Final NF18 P2024

- Tous les documents papier sont autorisés.
- Aucun appareil connecté n'est autorisé (ordinateur, téléphone, montre connectée, etc.).
- Les réponses peuvent être données en anglais pour les étudiants étrangers (le préciser sur la copie).
- Le nom doit être écrit lisiblement sur chaque copie.
- La durée totale de l'examen est de 1h30 (2h pour les tiers-temps).
- Vous devez répondre aux **exercices 1 et 2 sur une copie**, et aux **exercices 3, 4 et 5 sur une autre copie** !

Barème :

- Ex 1 - [35 min]
  Q1 : 3 pts, Q2 : 3pts
- Ex 2 - [30min]
  Q1 : 2pts, Q2 : 1 pt, Q3 : 1 pts
- Ex 3 - [30 mn]
  Q1 : 3pt, Q2 : 1pt, Q3 : 1pt
- Ex 4 - [20 mn]
  Q1 : 1pt, Q2 : 2 pt
- Ex 5 - [10 mn]
  Q1 : 1pt, Q2 : 1 pt

1. Conception de bases de données

Transport maritime

L'agence gouvernementale pour le transport maritime souhaite faire évoluer son système informatique avec la création d'une nouvelle base de données, afin d'organiser son activité de suivi des transports de marchandise et de personnes sur les ports en France, effectuée par des transporteurs maritimes. Un transporteur maritime est soit un armateur (propriétaires des navires) soit un affréteur (locateurs de navires). Ils sont chacun identifié par un code unique, ont un nom, et sont installés dans un pays.

Un armateur possède une part de propriété dans un ou plusieurs navires (supérieur à 0% et inférieure ou égale à 100%). Ainsi un navire est la propriété d'un ou de plusieurs armateurs, selon une part de propriété qu'on voudra renseigner. Un affréteur loue des navires. On souhaite maintenir à jour un registre de location, avec les différents navires loués et ses affréteurs respectifs. On enregistrera la date de début et de fin de location. Un navire pourra être loué plusieurs fois à un même affréteur à des dates différentes.

Les navires sont identifiés par un code unique et ont un volume utile disponible. Il y a plusieurs types de navires parmi lesquels on veut distinguer les suivants : pétroliers, porte-conteneurs, gaziers, navires de croisière. Les porte-conteneurs ont un nombre maximal de conteneurs. Les navires de croisière sont utilisés pour le transport de personnes, et on notera le nombre maximum de passagers. Pour les autres types de navires (y compris gaziers et pétroliers), on notera seulement le code et le volume utile.

Les armateurs et les affréteurs inscrits sur la base gèrent chacun des voyage, mais au moins un. Un voyage est identifié par un code unique et utilise un navire. Pour chaque voyage on notera le port de départ et le port d'arrivée. Les ports sont identifiés par un code et leur ville.

Les villes sont identifiées par leur nom et leur pays. Les pays sont identifiés par un code unique international et ont un nom.

Question 1

Proposez un MCD en UML pour modéliser la base de données selon les informations fournies. Vous utiliserez, au moins, deux association d'héritage, une composition, une classe d'association. Pensez à bien représenter toutes les contraintes existantes. Posez des hypothèses si des informations semblent vous manquer dans le sujet.

Question 2

Proposez une transformation de votre MCD en MLD relationnel, sans perte d'informations. Justifiez les transformations d'héritage que vous avez utilisez.

2. Requêtes SQL

Transport maritime II

Pour élargir la base de données des Transporteurs maritimes, on rajoute les employés qui compose des équipages des navires pour le transport international. Chaque employé est identifié par son matricule et on note leur nom, prénom, leur pays d'origine et la société de transport maritime qui l'emploie. Les employés peuvent être affectés à des voyages en composant ainsi l'équipage d'un navire. Pour chaque voyage l'employé aura un rôle précis, qui peut être Capitaine de vaisseau (CV) ou Mécanicien (M) ou Assistant-matelot (AM). Un MLD pour cette partie a été proposé comme suit (attention, certaines relations de ce MLD peuvent différer du MLD demandé à l'exercice précédent) :

```
1 Pays(#nom : string)
2 Employe(#mat : integer, nom: string, prenom : string, employeur : string,
  pays_origine => Pays)
3 Voyage(#code: integer, pays_dep => Pays, pays_dest => Pays, longueur_trajet :
  float) avec {longueur_trajet > 0}
4 Fonction(#emp => Employe, #voyage => Voyage, rôle: {'CV','M','AM'})
```

Question 1

Ecrire en AR et en SQL la requête qui donne la matricule des employés qui n'ont jamais fait des voyages à destination de la Grande Bretagne (sans utiliser de sous-requêtes ni des opérateurs ensemblistes).

Question 2

Ecrire une requête SQL qui affiche pour chaque employé (s'il a réalisé au moins un voyage en tant que capitaine de vaisseau) son matricule, son nom, le nombre de voyages qu'il a réalisé en tant que capitaine de vaisseau, et la longueur moyenne de ces voyages.

Question 3

Ecrire une requête SQL qui donne pour chaque voyage le nombre d'employés de chaque employeur sur le voyage, à condition qu'il y ait au moins 2 employés de cet employeur dans le voyage.

Changez de copie !

3. Normalisation fonctionnelle de rendez-vous médicaux

Un hôpital veut mettre en place une base de données pour gérer les rendez-vous de ses médecins et de ses patients. Il a identifié les attributs suivants : num_medecin (le numéro de diplôme d'un médecin), nom_medecin (le nom d'un medecin), bureau (le numéro de bureau d'un médecin), service (le service où travaille un médecin), ss_patient (le numéro de sécurité sociale d'un patient), nom_patient (le nom d'un patient), prénom_patient (le prénom d'un patient), mail_patient (l'adresse mail d'un patient), maladie (la maladie pour laquelle le patient est traité par le médecin), traitement (le type de traitement que suit le patient), type_maladie (le type de la maladie) et date_rdv (la date et l'heure du rendez-vous).

Les dépendances fonctionnelles suivantes ont été identifiées :

- num_medecin → nom_medecin, bureau, service
- bureau → service
- ss_patient → nom_patient, prenom_patient, mail_patient
- mail_patient → ss_patient, nom_patient
- num_medecin, ss_patient → maladie, traitement, mail_patient
- maladie → traitement, type_maladie

Question 1

Donnez la fermeture transitive, une couverture minimale et la ou les clés de la relation contenant l'ensemble des attributs identifiés. Justifiez vos réponses.

Question 2

Quelle est la forme normale de cette relation ? Justifiez.

Question 3

Proposez un découpage en 3NF sans perte d'informations.

4. Base de données NoSQL de parcs d'attraction

Le groupe de restauration Tacos King veut mettre en place une base de données NoSQL en plsql/JSON de ses parcs d'attraction. Un parc d'attraction a un nom unique, une adresse (composée d'un numéro, d'une rue et d'une ville), d'un thème, et d'une liste d'attractions, chacune ayant un nom, un nombre de places maximum (un entier supérieur à 0), un âge minimal supérieur à 0 facultatif, et une note entre 0 et 10. On vous donne le MLD suivant (tous les attributs sont NOT NULL) :

parcs (#nom : str, adresse : JSON, thème : enum{western, sci-fi, fantasy}, attraction : JSON)

Question 1

Donnez les instructions SQL pour créer la table et insérer le restaurant 'wonderland UTC', situé au numéro 42 rue Roger Coutolenc à Compiègne, dont le thème est fantasy, et ayant deux attractions :

- l'attraction 'grimoire de bases de données', qui a un nombre de place de 100, pas d'âge minimal, et une note de 2.
- l'attraction 'donjons et programmation orientée objet' qui a un nombre de place de 120, un âge minimal de 20, et une note de 7.

Question 2

Répondez à chaque interrogation suivante par une requête SQL (on considérera que tous les champs JSON sont bien remplis sans erreurs) :

- Donnez le nombre de parcs d'attraction de la ville de Compiègne.
- Donnez pour chaque parc d'attraction le nombre de places total de ses attractions qui ont une note supérieure à 6.

Indice :

La fonction SQL CAST(expression AS type) permet de convertir une expression en un type particulier, par exemple une expression sous forme chaîne de caractère en INT.

5. Gestion de biomes

Kif Kroker est chargé de gérer les biomes où sont recueillies des espèces en voie d'extinction sur la station spatiale 'Last Resort'. Il stocke ses données en utilisant le modèle relationnel suivant (tous les attributs sont NOT NULL) :

Biome (#num : int, type : enum{eau, désert, jungle})

Espèces(#nom : str, nb_pattes :int, nb_yeux : int, tres_dangereux : boolean) avec {nb_pattes >= 0 ; nb_yeux > = 0}

Occupe(#biome=>Biomes(num), #espece=>Especes(num), nombre :int) avec {nombre > 0}

Question 1

Kif met à jour sa base de données en implémentant le déplacement de deux très dangereux lapins de Caerbannog par les intruction suivantes :

UPDATE TABLE Occupe SET nombre = nombre -2 WHERE biome = 2 AND especе = 'lapin')

UPDATE TABLE Occupe SET nombre = nombre +2 WHERE biome = 4 AND especе = 'lapin')

Malheureusement, une coupure de courant frappe la station pendant son travail, et quand le courant est rétabli, la requête suivante donne le résultat que deux lapins manquent dans les biomes de la station spatiale :

SELECT SUM(nombre)

FROM Occupe

WHERE especе = 'lapin' ;

Expliquez à Kif ce qui s'est passé avant qu'il ne sonne l'alerte en croyant deux dangereux animaux en liberté dans la station. Qu'aurait-il du faire pour éviter le problème ?

Question 2

Kif utilise très souvent la requête suivante, où [type_entré] est un type de biome donné en entrée :

SELECT occupe.biome, occupe.especе, occupe.num

FROM biome JOIN occupe ON biome.num = occupe.biome

WHERE biome.type = [type_entré] ;

Quelle modification de la base de données lui proposeriez-vous pour optimiser cette requête ? Réécrivez la requête suite à cette modification.