant

```
SELECT CONSTRAINT_NAME FROM ALL_CONSTRAINTS WHERE TABLE_NAME = 'TABLE_TESTEUR';


CONSTRAINT_NAME
------------------------------
SYS_C007211
```

16.11 Requête/Résulta commande de définition de donnée

```
SQL>  ALTER TABLE TABLE_TESTEUR ADD CONSTRAINT CH_NBR8POSITIF CHECK ( NBR > 0);

Table altered.
```

16.12 Requête/Résultat après

```
SQL>  SELECT CONSTRAINT_NAME FROM ALL_CONSTRAINTS WHERE TABLE_NAME = 'TABLE_TES
EUR';

CONSTRAINT_NAME
------------------------------
SYS_C007211
CH_NBR8POSITIF
```

Accorder des privilèges à un utilisateur :

16.13 Requête/Résultat avant

```
 SELECT * FROM ALL_TAB_PRIVS WHERE TABLE_NAME = 'TABLE_TESTEUR';

 no rows selected
```

16.14 Requête/Résulta commande de définition de donnée

```
SQL> GRANT SELECT ON TABLE_TESTEUR TO TESTEUR;

Grant succeeded.
```

16.15 Requête/Résultat après

```
SQL>   SELECT * FROM ALL_TAB_PRIVS WHERE TABLE_NAME = 'TABLE_TESTEUR';

GRANTOR      GRANTEE   TABLE_SCHEMA     TABLE_NAME      PRIVILEGE     GRA HIE
------------ --------- ---------------- --------------- ------------- --- ---
SYSTEM       TESTEUR   SYSTEM           TABLE_TESTEUR   SELECT        NO  NO
```

Retirer des privilèges à un utilisateur :

16.16 Requête/Résultat avant

```
SQL>   SELECT * FROM ALL_TAB_PRIVS WHERE TABLE_NAME = 'TABLE_TESTEUR';

GRANTOR      GRANTEE   TABLE_SCHEMA     TABLE_NAME      PRIVILEGE     GRA HIE
------------ --------- ---------------- --------------- ------------- --- ---
SYSTEM       TESTEUR   SYSTEM           TABLE_TESTEUR   SELECT        NO  NO
```

16.17 Requête/Résulta commande de définition de donnée

```
SQL> REVOKE SELECT ON TABLE\_TESTEUR FROM TESTEUR;

Revoke succeeded.
```

16.18 Requête/Résultat après

```
SQL>   SELECT * FROM ALL_TAB_PRIVS WHERE TABLE_NAME = 'TABLE_TESTEUR';

no rows selected
```

Toutes les commandes de définition de données présentées dans cette question reprennent celles du TP1 avec d'autres noms uniquement pour limiter la taille des affichages.

Ainsi nous constatant que les commandes de définition de données ont un impacte sur le dictionnaire, car ce dernier stock sous forme de table toutes les informations relatives au table/view/index/user de la base de donnée.