Nom :	Prénom :	page 1

Module Bases de Données et Web
Examen réparti du 5 novembre 2010

Les documents sont autorisés – Durée : 2h.

Répondre aux questions sur la feuille du sujet dans les cadres appropriés. La taille des cadres suggère celle de la réponse attendue. Utiliser le dos de la feuille précédente si la réponse déborde du cadre. Le barème est donné à titre indicatif. La qualité de la rédaction sera prise en compte. Ecrire à l'encre bleue ou noire. Ne pas dégrafer le sujet.

Exercice 1. SQL3 — 12 pts

On considère le schéma SQL3 suivant :

```
Create type T_Pays ;
Create type T_frontiere as object (
     Pays ref T_Pays,
     Lg number
) ;
Create type Ensfrontieres as table of T_frontiere ;

Create type T_Ville as object (
     Nom varchar2(30),
     Pays ref T_Pays,
     Population number
);
Create type EnsVilles as table of ref T_Ville;
Create type T_Pays as object (
     Nom varchar2(30),
     Capitale ref T_Ville,
     villesPrincipales EnsVilles,
     population number,
     limites Ensfrontieres
) ;
Create type EnsPays as table of ref T_Pays;


Create table LesVilles of T_Ville;

Insert into LesVilles values('Paris', NULL, 2000000);
Insert into LesVilles values ('Lyon', NULL, 500000);
Insert into LesVilles values ('Bruxelles', NULL, 1000000);
```

Question 1. Définir un type T_Federation qui a un nom et un ensemble de pays membres.

Question 2. Définir les tables LesPays et LesFederations permettant de stocker les instances de T_Pays et T_Federation respectivement.

Lettres initiales du Prénom et du Nom:	page 2

Question 3. Effectuez les insertions suivantes :

Le pays France, dont la capitale est Paris, qui a Lyon comme ville principale et qui comporte 62 000 000 habitants ; (pour le moment, le champ limites est vide).

La ville de Marseille, en France, qui comporte 900 000 habitants ;

Insérez la ville de Marseille dans les villes principales de France ;

Insérez le pays Belgique, qui a Bruxelles comme capitale, qui a 10 000 000 habitants, et qui a une frontière avec la France de 620 km de long.

Insérez la fédération nommée FB, qui comporte la France et la Belgique ;

Question 4. Ecrivez en SQL3 les requêtes suivantes :

1. Pour chaque pays, calculez la longueur de ses frontières.

```
select

from

...
```

2. Nom des pays frontaliers de la Belgique, et le nom de leur capitale.

```
select

from

where
```

3. Nom des villes principales des pays membres de la fédération FB.

```
select

from

where
```

4. Nom des pays qui ont plus de 3 villes de plus de 100 000 habitants (on considère que la capitale est comprise dans la liste des villes principales)

```
select

from

where
```


Exercice 2 : OQL — 8 pts

Soit le schéma ODL d'une base pour gérer les utilisateurs d'un réseau social. Un internaute possède un mur sur lequel il partage ses photos. Un internaute peut commenter les photos des autres internautes. Un internaute peut voter pour (avis positif) ou contre (avis négatif) une photo. Une photo peut être associée avec les personnes qui apparaissent en portrait sur la photo. Un internaute a des amis directs, toujours mutuels. Les amis indirects, éloignés d'un chemin de longueur d, sont appelés les proches, *cf.* la méthode proches(d).

```
interface Personne { keys mail;
attribute string mail;
attribute string nom;
attribute string prénom;
relationship set<Photo> paraît_sur inverse Photo::portrait_de
};
```

```
interface Mur { extent Murs; keys id;
attribute string id;
attribute string nom;
relationship Internaute propriétaire inverse Internaute::mur;
relationship set<Photo> contient inverse Photo::sur;
Photo meilleur_avis();
set<Photo> x() ;
};
```

```
interface Internaute : Personne { extent Internautes;
attribute string login;
attribute long age;
attribute string ville;
relationship set<Internaute> amis inverse Internaute::amis;
relationship Mur mur inverse Mur::propriétaire;
set<Internaute> proches(long d);
};
```

```
interface Photo { extent Photos; keys numéro;
attribute long numéro ;
attribute string nom;
attribute long hauteur;
attribute long largeur;
attribute long avis_positifs ;
attribute long avis_négatifs ;
relationship Mur sur inverse Mur::contient;
relationship set<Personne> portrait_de inverse Personne::paraît_sur;
relationship set<Commentaire> commentaires inverse Commentaire::sujet;
};
```

```
interface Commentaire { keys numéro;
attribute long numéro;
attribute string nom;
attribute string texte;
relationship Internaute écrit_par;
relationship Photo sujet
        inverse Photo::commentaires;
} ;
```

Question 1. Ecrire en OQL les requêtes suivantes :

R1 : Quel est le nom des photos commentées par l'internaute dont le login est 'jb007' et qui habite à Paris?

R2 : Quels sont les internautes qui ont commenté au moins une photo d'un de leurs amis ? Afficher le login des internautes en question. Donner deux réponses équivalentes, la première en utilisant la racine *Internautes*, la deuxième utilisant la racine *Photos*.

```
select
from i in Internautes
where
```

```
select
from p in Photos
where
```

R3 : Pour chaque internaute, combien a-t-il d'amis? Afficher le login de l'internaute et son nombre d'amis.

R4 : Quel est le format (hauteur et largeur) de toutes les photos partagées sur les murs des amis de Robin Débois ?

Question 2

Ecrire le corps de la méthode *meilleur_avis()* qui retourne la photo ayant le plus grand score, parmi les photos dudit mur. Le score d'une photo est calculé avec la formule : *score = avis_positifs – avis_négatifs*.

```
Photo Mur::meilleur_avis() {




}
```

Question 3

a) Expliquer ce que retourne la méthode x() définie ci-dessous. Quel est l'inconvénient majeur de cette méthode ?

```
set<Photo> Mur::x() {
  return (this.contient
           union
          select distinct p
          from a in this.propriétaire.amis, p in a.mur.x());
}
```


b) Ecrire le corps de la méthode *proches(d)* qui retourne l'ensemble des amis (et amis d'amis par transitivité) atteignables par un chemin de longueur inférieure ou égale à d. Remarque : si nécessaire vous pouvez utiliser une instruction conditionnelle *if then else endif*.

```
set<Internaute> Internaute::proches(long d) {




}
```

c) Etant donné une personne *A*, on veut déterminer l'ensemble *E* des personnes qui apparaissent soit sur la même photo que *A*, soit sur la même photo que quelqu'un appartenant à *E*. Expliquer brièvement comment obtenir *E*.