

TP Base de Données UE FMIN206

1. Connexion

Pour se connecter à Oracle

```
sqlplus /@venus/MASTER.info
```

2. Modèle

Nous partirons du diagramme de classes UML simple ci-dessous.

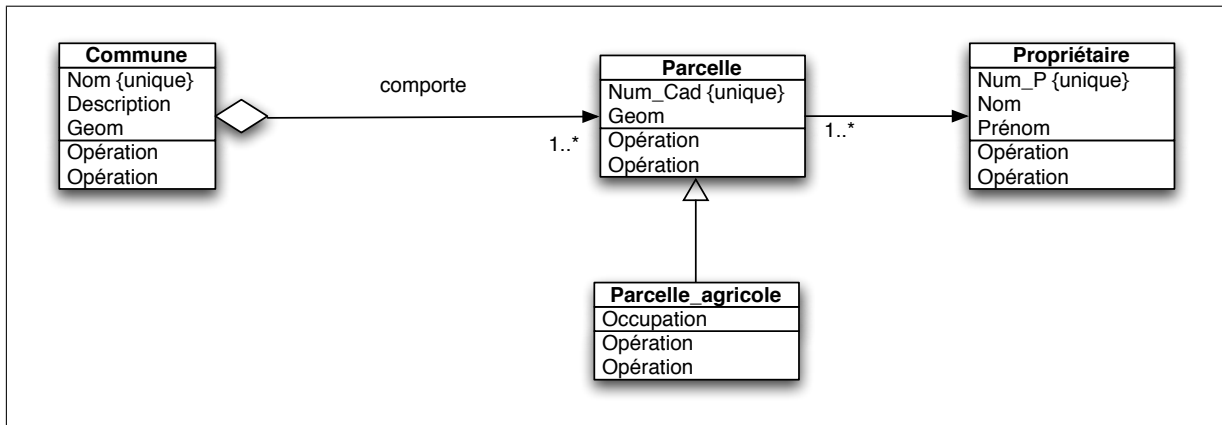


FIGURE 1 – Modèle

3. Premier pas : création de types et de tables objet utilisant ces types

En respectant la syntaxe ci-dessous,

```
create [or replace] type <nom type> AS Object
(col1 type1, col2 type2, ...)
/
```

1. Créer les types correspondants aux classes Propriétaire(Propriétaire.T), Parcelle(Parcelle.T) et Commune (Commune.T) . Vérifier la description de ces types (desc) et vérifier les descriptions au sein des tables de la métabase USER_TYPES et USER_TYPE_ATTRS.
2. Créer la table objet Parcelles définie sur le type Parcelle.T en ajoutant les contraintes suivantes :
 - la clé primaire est définie sur Num_{Cad}, Vérifier en consultant USER_OBJECT_TABLES et USER.Constraints.
3. Insérer quelques objets dans la table Parcelles (100, 110, 120, 130), vérifier que les contraintes sont opérantes.
4. Interrogations
 - Donner la totalité des informations de toutes les parcelles
 - Donner le numéro et le propriétaire de toutes les parcelles
 - Donner les références de toutes les parcelles

4. Deuxième étape : les collections

D'après le modèle une commune comporte plusieurs parcelles.

1. Créer le type correspondant à la classe Parcelle (Parcelle_T). Vérifier la description de ce types (desc) et vérifier la description au sein des tables de la métabase USER_TYPES et USER_TYPE_ATTRS.
2. Pour représenter la collection de parcelles d'une commune, deux solutions sont possibles correspondant aux types collections possibles dans Oracle : les collections illimitées qui seront utilisées comme tables imbriquées (nested tables) et les tableaux prédimensionnés (varray).
 - solution 1 Créer un type groupeParcelles_T par la directive


```
create [or replace] type <nom type collection> AS Table OF <nom type>
/
```
 - solution 2 Créer un type groupeParcellebis_T par la directive


```
create [or replace] type <nom type collection> AS VARRAY(dim) OF <nom type>
/
```
3. Créer ensuite des tables Communes relationnelles utilisant ces collections (en prévoyant les contraintes de clés primaires).
4. Insertions (à réaliser dans les divers cas de figures)

Insérer des communes et leur collection de parcelles, par exemple

```
PARIS, capitale
et la collection de parcelles
324 ; 325; 326; 327 etc...
```

5. Interrogations (à réaliser dans les divers cas de figures)
 - Donner le nom et la collection de parcelles de chaque commune
6. solution 2 Créer un type groupebisParcelles_T par la directive


```
create [or replace] type <nom type collection> AS VARRAY(dim) OF <nom type>
/
```

Créer ensuite des tables Communes relationnelles utilisant ces collections (en prévoyant les contraintes de clés primaires).

5. Troisième étape : la spécialisation et les références

Nous allons traiter maintenant, le cas de la classe Parcelle_{agricole} qui spécialise Parcelle et qui référence un Propriétaire.

1. Le type Parcelle_T doit être modifié (la clause NOT FINAL doit être ajoutée pour permettre la spécialisation). Un type ne peut être modifié si des tables ou d'autres types l'utilisent. Vous devrez donc supprimer la table Parcelles afin de pouvoir modifier le type Parcelle_T (vous pourrez bien sûr la recréer par la suite).
2. Créer le type Parcelle_Agricole_T selon la syntaxe

```
Create type <nomtype> UNDER <nomtypegeneral>
(col1 type 1, ...)
Not final /Final
/
```

Créer une table objet Parcelles_Agri

Réaliser des insertions dans cette table. On peut commencer à insérer des valeurs correspondant à des valeurs de la table Parcelles_Agri et initialiser le propriétaire à la valeur NULL. Ensuite pour une parcelle agricole donnée on peut faire une mise à jour de la table Parcelles_Agri en lui affectant un des propriétaires préalablement créés (via la référence objet correspondante).

Interrogations diverses

6. Création de base spatiale

Nous allons utiliser des types géométriques prédéfinis.

Le type objet SDO_GEOMETRY

```
CREATE TYPE sdo_geometry AS OBJECT (
  SDO_GTYPE NUMBER,
  SDO_SRID NUMBER,
  SDO_POINT SDO_POINT_TYPE,
  SDO_ELEM_INFO SDO_ELEM_INFO_ARRAY,
  SDO_ORDINATES SDO_ORDINATE_ARRAY);
```

Desc SDO_GEOMETRY

Name	Null?	Type
SDO_GTYPE		NUMBER
SDO_SRID		NUMBER
SDO_POINT		MDSYS.SDO_POINT_TYPE
SDO_ELEM_INFO		MDSYS.SDO_ELEM_INFO_ARRAY
SDO_ORDINATES		MDSYS.SDO_ORDINATE_ARRAY

Ce type est défini à partir d'autres types objets

```
CREATE TYPE sdo_point_type AS OBJECT (
  X NUMBER,
  Y NUMBER,
  Z NUMBER);
```

```
CREATE TYPE sdo_elem_info_array AS VARRAY (1048576) of NUMBER;
```

```
CREATE TYPE sdo_ordinate_array AS VARRAY (1048576) of NUMBER;
```

Vous créez la base spatiale simple correspondant au schéma suivant

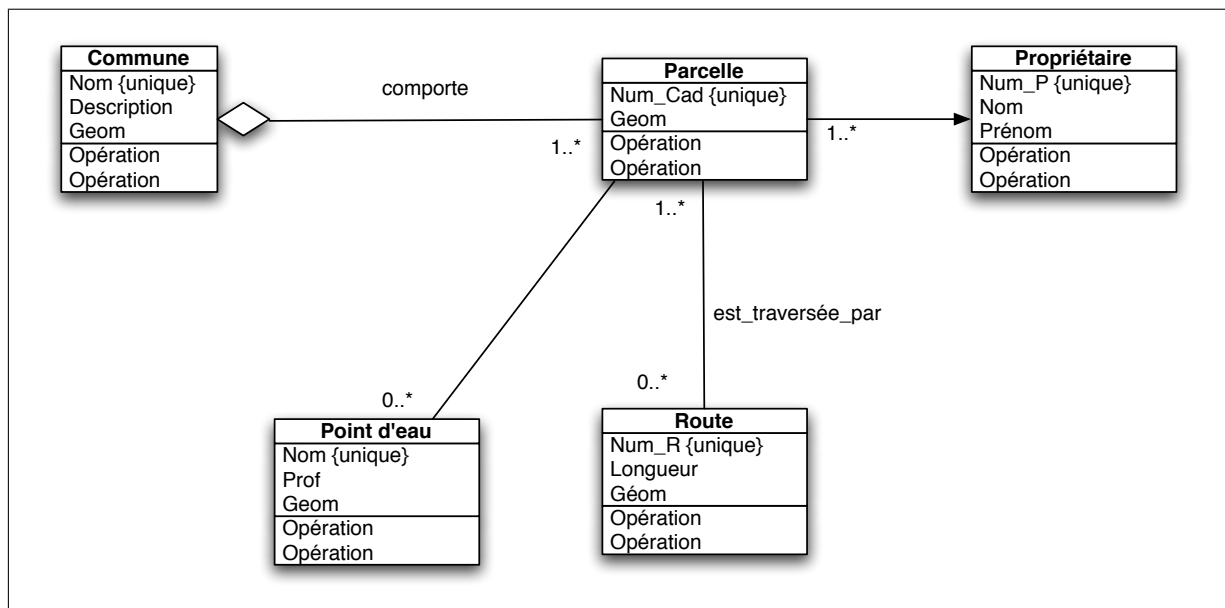


FIGURE 2 – Modèle pour BD spatiale

à partir du script donné (SCRIPTBDS). qui correspond en gros à cette vision
Effectuez ensuite des interrogations diverses :

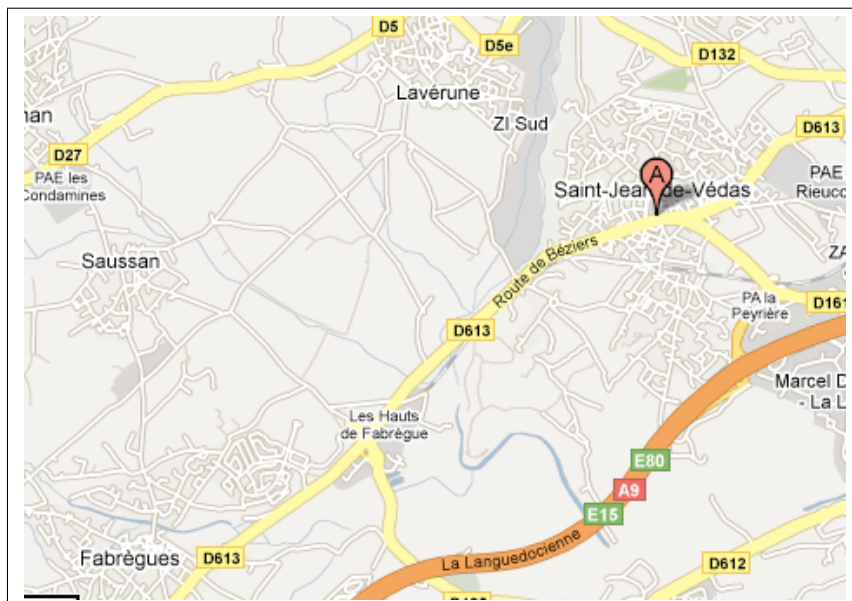


FIGURE 3 – Carte

- Création de zones tampon
SDO_GEOM.SDO_BUFFER
- Utilisation de fonction d'analyse spatiale
SDO_GEOM.SDO_UNION
SDO_GEOM.SDO_INTERSECTION
etc.
- Calculs surface, distance, longueur
SDO_GEOM.SDO_AREA
SDO_GEOM.SDO_DISTANCE
SDO_GEOM.SDO_LENGTH
- Détermination de relations
SDO_GEOM.RELATE

et bien d'autres ... (voir document Oracle)