Donnez les réponses sur le sujet que vous joindrez à votre copie.

1 Réseau social : contraintes et requêtes

Dans tous les exercices suivants, on fixe une base de données utilisée par un réseau social. La base de données respecte le schéma ci-dessous, où les clefs primaires sont soulignées et les clefs étrangères listées à la suite.

```
user(id, name, surname, birth_year, cityU)
place(id, name, category, cityP)
follow(idU1,idU2, date)
like(idU, idP, date)
```

Clefs étrangères : `idU1`, `idU2` dans `follow` et `idU` dans `like` font référence à `id` dans `user` ; `idP` dans `like` fait référence à `id` dans `place`.

Les données concernant les utilisateurs du réseau social sont stockées dans la table `user`. Certains lieux sont par ailleurs référencés dans la table `place`. Ils peuvent avoir diverses catégories telles que `restaurant`, `concert hall`, `shop`, etc. Les utilisateurs peuvent se suivre mutuellement (table `follow`, où `idU1` suit `idU2`) ou aimer des lieux (table `like`, où l'utilisateur `idU1` aime le lieu `idP`).

Exercice 1 : Lecture du schéma relationnel

Pour chacune des questions suivantes, écrivez `autorisé` si le schéma permet les situations décrites et `interdit` sinon.

1) Deux restaurants (`category='restaurant'`) situés dans la même ville peuvent avoir le même nom.

2) On peut enregistrer le fait qu'une personne aime un lieu à deux dates différentes.

3) Une personne peut suivre une autre personne qui ne la suit pas.

Exercice 2 : création des tables

Pour chacune des questions suivantes, dites quels choix faire (contraintes, types de données, etc.) à la création des tables pour obtenir les comportements voulus.

La syntaxe SQL précise n'est pas obligatoire : indiquez seulement la nature du choix, et les tables et attributs concernés.

1) Dans une même ville deux lieux différents ne peuvent avoir le même nom.

2) La catégorie d'un lieu est forcément renseignée.

3) Si un lieu disparait de la base de données, cette modification est automatiquement répercutée dans l'ensemble de la base de données.

4) Il n'y a aucun utilisateur né avant le vingtième siècle.

Exercice 3 : requêtes SQL

Proposez des requêtes SQL permettant d'extraire les informations demandées.

1) Les noms des lieux dont la catégorie n'est pas renseignée.

2) La liste des utilisateurs qui aiment un restaurant situé hors de leur ville (tableau résultat : `name, surname`).

3) La liste des utilisateurs qui n'ont aucun follower.

4) Le ou les utilisateurs qui ont le plus de followers.

5) Pour chaque utilisateur comptant au moins 10 followers, son nombre de followers (tableau résultat : (`idU`, `nb`)). Les résultats doivent être triés dans l'ordre décroissant du nombre total de followers.

Exercice 4 : vues et requêtes récursives avec postgres

1) Afin d'améliorer l'efficacité de la détection des faux profils dans notre base de données, la vue suivante est créée (avec `requête 3.3` la requête demandée à la question 3 de l'exercice 3) :

```
CREATE VIEW bot(id) AS
(requête 3.3);
```

Un utilisateur classifié par le système comme fantôme voit son compte bloqué. Il fait donc un recours pour que son compte soit rétabli : le pauvre n'a juste pas d'amis... Un stagiaire essaie alors de modifier la vue `bot` afin de réintégrer l'utilisateur (dont l'identifiant est 42). Il utilise la requête suivante :

```
DELETE FROM bot WHERE id=42 ;
```

Que se passe-t-il ? Expliquez bien les raisons du comportement observé. Expliquez également au stagiaire comment améliorer sa solution.

2) Proposez une requête SQL retournant les utilisateurs u_n connectés à l'utilisateur 66 par une chaîne de followers $u_1, u_2, ..., u_{n-1}$, i.e., les utilisateurs u_n tels que 66 suit u_1, u_1 suit u_2, ..., u_{n-1} suit u_n. Vous retournerez seulement les identifiants d'utilisateur.

Exercice 5 : requêtes en algèbre relationnelle

Proposez des requêtes d'algèbre relationnelle permettant d'extraire les informations demandées.

1) Les utilisateurs qui n'ont aucun follower (tableau résultat : `id`).

2) Les noms et prénoms des personnes qui aiment le lieu `'Ravenink'` dans la catégorie `'tatoo'`.

2 Modélisation : reverse engineering, enrichissement et amélioration du schéma

1. Proposez une modélisation E/R correspondant au schéma relationnel décrit précédemment. On précisera les entités, les identifiants, les associations, on indiquera pour chaque association les cardinalités. Listez les contraintes externes avec soin et expliquez comment les implémenter lorsque ceci est possible avec les notions vues en cours.
2. On décide d'étendre les possibilités offertes par le réseau social. Désormais les utilisateurs pourront publier des posts. On enregistrera l'auteur, le contenu (au format `text`), ainsi que la date des posts. Tout post pourra être aimé par un utilisateur, à la condition qu'il n'en soit pas l'auteur. Par ailleurs, on décide de raffiner la catégorisation des lieux en autorisant un même lieu à appartenir à plusieurs catégories (ainsi, 'RavenInk' appartiendra non seulement à la catégorie `tatoo`, mais aussi à la catégorie `shop`). Un même lieu ne pourra cependant pas appartenir à plus de quatre catégories différentes. Chaque catégorie aura non seulement un nom, mais également une description. Enfin, les posts pourront être tagués avec le même type de catégories que celles qualifiant les lieux. Modifiez le schéma E/R que vous avez proposé afin d'accommoder ces nouveaux besoins. N'oubliez pas de lister les contraintes externes.
3. Proposez une traduction de votre diagramme dans le modèle relationnel. Précisez les clefs primaires et les contraintes référentielles (en particulier les clefs étrangères). Si certaines contraintes du modèle n'ont pas pu être implementées, précisez-le.