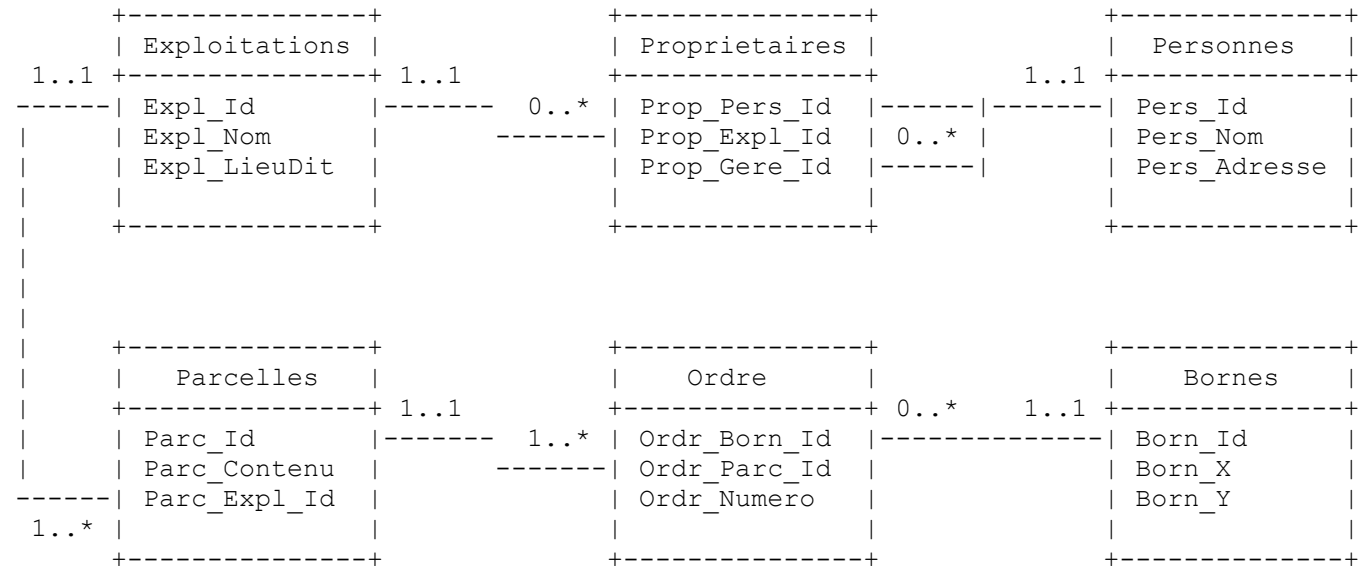
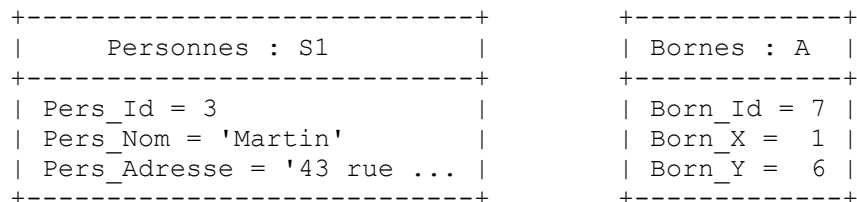


# Cadastre

1) Donner le diagramme de classe associé à cet énoncé.



2) Donner un exemple de diagramme d'objet associé à cet énoncé.



Exploitations : E1	Proprietaires : R1
Expl_Id = 0	Prop_Pers_Id = 3
Expl_Nom = 'Chateau Margaux'	Prop_Expl_Id = 0
Expl_LieuDit = 'Bord de Gironde'	Prop_Gere_Id = 1

Parcelles : P1	Ordre : N1
Parc_Id = 2	Ordr_Born_Id = 7
Parc_Contenu = 'Bois'	Ordr_Parc_Id = 2
Parc_Expl_Id = 0	Ordr_Numero = 1

### 3) Donner les ordres SQL de création des tables de la base de données.

```
CREATE TABLE Bornes (
  Born_Id          smallint UNIQUE NOT NULL,
  Born_X           smallint,
  Born_Y           smallint,
  CONSTRAINT pk_Born_Id PRIMARY KEY (Born_Id)
);
```

```
CREATE TABLE Personnes (
  Pers_Id          smallint UNIQUE NOT NULL,
  Pers_Nom         varchar(30),
  Pers_Adresse     varchar(60) UNIQUE,
  CONSTRAINT pk_Pers_Id PRIMARY KEY (Pers_Id)
);
```

```
CREATE TABLE Exploitations (  
    Expl_Id          smallint UNIQUE NOT NULL,  
    Expl_Nom         varchar(30),  
    Expl_LieuDit     varchar(30),  
    CONSTRAINT pk_Expl_Id PRIMARY KEY (Expl_Id)  
);
```

```
CREATE TABLE Proprietaires (  
    Prop_Pers_Id     smallint NOT NULL,  
    Prop_Expl_Id     smallint NOT NULL,  
    Prop_Gere_Id     smallint NOT NULL,  
    CONSTRAINT pk_Propt PRIMARY KEY (Prop_Pers_Id, Prop_Expl_Id, Prop_Gere_Id),  
    CONSTRAINT fk_Prop_Pers_Id FOREIGN KEY (Prop_Pers_Id) REFERENCES Personnes (Pers_Id),  
    CONSTRAINT fk_Prop_Gere_Id FOREIGN KEY (Prop_Gere_Id) REFERENCES Personnes (Pers_Id),  
    CONSTRAINT fk_Prop_Expl_Id FOREIGN KEY (Prop_Expl_Id) REFERENCES Exploitations (Expl_Id)  
);
```

```
CREATE TABLE Parcelles (  
    Parc_Id          smallint UNIQUE NOT NULL,  
    Parc_Contenu     varchar(15),  
    Parc_Expl_Id     smallint NOT NULL,  
    CONSTRAINT pk_Parc_Id PRIMARY KEY (Parc_Id),  
    CONSTRAINT ck_Parc_Contenu CHECK (Parc_Contenu IN ('Terre', 'Bois', 'Habitation') ),  
    CONSTRAINT fk_Parc_Expl_Id FOREIGN KEY (Parc_Expl_Id) REFERENCES Exploitations (Expl_Id)  
);
```

```
CREATE TABLE Ordre (  
    Ordr_Born_Id     smallint NOT NULL,  
    Ordr_Parc_Id     smallint NOT NULL,  
    Ordr_Numero      smallint,  
    CONSTRAINT pk_Ordr PRIMARY KEY (Ordr_Born_Id, Ordr_Parc_Id),  
    CONSTRAINT fk_Ordr_Born_Id FOREIGN KEY (Ordr_Born_Id) REFERENCES Bornes (Born_Id),  
    CONSTRAINT fk_Ordr_Parc_Id FOREIGN KEY (Ordr_Parc_Id) REFERENCES Parcelles (Parc_Id)  
);
```

```

+-----+
| Table : Personnes |
+-----+
| Pers_Id | Pers_Nom | Pers_Adresse |
+-----+
| 0 | Martin | 43 rue de la Devise 33000 Bordeaux |
| 1 | Dupont | 34 rue Ferrère 33000 Bordeaux |
| 2 | Petit | 72 rue de Rivière 33000 Bordeaux |
| 3 | Rivel | 18 rue des Orangers 33000 Bordeaux |
| 4 | Villard | 21 rue Beaumartin 33700 Mérignac |
| 5 | Leroy | 3 rue Gabriel Fauré 33520 Bruges |
| 6 | Blanc | 6 rue Moulerin 33530 Bassens |
+-----+

```

```

+-----+
| Table : Proprietaires |
+-----+
| Prop_Pers_Id | Prop_Expl_Id | Prop_Gere_Id |
+-----+
| 1 | 1 | 0 |
| 4 | 0 | 4 |
| 1 | 0 | 4 |
| 2 | 1 | 0 |
| 3 | 0 | 4 |
| 5 | 0 | 4 |
| 4 | 1 | 0 |
+-----+

```

```

+-----+
| Table : Exploitations |
+-----+
| Expl_Id | Expl_Nom | Expl_LieuDit |
+-----+
| 0 | Chateau Margaux | Bord de Gironde |
| 1 | Chateau Yquem | Coteaux de Gironde |
+-----+

```

```

+-----+
| Table : Parcelles |
+-----+
| Parc_Id | Parc_Contenu | Parc_Expl_Id |
+-----+
|      0 | Habitation   |           0 |
|      1 | Habitation   |           1 |
|      2 | Terre        |           0 |
|      3 | Terre        |           1 |
|      4 | Bois         |           0 |
+-----+

```

```

+-----+
| Table : Ordre |
+-----+
| Ordre_Born_Id | Ordre_Parc_Id | Ordre_Numero |
+-----+
|           0 |           0 |           1 | % P1 & A
|           7 |           0 |           2 | % P1 & H
|           6 |           0 |           3 | % P1 & G
|           1 |           0 |           4 | % P1 & B
|           1 |           1 |           1 | % P2 & B
|           6 |           1 |           2 | % P2 & G
|           5 |           1 |           3 | % P2 & F
|           4 |           1 |           4 | % P2 & E
|           3 |           1 |           5 | % P2 & D
|           2 |           1 |           6 | % P2 & C
|           0 |           2 |           1 | % P3 & A
|           8 |           2 |           2 | % P3 & I
|          12 |           2 |           3 | % P3 & M
|          11 |           2 |           4 | % P3 & L
|           5 |           2 |           5 | % P3 & F
|           6 |           2 |           6 | % P3 & G
|           7 |           2 |           7 | % P3 & H
|           2 |           3 |           1 | % P4 & C
|           3 |           3 |           2 | % P4 & D
|           4 |           3 |           3 | % P4 & E
|           5 |           3 |           4 | % P4 & F
|          11 |           3 |           5 | % P4 & L
|          10 |           3 |           6 | % P4 & K
|           0 |           4 |           1 | % P5 & A
|           1 |           4 |           2 | % P5 & B

```

2	4	3	% P5 & C
10	4	4	% P5 & K
9	4	5	% P5 & J
8	4	6	% P5 & I

```

+-----+
| Table : Bornes |
+-----+
| Born_Id | Born_X | Born_Y |
+-----+
| 0 | 1 | 6 | % A
| 1 | 5 | 11 | % B
| 2 | 10 | 13 | % C
| 3 | 9 | 10 | % D
| 4 | 14 | 9 | % E
| 5 | 12 | 6 | % F
| 6 | 7 | 6 | % G
| 7 | 3 | 4 | % H
| 8 | 0 | 0 | % I
| 9 | 0 | 14 | % J
| 10 | 16 | 14 | % K
| 11 | 16 | 0 | % L
| 12 | 10 | 0 | % M
+-----+

```

#### 4) Donner les ordres SQL d'insertion des données dans les tables.

```

INSERT INTO Bornes VALUES ( 0, 1, 6); /* A */
INSERT INTO Bornes VALUES ( 1, 5, 11); /* B */
INSERT INTO Bornes VALUES ( 2, 10, 13); /* C */
INSERT INTO Bornes VALUES ( 3, 9, 10); /* D */
INSERT INTO Bornes VALUES ( 4, 14, 9); /* E */
INSERT INTO Bornes VALUES ( 5, 12, 6); /* F */
INSERT INTO Bornes VALUES ( 6, 7, 6); /* G */
INSERT INTO Bornes VALUES ( 7, 3, 4); /* H */
INSERT INTO Bornes VALUES ( 8, 0, 0); /* I */
INSERT INTO Bornes VALUES ( 9, 0, 14); /* J */
INSERT INTO Bornes VALUES (10, 16, 14); /* K */
INSERT INTO Bornes VALUES (11, 16, 0); /* L */
INSERT INTO Bornes VALUES (12, 10, 0); /* M */

```

```
INSERT INTO Personnes VALUES (0, 'Martin', '43 rue de la Devise 33000 Bordeaux');
INSERT INTO Personnes VALUES (1, 'Dupont', '34 rue Ferrère 33000 Bordeaux');
INSERT INTO Personnes VALUES (2, 'Petit', '72 rue de Rivière 33000 Bordeaux');
INSERT INTO Personnes VALUES (3, 'Rivel', '18 rue des Orangers 33000 Bordeaux');
INSERT INTO Personnes VALUES (4, 'Villard', '21 rue Beaumartin 33700 Mérignac');
INSERT INTO Personnes VALUES (5, 'Leroy', '3 rue Gabriel Fauré 33520 Bruges');
INSERT INTO Personnes VALUES (6, 'Blanc', '6 rue Moulerin 33530 Bassens');
```

```
INSERT INTO Exploitations VALUES (0, 'Chateau Margaux', 'Bord de Gironde'); /* E1 */
INSERT INTO Exploitations VALUES (1, 'Chateau Yquem', 'Coteaux de Gironde'); /* E2 */
```

```
INSERT INTO Parcelles VALUES (0, 'Habitation', 0); /* P1 */
INSERT INTO Parcelles VALUES (1, 'Habitation', 1); /* P2 */
INSERT INTO Parcelles VALUES (2, 'Terre', 0); /* P3 */
INSERT INTO Parcelles VALUES (3, 'Terre', 1); /* P4 */
INSERT INTO Parcelles VALUES (4, 'Bois', 0); /* P5 */
```

```
INSERT INTO Proprietaires VALUES (1, 1, 0);
INSERT INTO Proprietaires VALUES (4, 0, 4);
INSERT INTO Proprietaires VALUES (1, 0, 4);
INSERT INTO Proprietaires VALUES (2, 1, 0);
INSERT INTO Proprietaires VALUES (3, 0, 4);
INSERT INTO Proprietaires VALUES (5, 0, 4);
INSERT INTO Proprietaires VALUES (4, 1, 0);
```

```
INSERT INTO Ordre VALUES ( 0, 0, 1); /* P1 & A */
INSERT INTO Ordre VALUES ( 7, 0, 2); /* P1 & H */
INSERT INTO Ordre VALUES ( 6, 0, 3); /* P1 & G */
INSERT INTO Ordre VALUES ( 1, 0, 4); /* P1 & B */
```

```
INSERT INTO Ordre VALUES ( 1, 1, 1); /* P2 & B */
INSERT INTO Ordre VALUES ( 6, 1, 2); /* P2 & G */
INSERT INTO Ordre VALUES ( 5, 1, 3); /* P2 & F */
INSERT INTO Ordre VALUES ( 4, 1, 4); /* P2 & E */
INSERT INTO Ordre VALUES ( 3, 1, 5); /* P2 & D */
INSERT INTO Ordre VALUES ( 2, 1, 6); /* P2 & C */
```

```

INSERT INTO Ordre VALUES ( 0, 2, 1); /* P3 & A */
INSERT INTO Ordre VALUES ( 8, 2, 2); /* P3 & I */
INSERT INTO Ordre VALUES (12, 2, 3); /* P3 & M */
INSERT INTO Ordre VALUES (11, 2, 4); /* P3 & L */
INSERT INTO Ordre VALUES ( 5, 2, 5); /* P3 & F */
INSERT INTO Ordre VALUES ( 6, 2, 6); /* P3 & G */
INSERT INTO Ordre VALUES ( 7, 2, 7); /* P3 & H */

INSERT INTO Ordre VALUES ( 2, 3, 1); /* P4 & C */
INSERT INTO Ordre VALUES ( 3, 3, 2); /* P4 & D */
INSERT INTO Ordre VALUES ( 4, 3, 3); /* P4 & E */
INSERT INTO Ordre VALUES ( 5, 3, 4); /* P4 & F */
INSERT INTO Ordre VALUES (11, 3, 5); /* P4 & L */
INSERT INTO Ordre VALUES (10, 3, 6); /* P4 & K */

INSERT INTO Ordre VALUES ( 0, 4, 1); /* P5 & A */
INSERT INTO Ordre VALUES ( 1, 4, 2); /* P5 & B */
INSERT INTO Ordre VALUES ( 2, 4, 3); /* P5 & C */
INSERT INTO Ordre VALUES (10, 4, 4); /* P5 & K */
INSERT INTO Ordre VALUES ( 9, 4, 5); /* P5 & J */
INSERT INTO Ordre VALUES ( 8, 4, 6); /* P5 & I */

```

### 5) Quelle est l'adresse du gérant de l'exploitation 'Chateau Margaux' ?

```

select distinct Pers_Adresse
from Personnes, Proprietaires, Exploitations
where Expl_Nom = 'Chateau Margaux' and
      Expl_Id = Prop_Expl_Id and
      Prop_Gere_Id = Pers_Id

```

### 6) Quelle est l'adresse des propriétaires de l'exploitation 'Chateau Margaux' ?

```

select Pers_Adresse
from Personnes, Proprietaires, Exploitations
where Expl_Nom = 'Chateau Margaux' and
      Expl_Id = Prop_Expl_Id and
      Prop_Pers_Id = Pers_Id

```

### 7) Quelles sont les parcelles de l'exploitation 'Chateau Margaux' ?

```
select Parc_Id
from Parcelles, Exploitations
where Expl_Nom = 'Chateau Margaux' and
      Expl_Id = Parc_Expl_Id
```

### 8) Quelles sont les coordonnées des bornes des parcelles de l'exploitation 'Chateau Margaux' ?

```
select distinct Born_X, Born_Y
from Bornes, Ordre, Parcelles, Exploitations
where Expl_Nom = 'Chateau Margaux' and
      Expl_Id = Parc_Expl_Id and
      Parc_Id = Ordr_Parc_Id and
      Ordr_Born_Id = Born_Id
```

### 9) De combien de parcelles est composée 'Chateau Margaux' ?

```
select Count(Parc_Id)
from Parcelles, Exploitations
where Expl_Nom = 'Chateau Margaux' and
      Expl_Id = Parc_Expl_Id
```

### 10) Quels sont les segments de la parcelle P3 ?

```
select Ordr_Born_Id
from Ordre
where Ordr_Parc_Id = 2
order by Ordr_Numero
```

### 11) Quelles sont les parcelles qui jouxtent la parcelle P3 ?

```
select distinct O2.Ordr_Parc_Id
from Ordre O1, Ordre O2
where O1.Ordr_Parc_Id = 2 and
      O2.Ordr_Parc_Id <> 2 and
      O1.Ordr_Born_Id = O2.Ordr_Born_Id and
      O1.Ordr_Numero <> O2.Ordr_Numero
```

### 12) Quelles sont les personnes qui gèrent l'une de leur propriétés ?

```
select Pers_Nom
from Personnes, Proprietaires
where Pers_Id = Prop_Pers_Id and
      Prop_Pers_Id = Prop_Gere_Id
```

### 13) Quelles sont les personnes qui gèrent une exploitation dont ils ne sont pas propriétaires ?

```
select Pers_Nom
from Personnes
where Pers_Id in (
  select Prop_Gere_Id
  from Proprietaires
  MINUS
  select Prop_Pers_Id
  from Proprietaires)
```

### 14) Nom des bornes du premier segment de la parcelle P2.

```
select O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Ordre O1, Ordre O2
where O1.Ordre_Parc_Id = 1 and
      O2.Ordre_Parc_Id = 1 and
      O1.Ordre_Numero = 1 and
      O2.Ordre_Numero = 2
```

### 15) Nom des bornes du dernier segment de la parcelle P2.

```
select O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Ordre O1, Ordre O2
where O1.Ordre_Parc_Id = 1 and
      O2.Ordre_Parc_Id = 1 and
      O1.Ordre_Numero = 1 and
      O2.Ordre_Numero in (
  select max(Ordre_Numero)
  from Ordre
  where Ordre_Parc_Id = 1
)
```

## 16) Nom des bornes délimitant chaque segment de la parcelle P2.

```
select O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Ordre O1, Ordre O2
where O1.Ordre_Parc_Id = 1 and
      O2.Ordre_Parc_Id = 1 and
      O2.Ordre_Numero = O1.Ordre_Numero+1
UNION
select O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Ordre O1, Ordre O2
where O1.Ordre_Parc_Id = 1 and
      O2.Ordre_Parc_Id = 1 and
      O2.Ordre_Numero = 1 and
      O1.Ordre_Numero in (
        select max(Ordre_Numero)
        from Ordre
        where Ordre_Parc_Id = 1
      )
)
```

## 17) Pour chaque segment d'une parcelle, donner son identifiant de parcelle et le nom des bornes délimitant le segment.

```
select O1.Ordre_Born_Id, O2.Ordre_Born_Id, O1.Ordre_Parc_Id
from Ordre O1, Ordre O2
where O1.Ordre_Parc_Id=O2.Ordre_Parc_Id and
      O2.Ordre_Numero = O1.Ordre_Numero+1
UNION
select O1.Ordre_Born_Id, O2.Ordre_Born_Id, O1.Ordre_Parc_Id
from Ordre O1, Ordre O2
where O1.Ordre_Parc_Id=O2.Ordre_Parc_Id and
      O2.Ordre_Numero = 1 and
      O1.Ordre_Numero in (
        select max(Ordre_Numero)
        from Ordre
        where Ordre_Parc_Id = O1.Ordre_Parc_Id
      )
)
```

Un polygone (une parcelle) est convexe si chacun de ses angles (internes) est inférieur à 180 degrés. La parcelle P1, par exemple est convexe : chacun de ses angles (internes) est inférieur à 180 degrés, en particulier l'angle de sommet H. La parcelle P3 est concave : l'angle de sommet H (interne à la parcelle P3) est supérieur à 180 degrés.

La formule, dite du produit croisé, permet de savoir si un angle d'un polygone est inférieur ou supérieur à 180 degrés. Cette formule est fonction :  
 - de l'ordonnancement des angles du polygone, que l'on supposera être dans le sens inverse des aiguilles d'une montre, à partir d'un angle pris au hasard,  
 - des coordonnées de l'angle du sommet précédent et du sommet suivant.

On suppose qu'un polygone a un angle S1 de coordonnées (x1, y1). Il est noté (S0, S1, S2) : en supposant que son sommet précédent de S1 est le sommet S0 de coordonnées (x0, y0), et que son sommet suivant de S1 est le sommet S2 de coordonnées (x2, y2). La formule du produit croisée est la suivante :

$$(x1 - x0) * (y2 - y0) - (x2 - x0) * (y1 - y0)$$

si la formule retourne une valeur positive, l'angle (S0, S1, S2) est inférieur à 180 degrés ; une valeur négative, l'angle (S0, S1, S2) est supérieur à 180 degrés.

**18) Appliquer la formule à l'angle (A, H, G) de la parcelle P1, et à l'angle (G, H, A) de la parcelle P3, afin de vérifier que le premier est concave et l'autre convexe.**

```
x, y
A : 1, 6
H : 3, 4
G : 7, 6
```

$$(A, H, G) \Rightarrow (3-1) * (6-6) - (7-1) * (4-6) = 2*0 - 6*(-2) = 0+12 = 12$$

$$(G, H, A) \Rightarrow (3-7) * (6-6) - (1-7) * (4-6) = (-4)*0 - (-6)*(-2) = 0-12 = -12$$

**19) Ecrire une requête qui retourne la valeur de l'angle (A, H, G) de la parcelle P1 et de celle de l'angle (G, H, A) de la parcelle P3.**

```
select (bH.Born_X-bA.Born_X) * (bG.Born_Y-bA.Born_Y) - (bG.Born_X-bA.Born_X) * (bH.Born_Y-bA.Born_Y) ,
       (bH.Born_X-bG.Born_X) * (bA.Born_Y-bG.Born_Y) - (bA.Born_X-bG.Born_X) * (bH.Born_Y-bG.Born_Y)
from Bornes bA, Bornes bH, Bornes bG
where bA.Born_Id = 0 and
      bH.Born_Id = 7 and
      bG.Born_Id = 6
```

**20) Ecrire une requête qui retourne pour chaque parcelle, son nom et le nom de son deuxième angle, c'est-à-dire le nom des 3 premiers sommets.**

```

select Parc_Id, O0.Ordre_Born_Id, O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2
where O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O0.Ordre_Numero = 1 and
      O1.Ordre_Numero = 2 and
      O2.Ordre_Numero = 3

```

**21) En supposant que vous ne connaissez pas le nombre de sommets délimitant la parcelle P1, écrire une requête qui retourne le nom de son premier angle ( nom du dernier sommet et des deux premiers sommets)**

```

select O0.Ordre_Born_Id, O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Ordre O0, Ordre O1, Ordre O2
where O0.Ordre_Parc_Id = 0 and
      O1.Ordre_Parc_Id = 0 and
      O2.Ordre_Parc_Id = 0 and
      O1.Ordre_Numero = 1 and
      O2.Ordre_Numero = 2 and
      O0.Ordre_Numero in (
        select max(Ordre_Numero)
        from Ordre
        where Ordre_Parc_Id = 0)

```

**22) Ecrire une requête qui retourne pour chaque parcelle, son nom et le nom de son premier angle (le nom du dernier sommet et des deux premiers sommets)**

```

select Parc_Id, O0.Ordre_Born_Id, O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2
where O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Numero = 1 and
      O2.Ordre_Numero = 2 and
      O0.Ordre_Numero in (
        select max(Ordre_Numero)
        from Ordre
        where Ordre_Parc_Id = O0.Ordre_Parc_Id)

```

### 23) Ecrire une requête qui retourne la liste des parcelles convexes.

```
/* Etape_1 : */

select Parc_Id, O0.Ordre_Born_Id, O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2
where O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Numero = O0.Ordre_Numero+1 and
      O2.Ordre_Numero = O1.Ordre_Numero+1

UNION

select Parc_Id, O0.Ordre_Born_Id, O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2
where O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Numero = 1 and
      O2.Ordre_Numero = 2 and
      O0.Ordre_Numero in (
        select max(Ordre_Numero)
        from Ordre
        where Ordre_Parc_Id = O0.Ordre_Parc_Id)

UNION

select Parc_Id, O0.Ordre_Born_Id, O1.Ordre_Born_Id, O2.Ordre_Born_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2
where O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Numero = 1 and
      O0.Ordre_Numero = O1.Ordre_Numero-1 and
      O1.Ordre_Numero in (
        select max(Ordre_Numero)
        from Ordre
        where Ordre_Parc_Id = O0.Ordre_Parc_Id)

/* Etape_2 : */

select Parc_Id
from Parcelles
where Parc_Id not in (
```

```

select Parc_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2, Bornes B0, Bornes B1, Bornes B2
where O0.Ordre_Born_Id = B0.Born_Id and
      O1.Ordre_Born_Id = B1.Born_Id and
      O2.Ordre_Born_Id = B2.Born_Id and
      O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Número = O0.Ordre_Número+1 and
      O2.Ordre_Número = O1.Ordre_Número+1 and
      (B1.Born_X-B0.Born_X) * (B2.Born_Y-B0.Born_Y) - (B2.Born_X-B0.Born_X) * (B1.Born_Y-B0.Born_Y) < 0 )

```

UNION

```

select Parc_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2, Bornes B0, Bornes B1, Bornes B2
where O0.Ordre_Born_Id = B0.Born_Id and
      O1.Ordre_Born_Id = B1.Born_Id and
      O2.Ordre_Born_Id = B2.Born_Id and
      O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Número = 1 and
      O2.Ordre_Número = 2 and
      O0.Ordre_Número in (
        select max(Ordre_Número)
        from Ordre
        where Ordre_Parc_Id = O0.Ordre_Parc_Id) and
      (B1.Born_X-B0.Born_X) * (B2.Born_Y-B0.Born_Y) - (B2.Born_X-B0.Born_X) * (B1.Born_Y-B0.Born_Y) < 0 )

```

UNION

```

select Parc_Id
from Parcelles, Ordre O0, Ordre O1, Ordre O2, Bornes B0, Bornes B1, Bornes B2
where O0.Ordre_Born_Id = B0.Born_Id and
      O1.Ordre_Born_Id = B1.Born_Id and
      O2.Ordre_Born_Id = B2.Born_Id and
      O0.Ordre_Parc_Id = Parc_Id and
      O1.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Parc_Id = Parc_Id and
      O2.Ordre_Número = 1 and
      O0.Ordre_Número = O1.Ordre_Número-1 and
      O1.Ordre_Número in (
        select max(Ordre_Número)
        from Ordre
        where Ordre_Parc_Id = O0.Ordre_Parc_Id) and

```

```
(B1.Born_X-B0.Born_X) * (B2.Born_Y-B0.Born_Y) - (B2.Born_X-B0.Born_X) * (B1.Born_Y-B0.Born_Y) < 0
)
```

#### 24) Ecrire une procédure en PL/SQL qui effectue le changement d'exploitation d'une parcelle.

```
/*
Il faut vérifier que :
- la parcelle existe
- l'anc et la new exploitation existe
- la parcelle appartient bien a l'anc exploitation
- dans l'anc exploitation il reste au moins une parcelle de type habitation
*/
```

```
CREATE OR REPLACE PROCEDURE changement_comp_exp
  (code_parcelle Parcelles.Parc_Id%TYPE,
  anc_propriete Exploitations.Expl_Nom%TYPE,
  new_propriete Exploitations.Expl_Nom%TYPE
  ) is
```

```
id          Parcelles.Parc_Id%TYPE;
contenu     Parcelles.Parc_Contenu%TYPE;
expl_anc    Exploitations.Expl_Nom%TYPE;
expl_new    Exploitations.Expl_Nom%TYPE;
flag INTEGER := 0;
```

```
BEGIN
  -- la parcelle existe ?
  select Parc_Id INTO id
  from Parcelles
  where Parc_Id = code_parcelle;

  -- l'ancienne exploitation existe ?
  flag := 1;
  select Expl_Nom INTO expl_anc
  from Exploitations
  where Expl_Nom = anc_propriete;

  -- la nouvelle exploitations existe ?
  flag := 2;
  select Expl_Nom INTO expl_new
  from Exploitations
```

```

where Expl_Nom = new_propriete;

-- la parcelle appartient bien a l'anc exploitation ?
flag := 3;
select Parc_Id INTO id
from Parcelles, Exploitations
where Parc_Id = code_parcelle
and Parc_Expl_Id = Expl_Id
and Expl_Nom = anc_propriete;

-- reste-t-il au moins une parcelle de type habitation dans l'ancienne exploitation
flag := 4;
select Parc_Contenu INTO contenu
from Parcelles
where Parc_Id = code_parcelle;
if (contenu='Habitation')
then select Parc_Id INTO id
      from Parcelles, Exploitations
      where Parc_Contenu = 'Habitation'
      and Parc_Id <> code_parcelle
      and Parc_expl_Id = (select Parc_Expl_Id
                          from Parcelles
                          where Parc_Id = code_parcelle);
end if;

-- Mise a jour de la parcelle concernee
update Parcelles
set Parc_Expl_Id = (select Expl_Id from Exploitations where Expl_Nom = new_propriete)
where Parc_Id = code_parcelle;
dbms_output.put_line('Modification effectuee : la parcelle ' || id || ' appartient a ' || new_propriete);

EXCEPTION
when NO_DATA_FOUND then
  IF(flag=0) THEN dbms_output.put_line(code_parcelle || ' : parcelle non trouvee');
  ELSE IF(flag=1) THEN dbms_output.put_line(anc_propriete || ' : exploitation inconnue!');
  ELSE IF(flag=2) THEN dbms_output.put_line(new_propriete || ' : exploitation inconnue!');
  ELSE IF(flag=3) THEN dbms_output.put_line('La parcelle ' || code_parcelle || ' n appartient pas a ' ||
                                          anc_propriete);
  ELSE dbms_output.put_line('Impossible : la parcelle ' || code_parcelle ||
                              ' est la seule Habitation de l exploitation ' || anc_propriete);
END IF;
END IF;

```

```

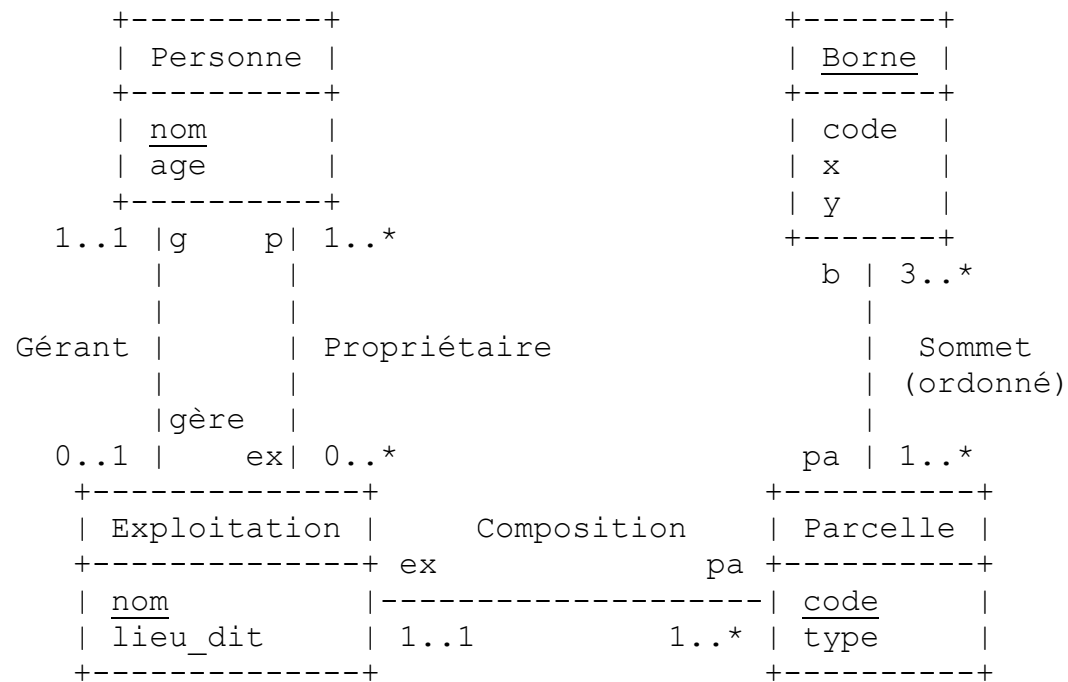
END IF;
END IF;

END;
/

exec changement_comp_exp(8, 'Chateau Margaux', 'test');
exec changement_comp_exp(1, 'Chateau Margaux', 'test');
exec changement_comp_exp(1, 'Chateau Margaux', 'Chateau Yquem');
exec changement_comp_exp(0, 'Chateau Margaux', 'Chateau Yquem');
exec changement_comp_exp(4, 'Chateau Margaux', 'Chateau Yquem');
exec changement_comp_exp(4, 'Chateau Yquem', 'Chateau Margaux');

```

**25) Construire le diagramme de classe sous USE avec les contraintes suivant la solution suivante.**



```
+-----+
|      |
|  cadastre.use  |
|      |
+-----+
```

```
-----
-- Personne --
-----
```

```
class Personne
attributes
  nom : String
  ddn : Integer
  ville : String
operations
  age() : Integer = 2003 - self.ddn
end
```

```
-----
-- Exploitation --
-----
```

```
class Exploitation
attributes
  nom : String
  lieu_dit : String
end
```

```
-----
-- Parcelle --
-----
```

```
class Parcelle
attributes
  code : String
  type : String
operations
  bornes_ordonnees() : Sequence( Borne ) = self.les_sommets->select( s | s.numero = 1 ).la_borne->asSequence

  trois_bornes_successives( num : Integer ) : Sequence( Borne ) =

if num <= self.les_sommets->size() - 2 then
```

```

self.les_sommets->select( s | s.numero = num  ).la_borne->asSequence->union(
self.les_sommets->select( s | s.numero = num+1 ).la_borne->asSequence )->union(
self.les_sommets->select( s | s.numero = num+2 ).la_borne->asSequence )

else if num = self.les_sommets->size() - 1
  then self.les_sommets->select( s | s.numero = num                ).la_borne->asSequence->union(
        self.les_sommets->select( s | s.numero = self.les_sommets->size() ).la_borne->asSequence )->union(
        self.les_sommets->select( s | s.numero = 1                      ).la_borne->asSequence )
  else self.les_sommets->select( s | s.numero = num                ).la_borne->asSequence->union(
        self.les_sommets->select( s | s.numero = 1                      ).la_borne->asSequence )->union(
        self.les_sommets->select( s | s.numero = 2                      ).la_borne->asSequence )
  endif
endif
end

```

end

```

-----
-- Borne --
-----

```

```

class Borne
attributes
  x : Integer
  y : Integer
operations
  angle( b1 : Borne, b2 : Borne ) : Integer =

      ( b1.x - self.x ) * ( b2.y - self.y ) -
      ( b2.x - self.x ) * ( b1.y - self.y )
end

```

```

-----
-- Sommet --
-----

```

```

class Sommet
attributes
  numero : Integer
end

```

```

-----
-- ASSOCIATIONS --
-----

```

```
association Gerant between
  Personne[1]      role est_geree_par
  Exploitation[0..1] role est_gerant_de
end
```

```
association Proprietaire between
  Personne[1..*]   role les_proprio
  Exploitation[*] role les_proprietes
end
```

```
association Composition between
  Exploitation[1] role l_exploitation
  Parcelle[1..*]  role les_parcelles
end
```

```
association ParcelleSommet between
  Parcelle[1]  role la_parcelle
  Sommet[3..*] role les_sommets
end
```

```
association SommetBorne between
  Sommet[1..*] role les_sommets
  Borne[1]     role la_borne
end
```

```
-----
-- CONTRAINTES --
-----
```

```
constraints
context Exploitation
inv au_moins_parc_habitation : self.les_parcelles->exists(p|p.type = 'habitation')
inv unique_expl : Exploitation.allInstances->forAll(e1,e2|e1<>e2 implies e1.nom <> e2.nom)
```

```
context Personne
inv unique_pers : Personne.allInstances->forAll(p1,p2|p1<>p2 implies p1.nom <> p2.nom)
```

```
context Parcelle
inv unique_parc : Parcelle.allInstances->forAll(c1,c2|c1<>c2 implies c1.code <> c2.code)
```

```
context Borne
inv unique_born : Borne.allInstances->forall(b1,b2|b1<>b2 implies b1.code <> b2.code)
```

## 26) Construire le diagramme d'objet sous USE.

```
+-----+
|      |
|  cadastre.cmd  |
|      |
+-----+

-----
-- Personne --
-----

!create pe1 : Personne
!set   pe1.nom = 'dupont'
!set   pe1.ddn = 1980
!set   pe1.ville = 'toulouse'

!create pe2 : Personne
!set   pe2.nom = 'dulong'
!set   pe2.ddn = 1970
!set   pe2.ville = 'toulouse'

!create pe3 : Personne
!set   pe3.nom = 'ducourt'
!set   pe3.ddn = 1975
!set   pe3.ville = 'bordeaux'

-----
-- Exploitation --
-----

!create ex1 : Exploitation
!set   ex1.nom = 'Chateau_Margaux'
!set   ex1.lieu_dit = 'bord_de_Gironde'

!create ex2 : Exploitation
!set   ex2.nom = 'Charteau_Rotchild'
!set   ex2.lieu_dit = 'bord_de_Gironde'
```

```
-----  
-- Parcelle --  
-----  
!create p1 : Parcelle  
!set   p1.code = 'p1'  
!set   p1.type = 'habitation'  
  
!create p2 : Parcelle  
!set   p2.code = 'p2'  
!set   p2.type = 'bois'  
  
!create p3 : Parcelle  
!set   p3.code = 'p3'  
!set   p3.type = 'terre'  
  
!create p4 : Parcelle  
!set   p4.code = 'p4'  
!set   p4.type = 'habitation'  
  
!create p5 : Parcelle  
!set   p5.code = 'p5'  
!set   p5.type = 'terre'  
  
-----  
-- Borne --  
-----  
!create bA : Borne  
!set   bA.x = 1  
!set   bA.y = 6  
  
!create bB : Borne  
!set   bB.x = 5  
!set   bB.y = 11  
  
!create bC : Borne  
!set   bC.x = 10  
!set   bC.y = 13  
  
!create bD : Borne  
!set   bD.x = 9  
!set   bD.y = 10
```

```
!create bE : Borne
!set    bE.x = 14
!set    bE.y = 9
```

```
!create bF : Borne
!set    bF.x = 12
!set    bF.y = 6
```

```
!create bG : Borne
!set    bG.x = 7
!set    bG.y = 6
```

```
!create bH : Borne
!set    bH.x = 3
!set    bH.y = 4
```

```
!create bI : Borne
!set    bI.x = 0
!set    bI.y = 0
```

```
!create bJ : Borne
!set    bJ.x = 0
!set    bJ.y = 14
```

```
!create bK : Borne
!set    bK.x = 16
!set    bK.y = 14
```

```
!create bL : Borne
!set    bL.x = 16
!set    bL.y = 0
```

```
!create bM : Borne
!set    bM.x = 10
!set    bM.y = 0
```

```
-----
-- Sommet --
-----
```

```
!create sbAp1 : Sommet
!set    sbAp1.numero = 1

!create sbHp1 : Sommet
!set    sbHp1.numero = 2

!create sbGp1 : Sommet
!set    sbGp1.numero = 3

!create sbBp1 : Sommet
!set    sbBp1.numero = 4

!create sbBp2 : Sommet
!set    sbBp2.numero = 1

!create sbGp2 : Sommet
!set    sbGp2.numero = 2

!create sbFp2 : Sommet
!set    sbFp2.numero = 3

!create sbEp2 : Sommet
!set    sbEp2.numero = 4

!create sbDp2 : Sommet
!set    sbDp2.numero = 5

!create sbCp2 : Sommet
!set    sbCp2.numero = 6

!create sbAp3 : Sommet
!set    sbAp3.numero = 1

!create sbIp3 : Sommet
!set    sbIp3.numero = 2

!create sbMp3 : Sommet
!set    sbMp3.numero = 3
```

```
!create sbLp3 : Sommet
!set    sbLp3.numero = 4
```

```
!create sbFp3 : Sommet
!set    sbFp3.numero = 5
```

```
!create sbGp3 : Sommet
!set    sbGp3.numero = 6
```

```
!create sbHp3 : Sommet
!set    sbHp3.numero = 7
```

```
!create sbCp4 : Sommet
!set    sbCp4.numero = 1
```

```
!create sbDp4 : Sommet
!set    sbDp4.numero = 2
```

```
!create sbEp4 : Sommet
!set    sbEp4.numero = 3
```

```
!create sbFp4 : Sommet
!set    sbFp4.numero = 4
```

```
!create sbLp4 : Sommet
!set    sbLp4.numero = 5
```

```
!create sbKp4 : Sommet
!set    sbKp4.numero = 6
```

```
!create sbAp5 : Sommet
!set    sbAp5.numero = 1
```

```
!create sbBp5 : Sommet
!set    sbBp5.numero = 2
```

```
!create sbCp5 : Sommet
!set    sbCp5.numero = 3
```

```
!create sbKp5 : Sommet
```

```

!set      sbKp5.numero = 4

!create sbJp5 : Sommet
!set      sbJp5.numero = 5

!create sbIp5 : Sommet
!set      sbIp5.numero = 6

-----
-- Gerant --
-----
!insert ( pe1, ex1 ) into Gerant
!insert ( pe2, ex2 ) into Gerant

-----
-- Proprietaire --
-----
!insert ( pe1, ex1 ) into Proprietaire
!insert ( pe2, ex2 ) into Proprietaire

-----
-- Composition --
-----
!insert ( ex1, p1 ) into Composition
!insert ( ex1, p2 ) into Composition
!insert ( ex1, p3 ) into Composition
!insert ( ex2, p4 ) into Composition
!insert ( ex2, p5 ) into Composition

-----
-- ParcelleSommet --
--      &      --
-- SommetBorne --
-----

-- Parcelle p1
!insert ( p1, sbAp1) into ParcelleSommet
!insert ( sbAp1, bA) into SommetBorne

```

```
!insert ( p1, sbHp1) into ParcelleSommet
!insert ( sbHp1, bH) into SommetBorne

!insert ( p1, sbGp1) into ParcelleSommet
!insert ( sbGp1, bG) into SommetBorne

!insert ( p1, sbBp1) into ParcelleSommet
!insert ( sbBp1, bB) into SommetBorne

-- Parcelle p2
!insert ( p2, sbEp2) into ParcelleSommet
!insert ( sbEp2, bE) into SommetBorne

!insert ( p2, sbGp2) into ParcelleSommet
!insert ( sbGp2, bG) into SommetBorne

!insert ( p2, sbFp2) into ParcelleSommet
!insert ( sbFp2, bF) into SommetBorne

!insert ( p2, sbEp2) into ParcelleSommet
!insert ( sbEp2, bE) into SommetBorne

!insert ( p2, sbDp2) into ParcelleSommet
!insert ( sbDp2, bD) into SommetBorne

!insert ( p2, sbCp2) into ParcelleSommet
!insert ( sbCp2, bC) into SommetBorne

-- Parcelle p3
!insert ( p3, sbAp3) into ParcelleSommet
!insert ( sbAp3, bA) into SommetBorne

!insert ( p3, sbIp3) into ParcelleSommet
!insert ( sbIp3, bI) into SommetBorne

!insert ( p3, sbMp3) into ParcelleSommet
!insert ( sbMp3, bM) into SommetBorne

!insert ( p3, sbLp3) into ParcelleSommet
!insert ( sbLp3, bL) into SommetBorne
```

```
!insert ( p3, sbFp3) into ParcelleSommet
!insert ( sbFp3, bF) into SommetBorne

!insert ( p3, sbGp3) into ParcelleSommet
!insert ( sbGp3, bG) into SommetBorne

!insert ( p3, sbHp3) into ParcelleSommet
!insert ( sbHp3, bH) into SommetBorne

-- Parcelle p4
!insert ( p4, sbCp4) into ParcelleSommet
!insert ( sbCp4, bC) into SommetBorne

!insert ( p4, sbDp4) into ParcelleSommet
!insert ( sbDp4, bD) into SommetBorne

!insert ( p4, sbEp4) into ParcelleSommet
!insert ( sbEp4, bE) into SommetBorne

!insert ( p4, sbFp4) into ParcelleSommet
!insert ( sbFp4, bF) into SommetBorne

!insert ( p4, sbLp4) into ParcelleSommet
!insert ( sbLp4, bL) into SommetBorne

!insert ( p4, sbKp4) into ParcelleSommet
!insert ( sbKp4, bK) into SommetBorne

-- Parcelle p5
!insert ( p5, sbAp5) into ParcelleSommet
!insert ( sbAp5, bA) into SommetBorne

!insert ( p5, sbBp5) into ParcelleSommet
!insert ( sbBp5, bB) into SommetBorne

!insert ( p5, sbCp5) into ParcelleSommet
!insert ( sbCp5, bC) into SommetBorne

!insert ( p5, sbKp5) into ParcelleSommet
!insert ( sbKp5, bK) into SommetBorne

!insert ( p5, sbJp5) into ParcelleSommet
```

```
!insert ( sbJp5, bJ) into SommetBorne
!insert ( p5, sbIp5) into ParcelleSommet
!insert ( sbIp5, bI) into SommetBorne
```

### **27) Quelles sont les parcelles de la propriété CH. Margaux ?**

```
ex1.les_parcelles
Set{@p1,@p2,@p3}: Set(Parcelle)
```

### **28) Donner les noms de personnes qui ne sont responsables d'aucune propriété.**

```
Personnes.allInstances->select(p|p.est_gerant_de->size()==0).nom
Bag{'ducourt'} : Bag(String)
```

### **29) Combien de bornes possède la parcelle P2 ?**

```
p2.les_sommets.la_borne->size()
6 : Integer
```

### **30) Pour chaque parcelle, donner son nombre de bornes ?**

```
Parcelle.allInstances->Collect(x|x.les_sommets.la_borne->size())
Set{4, 6, 7, 6, 6} : Set(Integer)
```

### **31) Quelle est la personne qui a le plus de propriété ?**

```
Personne.allInstances->select(p1|p1.est_proprio_de->size()==
  Personne.allInstances->Collect(p2|p2.est_proprio_de->size())->
  iterate(n ;max :Integer=0 | if n>max then n else max endif)).nom
Bag{'dupont'}: Bag(String)
```