Examen MIF18 - Gestion de données pour le Web - session 1 - 15 décembre 2014
Durée : 1h30
Documents autorisés

Numéro de copie :

Il faut rendre ces feuilles en les glissant dans votre copie anonyme. Ne pas l'utiliser comme brouillon. Les réponses sont à donner sur ces feuilles, pas dans la copie. Remplir le champ ci-dessus avec le *numéro de la copie* dans laquelle vous allez la glisser. Remplir la partie d'anonymat de la copie, puis coller le coin.

Exercice 1: Relationnel et DTD (4 points)

On considère le schéma relationnel suivant :

```
CREATE TABLE hotel(idh INTEGER, nom VARCHAR(50),
                   idh PRIMARY KEY);
CREATE TABLE chambre(idh INTEGER, idc INTEGER, type VARCHAR(10),
                     (idh,idc) PRIMARY KEY,
                     (idh) REFERENCES hotel(idh));
CREATE TABLE client(c INTEGER, nom VARCHAR(50),
                    c PRIMARY KEY);
CREATE TABLE reservation(idh INTEGER, idc INTEGER, c INTEGER, jour DATE,
                     (idh,idc,c,jour) PRIMARY KEY,
                     (idh,idc) REFERENCES chambre(idh,idc),
                     (c) REFERENCES client(c));
```

1. Écrire une DTD pour des documents XML qui doivent contenir des données issues, ou à destination, d'une base de données ayant le schéma relationnel ci-dessus. On construira cette DTD de façon à garantir le respect d'un maximum de contraintes de clé et de clé étrangère du schéma relationnel. On réutilisera quand c'est possible les noms issus du schéma relationnel pour les noms d'éléments/d'attributs correspondant dans la DTD.

2. Indiquer la/les contrainte(s) (clé/clé étrangère) du modèle relationnel qui ne sont pas forcément respectées dans un document respectant la DTD de la question précédente.

Exercice 2: Inférence de DTD (2 points)

On considère le document xml suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<pieces>
<piece numero="5">
  <nom>Cardan</nom>
  <description>
    Cette piece transmet la motricite aux
    <pc href="roues.xml/3">roues</pc>
  </description>
</piece>
<piece numero="3">
  <nom>Boite de vitesse</nom>
  <description>Contient les <pc ref="4">engrenages</pc> pour les
     differents rapports de vitesse</description>
  <piece numero="4">
    <nom>Engrenage 32 dents</nom>
  </piece>
</piece>
</pieces>
```

Cocher parmi les déclarations de DTD suivantes celles qui correspondent au document ci-dessus (c'est-à-dire pour lesquelles le document ci-dessus est valide) :

- ☐ `<!ELEMENT pieces (piece+)>`
- ☐ `<!ELEMENT pieces (piece*)>`
- ☐ `<!ELEMENT piece (nom,description)>`
- ☐ `<!ELEMENT piece (numero)>`
- ☐ `<!ELEMENT piece (nom,description,piece+)>`
- ☐ `<!ELEMENT nom (#PCDATA)>`
- ☐ `<!ELEMENT nom ((#PCDATA|pc)*)>`
- ☐ `<!ELEMENT description ((#PCDATA|pc)*)>`
- ☐ `<!ATTLIST piece nom CDATA #REQUIRED>`
- ☐ `<!ATTLIST piece numero IDREF #REQUIRED>`
- ☐ `<!ATTLIST pc href CDATA #IMPLIED ref IDREF #IMPLIED>`

Remarque : l'ensemble déclarations qui correspondent à des cases cochées n'a pas à former une DTD cohérente, on considère chaque déclaration indépendamment des autres.

Exercice 3: Correspondance Objet-Relationnel (3 points)

On considère deux entités A et B représentées en modèle relationnel par des tables T_A et T_B respectivement. On fait correspondre via un ORM ces tables à des classes C_A et C_B. On suppose que A et B sont reliées par une association $1 - n$ (à un A peut correspondre n B et à chaque B correspond un seul A).

1. Dans le modèle relationnel, cette association **peut** être représentée par (cocher les cases correspondant aux possibilités valides) :

   - ☐ Un attribut dans la table T_A et une clé étrangère de cet attribut vers la table T_B
   - ☐ Un attribut dans la table T_B et une clé étrangère de cet attribut vers la table T_A
   - ☐ Une nouvelle table T_{assoc} avec une clé étrangère vers la table T_A et une clé étrangère vers la table T_B

2. Dans le modèle objet, cette association **peut** être représentée par (cocher les cases correspondant aux possibilités valides, on supposera que l'utilisation d'une collection n'a de sens que pour stocker plusieurs éléments) :

   - ☐ Un champ dans la classe C_A de type C_B
   - ☐ Un champ dans la classe C_B de type C_A
   - ☐ Un champ dans la classe C_A de type collection de C_B
   - ☐ Un champ dans la classe C_B de type collection de C_A
   - ☐ Un champ dans la classe C_A de type C_B et un champ dans la classe C_B de type collection de C_A
   - ☐ Un champ dans la classe C_B de type C_A et un champ dans la classe C_A de type collection de C_B

3. Justifiez brièvement l'obligation pour une entité (i.e. une classe correspondant à une table via un ORM) de posséder un identifiant.

Exercice 4: Calculs de statistiques avec MapReduce sur MongoDB (7 points)

On considère la collection `zips` utilisée en TP. Les enregistrements ont la structure suivante, où la clef de document `_id` est un code postal des États-Unis.

```
> db.zips.findOne();
{
  "_id" : "01001",
  "city" : "AGAWAM",
  "loc" : [
    -72.622739,
    42.070206
  ],
  "pop" : 15338,
  "state" : "MA"
}
```

On donne l'exemple suivant de fonction `map` et `reduce` en MongoDB

```
map_ex = function () {
  emit(0, 1);
}

red_ex = function (key, values) {
  var sum = 0;
  for(var i=0; i<values.length; i++)
    sum += values[i];

  return sum;
}
```

On souhaite calculer quatre statistiques sur les populations des états des États-Unis : le nombre d'états, la population de l'état le plus peuplé, la moyenne des populations ainsi que l'écart type sur les populations. On rappelle la définition de la moyenne notée $\bar{x}$ et les deux définitions équivalentes de l'écart-type noté σ :

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \text{ et } \sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}} = \sqrt{\frac{(\sum_{i=1}^{n} x_i^2) - n\bar{x}^2}{n-1}}$$

1. Expliquer ce que calcule le job Map/Reduce donné en exemple.

2. Donner la fonction `map` et la fonction `reduce` permettant de calculer la population totale de chaque état. On suppose que l'on exécute la commande suivante `db.zips.mapReduce(map_pop_state,red_pop_state, out : "pop_state");` qui permet d'enregistrer le résultat dans la collection nommée `pop_state`.

3. À partir du résultat de la question précédente, on souhaite calculer les quatre statistiques *en utilisant un seul job Map/Reduce*. Entre les deux définitions équivalentes de l'écart-type, justifier le choix de celle qu'il faut utiliser.

4. On donne la fonction `map_stats` définie ci-après. Expliquer le choix de la structure contenant quatre champs pour la valeur émise.

```
map_stats = function () {
  emit(0, {nb :1, s : this.value, m : this.value, sq : this.value*
     this.value});
}
```

5. En utilisant la fonction `map_stats`, donner la fonction `reduce` et la fonction `finalize` [1] permettant de calculer les quatre statistiques en une seule exécution. La fonction javascript pour calculer la racine carrée est `Math.sqrt()`.

1. C'est la fonction qui est exécutée une fois tous les reducers terminés.

Exercice 5: Saturation RDF Schema avec MapReduce sur MongoDB (7 points)

On considère un graphe RDF représenté par une collection `graphe` dans une base MongoDB. Chaque document de la collection correspond à un triplet. On suppose que pour un triplet t, son sujet (respectivement son prédicat et son objet) est accessible via $t.sujet$ (respectivement $t.predicat$ et $t.objet$).

On considère la règle d'inférence RDF Schema suivante :

$$P \texttt{ rdfs:domain } Y \wedge S\ P\ O \Rightarrow S \texttt{ rdf:type } Y$$

1. On considère le graphe RDF suivant comme exemple :

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX ex:<http://www.example.com#>

ex:Alice ex:parent ex:Bob.
ex:parent rdfs:domain ex:personne.
rdfs:domain rdfs:domain rdf:Property.
```

   Donner le résultat de l'application de la règle d'inférence ci-dessus sur ce graphe exemple.

2. Donner la fonction `map`, la fonction `reduce` et la fonction `finalize` permettant d'inférer les triplets qui peuvent être déduits à partir du graphe courant en utilisant cette règle.

   Remarque : Il est possible d'utiliser des URI comme clés.

fonction `map` :

fonction `reduce` :

fonction `finalize` :

3. On considère maintenant la règle d'inférence suivante

$$C \texttt{ rdfs:subClassOf } D \wedge D \texttt{ rdfs:subClassOf } E \Rightarrow C \texttt{ rdfs:subClassOf } E$$

En supposant que l'on utilise un job MapReduce similaire au précédent pour calculer les triplets qui peuvent être déduits à partir du graphe courant en utilisant cette règle, aura-t-on fait toutes les déductions possibles (concernant cette règle) ? Justifier.