

Licence 2 — Info3Bb

Contrôle continu du Vendredi 20 Décembre 2024

Durée 2h • Document autorisé : 1 feuille A4

Le barème est donné à titre indicatif



Si un exercice vous conduit à faire des hypothèses, indiquez-les clairement sur votre copie.
Justifiez vos réponses.

Exercice 1. Questions de cours — (7pts)

1. Donner les bonnes pratiques d'une modélisation relationnelle OLTP.
2. Discuter des difficultés pour appliquer la garantie de durabilité et des solutions mises en place pour la respecter.
3. Donner les problèmes qui peuvent apparaître lorsque la garantie d'isolation n'est pas respectée. Quels mécanismes permettent d'assurer la garantie d'isolation ?

Exercice 2. Modélisation de schéma — (6pts)

Une entreprise fabriquant des puces électroniques veut suivre sa production, afin d'améliorer sa gestion à l'aide des données concernant l'utilisation de ses puces. Pour cela, elle souhaite enregistrer toutes les puces fabriquées avec leurs caractéristiques (par exemple ses dimensions, les matériaux utilisés, etc.). Elle souhaite également avoir toutes les commandes que des entreprises lui ont passées, avec sa date et la quantité de chaque type de puce. Lorsque les informations sont disponibles, elle souhaite enregistrer l'utilisation que chaque entreprise fait de ses puces, afin de pouvoir surveiller les avancées technologiques dans ce domaine et proposer des nouvelles solutions compétitives.

1. Proposer un modèle entité-association permettant de modéliser l'énoncé.
2. Justifier les choix réalisés.
3. Écrire un script SQL pour réaliser le schéma relationnel correspondant à votre modélisation.

Exercice 3. Interrogation — (7pts)

```
CREATE TABLE bateau (  
    id INTEGER PRIMARY KEY,  
    capacite INTEGER  
);
```

```
CREATE TABLE container (  
    id INTEGER PRIMARY KEY,  
    hauteur NUMERIC,  
    largeur NUMERIC,  
    profondeur NUMERIC  
);
```

```
CREATE TABLE marchandise (  
    id INTEGER PRIMARY KEY,  
    nom TEXT,  
    description TEXT  
);
```

```
CREATE TABLE chargement (  
    id INTEGER PRIMARY KEY,  
    container_id INTEGER REFERENCES  
        container(id),  
    bateau_id INTEGER REFERENCES bateau(id),  
    date_chargement DATE,  
    date_livraison DATE,  
    UNIQUE(container_id, bateau_id,  
        date_chargement)  
);
```

```
CREATE TABLE chargement_marchandise (  
    id INTEGER PRIMARY KEY,  
    chargement_id INTEGER REFERENCES  
        chargement(id),  
    quantite INTEGER  
);
```

Une entreprise d'envoi de containers par bateau dispose d'une base de données ayant le schéma ci-dessus. Un bateau possède une *capacite* représentant le nombre de containers qu'il peut contenir. Un container possède une *hauteur*, une *largeur* et une *profondeur*. La table *chargement* permet de spécifier quel container est stocké sur quel bateau, la date à laquelle il a été chargé (*date_chargement*) et la date à laquelle il a été déchargé du bateau (*date_livraison*). La table *chargement_marchandise* permet de spécifier quelle marchandise est contenue dans un container pour un chargement spécifique, ainsi que sa quantité. À partir des informations données, proposer des requêtes SQL permettant de répondre aux questions suivantes :

1. Indiquer les containers ayant effectué le plus de voyages.
2. Classer les containers du plus volumineux au moins volumineux ($\text{volume} = \text{hauteur} \times \text{largeur} \times \text{profondeur}$), et afficher uniquement ceux ayant une capacité supérieure à 100.
3. Identifier les conflits de chargement, c'est-à-dire les containers qui seraient chargés sur plusieurs bateaux différents à une même date.
4. Donner le top 5 des marchandises livrées en moyenne le plus rapidement (c'est-à-dire ayant la plus petite différence entre la date de chargement et la date de livraison). La requête doit également retourner le temps moyen de livraison pour ces marchandises.