

*RAPPORT*

*TP3*

*DECTIONNNAIRE*

*ORACLE :*

YOUSFI

ZAKARIA

171732026950

GRP TP : 3

## 1) Connecter en tant que « System » :

Pour lister le catalogue Dict, On utilise une simple requête d'interrogation select:

```
Select * from dict;
```

Elle nous donne un résultat de 1821 lignes, ou bien on peut utiliser la fonction count pour calculer le nombre d'instances :

```
Select count(*) from dict;
```

```
sqlplus.exe
V$SERV_MOD_ACT_STATS
Synonym for V_$SERV_MOD_ACT_STATS

GV$SERV_MOD_ACT_STATS
Synonym for GV_$SERV_MOD_ACT_STATS

V$SERVICE_STATS
TABLE_NAME
-----
COMMENTS
-----
Synonym for V_$SERVICE_STATS

GV$SERVICE_STATS
Synonym for GV_$SERVICE_STATS

V$SYS_TIME_MODEL
Synonym for V_$SYS_TIME_MODEL

GV$SYS_TIME_MODEL
TABLE_NAME
-----
COMMENTS
-----
Synonym for GV_$SYS_TIME_MODEL

V$SESS_TIME_MODEL
Synonym for V_$SESS_TIME_MODEL

GV$SESS_TIME_MODEL
Synonym for GV_$SESS_TIME_MODEL

DBA_SQLSET_DEFINITIONS
TABLE_NAME
-----
COMMENTS
-----
Synonym for DBA_SQLSET

USER_SQLSET_DEFINITIONS
Synonym for USER_SQLSET

1821 ligne(s) sélectionné(s).
SQL> select count(*) from dict;
  COUNT(*)
  -----
       1821
SQL>
```

On peut utiliser la requête desc pour afficher la structure du dict:

**Desc Dict;**

```
SQL> desc dict;
Nom                                NULL ?    Type
-----
TABLE_NAME                        VARCHAR2(30)
COMMENTS                          VARCHAR2(4000)
SQL>
```

Donc le catalogue DICT contient 2 colonnes qui représentant le nom des tables et des commentaires relatives à ses tables.

## 2) **Le rôle et la Structure :**

Pour le rôle on peut utiliser une requête pour extraire le commentaire de chaque table :

**select comments from dict where table\_name=Nom\_table';**

Et pour la structure il est évident que on peut utiliser la requête desc Nom\_table

### **ALL TAB COLUMNS:**

Rôle : c'est une table qui regroupe pour l'utilisateur courant les colonnes de leurs tables/views/clusters.

```
SQL> select comments from dict where table_name='ALL_TAB_COLUMNS';
COMMENTS
-----
Columns of user's tables, views and clusters
SQL>
```

Structure :

```
SQL> desc ALL_TAB_COLUMNS
```

Nom	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(30)
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)

## USER\_USERS:

Rôle: elle contient des informations sur l'utilisateur courant

```
SQL> select comments from dict where table_name='USER_USERS';
```

COMMENTS
-----
Information about the current user

```
SQL>
```

Structure :

```
SQL> desc USER_USERS;
```

Nom	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>
```

## ALL CONSTRAINTS:

Rôle : Contient les définitions des constraints sur la table accessible à l'utilisateur courant.

```
SQL> select comments from dict where table_name='ALL_CONSTRAINTS';
```

COMMENTS

-----  
Constraint definitions on accessible tables

Structure :

```
SQL> desc all_constraints;
```

Nom	NULL ?	Type
OWNER	NOT NULL	VARCHAR2(30)
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)
CONSTRAINT_TYPE		VARCHAR2(1)
TABLE_NAME	NOT NULL	VARCHAR2(30)
SEARCH_CONDITION		LONG
R_OWNER		VARCHAR2(30)
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)

## USER\_TAB PRIVS:

Rôle : elle contient les privilèges concernant des objets ou l'utilisateur courant est le propriétaire/a donné ou bien été donné le privilège.

```
SQL> select comments from dict where table_name='USER_TAB_PRIVS';
```

COMMENTS

-----  
Grants on objects for which the user is the owner, grantor or grantee

Structure :

```
SQL> desc user_tab_privs;
```

Nom	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) **Nom d'utilisateur connecté :**

On utilise USER\_USERS car elle contient l'information d'utilisateur courant, avec la requête :

Select username from USER\_USERS;

```
SQL> select username from USER_USERS;
```

```
USERNAME
```

```
-----
```

```
SYSTEM
```

```
SQL>
```

### 4) **Comparison entre all\_tab\_columns and user\_tab\_columns:**

La différence est que all\_tab\_columns a un attribut en plus qui est le 'owner' c.-à-d. le propriétaire. Donc all\_tab\_columns contient des informations sur toutes les tables de la BD pour tous les utilisateurs. Un simple count retourne plus de 45000 résultats mais user\_tab\_columns est propre à l'utilisateur courant. Un simple count en utilisant différents utilisateurs nous donne des résultats différents. Dans la figure suivante on remarque que quelque soit l'utilisateur courant le compte reste le même pour all\_tab\_columns mais change pour user\_tab\_columns.



```

SQL> connect dbaintervention/user1337;
Connecté.
SQL> select distinct owner from user_tab_columns;
select distinct owner from user_tab_columns
*
ERREUR Ó la ligne 1 :
ORA-00904: "OWNER" : identificateur non valide

SQL> select count(*) from user_tab_columns;

COUNT(*)
-----
41

SQL> select count(*) from all_tab_columns;

COUNT(*)
-----
45629

SQL> connect system/oracle123456
Connecté.
SQL> select count(*) from user_tab_columns;

COUNT(*)
-----
1404

SQL> select count(*) from all_tab_columns;

COUNT(*)
-----
45629

SQL>

```

```
SQL> desc all_tab_columns;
```

```

Nom
-----
OWNER
TABLE_NAME
COLUMN_NAME
DATA_TYPE
DATA_TYPE_MOD
DATA_TYPE_OWNER
DATA_LENGTH
DATA_PRECISION
DATA_SCALE
NULLABLE
COLUMN_ID
DEFAULT_LENGTH
DATA_DEFAULT
NUM_DISTINCT
LOW_VALUE
HIGH_VALUE
DENSITY
NUM_NULLS
NUM_BUCKETS
LAST_ANALYZED
SAMPLE_SIZE
CHARACTER_SET_NAME
CHAR_COL_DECL_LENGTH
GLOBAL_STATS
USER_STATS
AVG_COL_LEN
CHAR_LENGTH
CHAR_USED
V80_FMT_IMAGE
DATA_UPGRADED
HISTOGRAM

```

```
SQL> desc user_tab_columns;
```

```

Nom
-----
TABLE_NAME
COLUMN_NAME
DATA_TYPE
DATA_TYPE_MOD
DATA_TYPE_OWNER
DATA_LENGTH
DATA_PRECISION
DATA_SCALE
NULLABLE
COLUMN_ID
DEFAULT_LENGTH
DATA_DEFAULT
NUM_DISTINCT
LOW_VALUE
HIGH_VALUE
DENSITY
NUM_NULLS
NUM_BUCKETS
LAST_ANALYZED
SAMPLE_SIZE
CHARACTER_SET_NAME
CHAR_COL_DECL_LENGTH
GLOBAL_STATS
USER_STATS
AVG_COL_LEN
CHAR_LENGTH
CHAR_USED
V80_FMT_IMAGE
DATA_UPGRADED
HISTOGRAM

```

```
VARCHAKZ (15)
```

### 5) Vérification d'existence des tables du tp1:

On peut directement avec l'utilisateur system en utilisant la table all\_tables avec la requête :

```
select table_name
from all_tables
where owner = upper('dbaintervention');
```

```
SQL> select table_name from all_tables where owner = upper('dbaintervention');
TABLE_NAME
-----
EMPLOYE
MARQUE
MODELE
CLIENT
VEHICULE
INTERVENTIONS
INTERVENANTS
TABLEERREURS
8 ligne(s) sélectionné(s).
```

ou on peut se connecte avec l'utilisateur dbaintervention d'abord puis on exécute la requête :

```
Select table_name form user_tables;
```

```
SQL> connect dbaintervention/user1337
Connecté.
SQL> select table_name from user_tables;
TABLE_NAME
-----
EMPLOYE
MARQUE
MODELE
CLIENT
VEHICULE
INTERVENTIONS
INTERVENANTS
TABLEERREURS
8 ligne(s) sélectionné(s).
```

Les tables sont bien créées.



Pour les informations sur ces tables on peut utiliser la requête :

**Select \* from all\_tables where owner = 'DBAINTERVENTION';**

```

DURATION      SKIP_COR MON CLUSTER_OWNER      DEPENDEN COMPRESS
-----
DRO
---
NO
OWNER          TABLE_NAME
-----
TABLESPACE_NAME CLUSTER_NAME
-----
IOT_NAME        STATUS      PCT_FREE    PCT_USED    INI_TRANS
-----
MAX_TRANS      INITIAL_EXTENT NEXT_EXTENT MIN_EXTENTS MAX_EXTENTS PCT_INCREASE
-----
FREELISTS      FREELIST_GROUPS LOG B      NUM_ROWS      BLOCKS EMPTY_BLOCKS AVG_SPACE
-----
CHAIN_CNT      AVG_ROW_LEN  AVG_SPACE_FREELIST_BLOCKS NUM_FREELIST_BLOCKS
-----
DEGREE
-----
INSTANCES
-----
SAMPLE_SIZE    LAST_ANAL PAR IOT_TYPE      T S NES BUFFER_    ROW_MOVE    GLO USE
-----
DURATION      SKIP_COR MON CLUSTER_OWNER      DEPENDEN COMPRESS
-----
DRO
---
8 ligne(s) sUlectionnnUe(s).

```

#### 6) lister les tables des utilisateurs system et dbaintervention :

On peut utiliser une seule requête pour les deux :

**Select owner,table\_name from all\_tables where owner='SYSTEM' or  
owner='DBAINTERVENTION' order by owner;**

Ou bien chaque utilisateur une requête :

**Select table\_name  
from all\_tables  
where owner = 'SYSTEM';**

ou comme on est connecté avec system:

**Select table\_name from user\_tables;**

Et

Select table\_name from all\_tables where owner='DBAINTERVENTION';

```
SQL> select owner,table_name from all_tables where owner='SYSTEM' or owner='DBAINTERVENTION' order by owner;
```

OWNER	TABLE_NAME
DBAINTERVENTION	EMPLOYE
DBAINTERVENTION	MARQUE
DBAINTERVENTION	MODELE
DBAINTERVENTION	TABLEERREURS
DBAINTERVENTION	VEHICULE
DBAINTERVENTION	INTERVENTIONS
DBAINTERVENTION	INTERVENANTS
DBAINTERVENTION	CLIENT
SYSTEM	MVIEWS_ADV_WORKLOAD
SYSTEM	MVIEWS_ADV_BASSETABLE
SYSTEM	MVIEWS_ADV_SQLDEPEND

  

OWNER	TABLE_NAME
SYSTEM	MVIEWS_ADV_PRETTY
SYSTEM	MVIEWS_ADV_TEMP
SYSTEM	MVIEWS_ADV_FILTER
SYSTEM	MVIEWS_ADV_LOG
SYSTEM	MVIEWS_ADV_FILTERINSTANCE
SYSTEM	MVIEWS_ADV_LEVEL
SYSTEM	MVIEWS_ADV_ROLLUP
SYSTEM	MVIEWS_ADV_AJG
SYSTEM	MVIEWS_ADV_FJG
SYSTEM	MVIEWS_ADV_GC
SYSTEM	MVIEWS_ADV_CLIQUE

  

OWNER	TABLE_NAME
SYSTEM	MVIEWS_ADV_FILTER

## 7) Description des attributs des tables VEHICULE et INTERVENTIONS :

Une simple requête pour chaque table :

Select column\_name,data\_type,data\_length from user\_tab\_columns  
where table\_name='VEHICULE';

```

SQL> Select column_name,data_type,data_length from user_tab_columns where table_name='VEHICULE';
aucune ligne sélectionnée

SQL> connect dbaintervention/user1337
ConnectU.
SQL> Select column_name,data_type,data_length from user_tab_columns where table_name='VEHICULE';

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
NUMVEHICULE
NUMBER          22

NUMCLIENT
NUMBER          22

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
NUMMODELE
NUMBER          22

NUMIMMAT
VARCHAR2

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
              20

ANNEE
VARCHAR2          4

```

Select column\_name,data\_type,data\_length from user\_tab\_columns  
where table\_name='INTERVENTIONS';

```

SQL> Select column_name,data_type,data_length from user_tab_columns where table_name='INTERVENTIONS';

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
NUMINTERVENTION
NUMBER          22

NUMVEHICULE
NUMBER          22

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
TYPEINTERVENTION
VARCHAR2          50

DATEDEBINTERV
DATE

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
              7

DATEFININTERV
DATE              7

COUTINTERV

COLUMN_NAME
-----
DATA_TYPE
-----
DATA_LENGTH
-----
NUMBER          22

```

**8) vérification qu'il y a une référence de clé étrangère entre les tables VEHICULE et INTERVENTIONS :**

Il faut vérifier les contraintes de la table intervention, on connecte avec dbaintervention et on exécute la requête :

```
select constraint_name, constraint_type  
from user_constraints  
where table_name='INTERVENTIONS';
```

```
SQL> connect dbaintervention/user1337  
ConnectU.  
SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE  
2 TABLE_NAME='INTERVENTIONS';
```

CONSTRAINT_NAME	C
-----	-
CK_DATEINTER	C
PK_INTERVENTION	P
FK_VEHICULE	R

On Remarque la dernière contrainte qui est la contrainte clé étrangère cherché.

**9) Les contraintes du TP1 et de leurs caractéristiques :**

On utilise la requête :

```
Select * from user_constraints ;
```

```

VIEW_RELATED
-----
NOT DEFERRABLE IMMEDIATE VALIDATED          USER NAME          29/10/22
OWNER                                CONSTRAINT_NAME          C
-----
TABLE_NAME
-----
SEARCH_CONDITION
-----
R_OWNER                                R_CONSTRAINT_NAME          DELETE_RU STATUS
-----
DEFERRABLE      DEFERRED  VALIDATED      GENERATED      BAD RELY LAST_CHA
-----
INDEX_OWNER                                INDEX_NAME          INVALID
-----
VIEW_RELATED
-----
DBAINTERVENTION          PK_INTERVENANTS
OWNER                                CONSTRAINT_NAME          C
-----
TABLE_NAME
-----
SEARCH_CONDITION
-----
R_OWNER                                R_CONSTRAINT_NAME          DELETE_RU STATUS
-----
DEFERRABLE      DEFERRED  VALIDATED      GENERATED      BAD RELY LAST_CHA
-----
INDEX_OWNER                                INDEX_NAME          INVALID
-----
VIEW_RELATED
-----
OWNER                                CONSTRAINT_NAME          C
-----
TABLE_NAME
-----
SEARCH_CONDITION
-----
R_OWNER                                R_CONSTRAINT_NAME          DELETE_RU STATUS
-----
DEFERRABLE      DEFERRED  VALIDATED      GENERATED      BAD RELY LAST_CHA
-----
INDEX_OWNER                                INDEX_NAME          INVALID
-----
VIEW_RELATED
-----

18 ligne(s) s lectionn e(s).

```

## 10) toutes les informations permettant de recrer la table INTERVENTIONS :

Une table est caract ris e par ses attribues et les constraints d'int grit  donc il faut chercher ses infos

Pour les attribues on peut exploiter user\_tab\_columns donc la requ te :

```

select column_name, data_type, data_length from user_tab_columns
where table_name='INTERVENTIONS';

```

```
SQL> select column_name, data_type, data_length from user_tab_columns
2 where table_name='INTERVENTIONS';
```

```
COLUMN_NAME
```

```
DATA_TYPE
```

```
DATA_LENGTH
```

```
NUMINTERVENTION
NUMBER
```

```
22
```

```
NUMVEHICULE
NUMBER
```

```
22
```

```
COLUMN_NAME
```

```
DATA_TYPE
```

```
DATA_LENGTH
```

```
TYPEINTERVENTION
VARCHAR2
```

```
50
```

```
DATEDEBINTERV
DATE
```

```
COLUMN_NAME
```

```
DATA_TYPE
```

```
DATA_LENGTH
```

```
7
```

```
DATEFININTERV
```

Et les constraints avec la requête :

```
Select constraint_name,r_constraint_name,constraint_type from
user_constraints where table_name='INTERVENTIONS';
```

```
SQL> Select constraint_name,r_constraint_name,constraint_type from user_constraints where table_name='INTERVENTIONS';
```

CONSTRAINT_NAME	R_CONSTRAINT_NAME	C
CK_DATEINTER		C
PK_INTERVENTION		P
FK_VEHICULE	PK_VEHICULE	R

11) tous les privilèges accordés à Admin :



Il faut utiliser user\_tab\_privs avec grantee=admin pour avoir les privilèges accordés à admin, la requête :

```
select privilege, table_name from user_tab_privs where  
grantee=upper('admin');
```

```
SQL> select privilege, table_name from user_tab_privs where grantee=upper('admin');
```

PRIVILEGE	TABLE_NAME
INDEX	EMPLOYE
UPDATE	EMPLOYE
SELECT	EMPLOYE
SELECT	INTERVENANTS

## 12) les rôles donnés à l'utilisateur Admin :

Soit on connecte avec l'utilisateur system et on exécute la requête :

```
select * from dba_role_privs where GRANTEE =upper('admin');
```

```
SQL> connect system/oracle123456  
Connecté.  
SQL>  
SQL> select * from dba_role_privs where GRANTEE =upper('admin');
```

GRANTEE	GRANTED_ROLE	ADM	DEF
ADMIN	GESTIONNAIRE_DES_INTERVENTIONS	NO	YES

Ou bien connecter comme admin et exécuter la requête :

```
select * from user_role_privs ;
```

```
SQL> grant create session to admin;  
Autorisation de privilèges (GRANT) acceptée.  
SQL> connect admin/admin1337  
Connecté.  
SQL> select * from user_role_privs;
```

USERNAME	GRANTED_ROLE	ADM	DEF	OS_
ADMIN	GESTIONNAIRE_DES_INTERVENTIONS	NO	YES	NO

## 13) tous les objets appartenant à Admin :

Du l'utilisateur system avec la requête :

```
select * from all_objects where owner=upper('admin');
```

```
SQL> connect system/oracle123456
ConnectU.
SQL> select * from all_objects where owner=upper('admin');
```

OWNER	OBJECT_NAME	SUBOBJECT_NAME	OBJECT_ID	DATA_OBJECT_ID	OBJECT_TYPE	CREATED	LAST_DDL	TIMESTAMP	STATUS	T	G	S
ADMIN	TEST		13668	13668	TABLE	30/10/22	30/10/22	2022-10-30:09:50:11	VALID	N	N	N
ADMIN	VIEW1		13669		VIEW	30/10/22	30/10/22	2022-10-30:09:54:01	VALID	N	N	N
ADMIN	NOMEMP_IX		13670	13670	INDEX	30/10/22	30/10/22	2022-10-30:11:33:42	VALID	N	N	N

Ou bien on connecte avec admin et on utilise la requête :

```
Select * from user_objects ;
```

```
SQL> connect admin/admin1337
ConnectU.
SQL> select * from user_objects;
```

OBJECT_NAME	SUBOBJECT_NAME	OBJECT_ID	DATA_OBJECT_ID	OBJECT_TYPE	CREATED	LAST_DDL	TIMESTAMP	STATUS	T	G	S
TEST		13668	13668	TABLE	30/10/22	30/10/22	2022-10-30:09:50:11	VALID	N	N	N
VIEW1		13669		VIEW	30/10/22	30/10/22	2022-10-30:09:54:01	VALID	N	N	N
NOMEMP_IX		13670	13670	INDEX	30/10/22	30/10/22	2022-10-30:11:33:42	VALID	N	N	N

14) **chercher le propriétaire de la table INTERVENTIONS :**

Une simple requête qui utilise la table all\_tables

```
select owner from all_tables where table_name='INTERVENTIONS';
```

```
SQL> select owner from all_tables where table_name='INTERVENTIONS';
OWNER
-----
DBAINTERVENTION
```

### 15) **La taille en Ko de la table INTERVENTIONS :**

On connecte avec system pour utiliser la vue dba\_segments, et comme la taille sera en octets alors on divise par 1024 pour avoir la taille en ko. La requête :

```
select bytes/1024 as size_ from dba_segments where
segment_name='INTERVENTIONS';
```

```
SQL> select bytes/1024 as size_ from dba_segments where segment_name='INTERVENTIONS';
SIZE_
-----
64
```

### 16) **Vérification de l'effet:**

Pour la création d'une table :

Avant :

```
SQL> connect dbaintervention/user1337;
Connecté.
SQL> select table_name from user_tables;

TABLE_NAME
-----
EMPLOYE
MARQUE
MODELE
CLIENT
VEHICULE
INTERVENTIONS
INTERVENANTS
TABLEERREURS

8 ligne(s) sélectionné(s).
```

## Après :

```
SQL> create table dummytable(dummyattribute varchar2(10) not null, dummyattribute2 number(5) not null);
Table créée.
SQL> select table_name from user_tables;
TABLE_NAME
-----
EMPLOYE
MARQUE
MODELE
CLIENT
VEHICULE
INTERVENTIONS
INTERVENANTS
TABLEERREURS
DUMMYTABLE
9 ligne(s) sélectionnée(s).
```

Pour les contraintes :

## Avant :

```
SQL> select table_name,constraint_name from user_constraints where table_name='DUMMYTABLE';
TABLE_NAME      CONSTRAINT_NAME
-----
DUMMYTABLE      SYS_C004058
DUMMYTABLE      SYS_C004059
```

## Après :

```
SQL> alter table dummytable add constraint pk_xd primary key(dummyattribute);
Table modifiée.
SQL> select table_name,constraint_name from User_constraints where table_name='DUMMYTABLE';
TABLE_NAME      CONSTRAINT_NAME
-----
DUMMYTABLE      SYS_C004058
DUMMYTABLE      SYS_C004059
DUMMYTABLE      PK_XD
```

Suppression d'une colonne :

## Avant et Après:

```
SQL> select column_name from user_tab_columns where table_name='DUMMYTABLE';
COLUMN_NAME
-----
DUMMYATTRIBUTE
DUMMYATTRIBUTE2
SQL> alter table dummytable drop column dummyattribute2;
Table modifiée.
SQL> select column_name from user_tab_columns where table_name='DUMMYTABLE';
COLUMN_NAME
-----
DUMMYATTRIBUTE
```

**THE END.**