Aharouite Ahmed

BDA / TP01 Oracle

Table des matières

1. Introduction

Ce TP sert de révision sur le langage SQL et PL/SQL.

2. Manipulation de la base de données

2.1. SQL

(3) Vérifier le contenu des tables avec la commande suivante :

```
SELECT * FROM Continent;
```

	NOM	SUPERFICIE
1	Asie	44579000
2	Afrique	30065000
3	Amérique	42189120
4	Antarctique	13209000
5	Europe	9938000
6	Océanie	7687000

```
SELECT * FROM Frontiere;
```

	NOMP	NOMF	LONGUEUR
1	Chine	Afghanistan	76
2	Chine	Bhoutan	470
639	Monaco	France	5
640	Vatican	Italie	3

```
SELECT * FROM Montagne;
```

	NM	NOM	ALTITUDE	CHAINE
1	1	Kilimandjaro - Kibo	5892	Vallée du grand rift
2	2	Mont Kenya	5199	Vallée du grand rift
234	234	Pics Doma	3568	Monts Bismarck
235	235	Mont Priora	3557	Monts Bismarck

```
SELECT * FROM Fleuve;
```

	NF	NOM	LONGUEUR	EMBOUCHURE
1	1	La Medjerda	460	mer Méditerranée
2	2	Le Bandama	1050	lagune Lahou
3	3	La Betsiboka	525	baie de Bombetoka
4	4	Le Chelif	733	mer Méditerranée
229	229	le Waitaki	110	océan Pacifique
230	230	la Whanganui	290	mer de Tasman

```
SELECT * FROM Traverser;
```

	NF	NOMP
1	1	Tunisie
2	2	Côte d'Ivoire
374	229	Nouvelle-Zélande
375	230	Nouvelle-Zélande

(4) A partir du dictionnaire de données Oracle, affichez les meta-données suivantes :

- à partir de USER_OBJECTS les tables (OBJECT_NAME) et leur date de création (CREATED)

```
SELECT object_name, created FROM User_Objects;
```

	OBJECT_NAME	CREATED
1	UK_PAYS_ISO3	18/09/20
2	UK_PAYS_ISO2	18/09/20
3	TRAVERSER	18/09/20
16	FRONTIERE	18/09/20
17	FLEUVE	18/09/20
18	CONTINENT	18/09/20

- à partir de USER_TAB_COLUMNS les attributs (COLUMN_NAME, DATA_TYPE, DATA_LENGTH, DATA_PRECISION) ordonnés par table (TABLE_NAME).

```
SELECT column_name, data_type, data_length, data_precision FROM User_Tab_Columns;
```

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	DATA_PRECISION
1	NOM	VARCHAR2	50	(null)
2	SUPERFICIE	NUMBER	22	(null)
23	NF	NUMBER	22	(null)
24	NOMP	VARCHAR2	50	(null)

- à partir de USER_CONSTRAINTS, les contraintes d'intégrités (TABLE_NAME, CONSTRAINT_NAME, CONSTRAINT_TYPE, SEARCH_CONDITION) ordonnées par table (TABLE_NAME), puis par type de contrainte.

```
SELECT table_name, constraint_name, constraint_type, search_condition FROM User_Constraints;
```

	TABLE_NAME	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	CONTINENT	CK_CONTINENT_SUPERFICIE	C	superficie > 0
2	PAYS	CK_PAYS_SUPERFICIE	C	superficie > 0
21	FLEUVE	PK_FLEUVE	P	(null)
22	TRAVERSER	PK_TRAVERSER	P	(null)

2.2. PL/SQL

(1) Créez une procédure stockée en PL/SQL conforme aux spécifications suivantes : (voir sujet TP)

```
CREATE OR REPLACE PROCEDURE Ajout_Pays
(vnom Pays.nom%TYPE, vcap Pays.capitale%TYPE, vsup Pays.superficie%TYPE,
vpop Pays.population%TYPE, viso3 Pays.iso3%TYPE, viso2 Pays.iso2%TYPE, vnomc Pays.nomc%TYPE)
IS

erreur_ck EXCEPTION;
PRAGMA EXCEPTION_INIT (erreur_ck, -02290);
erreur_fk EXCEPTION;
PRAGMA EXCEPTION_INIT (erreur_fk, -02291);
erreur_uk EXCEPTION;
PRAGMA EXCEPTION_INIT (erreur_uk, -00001);

BEGIN
INSERT INTO Pays VALUES(vnom, vcap, vsup, vpop, viso3, viso2, vnomc);
DBMS_OUTPUT.PUT_LINE('Pays enregistré');
COMMIT;
```

```
EXCEPTION
WHEN erreur_fk THEN
    IF (SQLERRM LIKE '%FK_PAYS_CONTINENT%') THEN
        DBMS_OUTPUT.PUT_LINE('Le continent est inconnu');
    END IF;
WHEN erreur_ck THEN
    IF (SQLERRM LIKE '%CK_PAYS_SUPERFICIE%') THEN
        DBMS_OUTPUT.PUT_LINE('La superficie est invalide');
    ELSIF (SQLERRM LIKE '%CK_PAYS_POPULATION%') THEN
        DBMS_OUTPUT.PUT_LINE('La superficie est invalide');
    END IF;
WHEN erreur_uk THEN
    IF (SQLERRM LIKE '%UK_PAYS_ISO3%') THEN
        DBMS_OUTPUT.PUT_LINE('Le code iso3 existe déjà');
    ELSIF (SQLERRM LIKE '%UK_PAYS_ISO2%') THEN
        DBMS_OUTPUT.PUT_LINE('Le code iso2 existe déjà');
    ELSIF (SQLERRM LIKE '%PK_PAYS%') THEN
        DBMS_OUTPUT.PUT_LINE('Le pays existe déjà ');
    END IF;

END;
/
```

(2) Testez votre procédure en ajoutant le pays suivant.

```
EXECUTE Ajout_Pays ('IUT de Blagnac', 'Dept Info', NULL, NULL, 'IUT', 'UT', 'Europe');
```

```
Pays enregistré
```

```
SELECT * FROM Pays WHERE nom = 'IUT de Blagnac';
```

	NOM	CAPITALE	SUPERFICIE	POPULATION	ISO3	ISO2	NOMC
1	IUT de Blagnac	Dept Info	(null)	(null)	IUT	UT	Europe

(3) Testez votre procédure avec tous les cas d'erreur.

➔ Pays existant

```
EXECUTE Ajout_Pays ('IUT de Blagnac', 'Dept Info', NULL, NULL, 'IUT', 'UT', 'Europe');
```

```
Le pays existe déjà


Procédure PL/SQL terminée.
```

➔ Code ISO3 existant

```
EXECUTE Ajout_Pays ('IUT de Blagnac2', 'Dept Info2', NULL, NULL, 'IUT', 'U2', 'Europe2');
```

```
Le code iso3 existe déjà


Procédure PL/SQL terminée.
```

➔ Code ISO2 existant

```
EXECUTE Ajout_Pays ('IUT de Blagnac2', 'Dept Info2', NULL, NULL, 'IU2', 'UT', 'Europe2');
```

```
Le code iso2 existe déjà


Procédure PL/SQL terminée.
```

➔ Continent inconnu

```
EXECUTE  Ajout_Pays('IUT de Blagnac2', 'Dept Info2', NULL, NULL, 'IU2', 'U2', 'Europ');
```

```
Le continent est inconnu


Procédure PL/SQL terminée.
```

➔ Superficie invalide

```
EXECUTE  Ajout_Pays('IUT de Blagnac2', 'Dept Info2', -45, NULL, 'IU2', 'U2', 'Europe');
```

```
La superficie est invalide

Procédure PL/SQL terminée.
```

➔ Population invalide

```
EXECUTE  Ajout_Pays('IUT de Blagnac2', 'Dept Info2', NULL, -45, 'IU2', 'U2', 'Europe');
```

```
La superficie est invalide

Procédure PL/SQL terminée.
```

3. Conclusion

Ce TP01 a pour but de mettre en application ce que nous avons déjà vu au cours de l'année précédente pour réviser, c'est à dire la manipulation de base de données grâce au langage SQL (Structured Query Langage) et le fonctionnement de blocs PL/SQL.

Durant ce TP, on a pu utiliser la requête SQL « SELECT » sur diverses tables pour observer leur contenu puis nous avons créé une procédure stockée en PL/SQL qui permet l'insertion d'un pays dans la table « Pays ».

Ce TP01 nous a permis de nous rappeler nos connaissances sur la gestion des bases de donnés en SQL et PL/SQL.