

## TP2 : PL/SQL (séance 1)

### 1. Schéma de base de données

---

Nous allons travailler sur une version de la base de données Commune (créée et peuplée la semaine dernière). Le schéma relationnel vous est rappelé ci-dessous :

- Region(**reg varchar(4)**, chef\_lieu varchar(46), nom\_reg varchar(30))
- Departement(reg varchar(4), **dep varchar(4)**, chef\_lieu varchar(46), nom\_dep varchar(30))  
avec Departement(reg)  $\subseteq$  Region(reg)
- Commune(dep varchar(4), com varchar(4), article varchar(4), com\_nom varchar(46), longitude float, latitude float, pop\_1975 float, pop\_1976 float, pop\_1977 float, pop\_1978 float, pop\_1979 float, pop\_1980 float, pop\_1981 float, pop\_1982 float, pop\_1983 float, pop\_1984 float, pop\_1985 float, pop\_1986 float, pop\_1987 float, pop\_1988 float, pop\_1989 float, pop\_1990 float, pop\_1991 float, pop\_1992 float, pop\_1993 float, pop\_1994 float, pop\_1995 float, pop\_1996 float, pop\_1997 float, pop\_1998 float, pop\_1999 float, pop\_2000 float, pop\_2001 float, pop\_2002 float, pop\_2003 float, pop\_2004 float, pop\_2005 float, pop\_2006 float, pop\_2007 float, pop\_2008 float, pop\_2009 float, pop\_2010 float, **code\_insee varchar(4)**)  
avec Commune(reg)  $\subseteq$  Region(reg)

### 2. Premiers triggers

---

1. Construisez un trigger qui vérifie que la population de 2010 (pop\_2010 de Commune) est toujours positive (ordres insert et update).
2. Les triggers ont souvent une écriture très simple (règle ECA) et s'appuient sur des procédures qui en masquent la complexité. Vous reprendrez l'écriture du trigger ouvrable. Vous définirez une procédure JoursEtHeuresOuvrables sans argument qui vérifie que la date du jour n'est pas un samedi ni un dimanche et qui renvoie un message d'erreur autrement. Vous redéfinirez le trigger ouvrable qui fera appel à cette procédure, dans le contexte de la table Region. Vous en testerez les effets.
3. Les triggers ont souvent un rôle de monitoring auprès des administrateurs de bases de données (indicateurs de la bonne santé des bases de données dont ils ont la charge). Vous créerez une table historique (dateOperation, nomUsager, typeOperation) qui va permettre de conserver la trace de toutes les opérations réalisées sur la table *Region*. Vous créerez le trigger qui va permettre d'alimenter cette table.  
Le nom de l'usage et la date système sont connus au travers des descripteurs USER et SYSDATE (cf. select user, sysdate from dual;)
4. Lors de la définition de contraintes de clés étrangères, Oracle autorise certaines fonctionnalités telle que la suppression en cascade des tuples dépendants au travers de la syntaxe réservée *on delete cascade*. A la différence de PostgreSQL, Oracle ne permet pas l'exploitation de la syntaxe

*on update cascade* qui permettrait de modifier les tuples dépendants en conséquence.

Vous construirez un trigger nommé *cascade* qui porte sur la table fonction et qui se charge à chaque évènement de suppression ou de modification d'une région (reg) dans *Region* de supprimer ou de modifier dans la table *Département*, les tuples de département dépendants de cette région (pensez à désactiver la contrainte de clé étrangère sur le numéro de département dans *Commune*).

## 2.1 Curseurs

1. Vous écrirez un curseur implicite qui permet d'afficher pour le département de l'Hérault (34), le nombre d'habitants en 2010 (somme de tous les habitants de toutes les communes de l'Hérault).
2. Vous écrirez un curseur explicite qui permet de retourner le nombre d'habitants en 2010 pour tous les départements français

## 2.2 Fonctions ou Procédures

1. Vous écrirez une fonction (ou une procédure) qui prend en entrée un numéro de département ainsi qu'une année (comprise entre 1975 et 2010) et qui retournera le nombre d'habitants du département sur l'année considérée (pensez à gérer les exceptions possibles).
2. Vous écrirez une fonction (ou une procédure) qui renvoie le nombre (et le pourcentage) d'utilisateurs connectés à la base. A cet effet, vous exploiterez les requêtes suivantes qui expriment respectivement : donner le nom des usagers "non système" de la base de données et donner le nom des usagers (qui en possèdent un) connectés.

```
select username from dba_users where username <> '%SYS%'
select username from v$session where username is not null
```

## 2.3 Objets du schéma

Vous exploiterez les tables du méta-schéma : *USER\_TRIGGERS*, *USER\_PROCEDURES*, *USER\_OBJECTS* et *USER\_SOURCE* pour avoir une vue d'ensemble sur les différents objets PL/SQL que vous venez d'ajouter à votre schéma. Vous supprimerez les déclencheurs construits.