POLYTECH SORBONNE

MAIN4

Bases de Données

Contrôle Final
2017-2018

(Correction)

La durée du contrôle est de 1h40. Cet énoncé contient 5 pages.

Si une question fermée (vrai/faux ou QCM) vous demande de **justifier** ou d'**expliquer** votre réponse, alors répondre sans justification vous rapportera 0 point.

Les questions ne sont pas ordonnées par difficulté croissante.

1 Questions de cours (20 min)

1. Qu'est-ce qu'une clé primaire ?
2. Accès concurrents
   (a) Qu'est-ce que le principe d'exclusion mutuelle ? En quoi peut-il provoquer un interblocage ?
   (b) Lorsque l'on manipule des verrous de transaction ternaires, dans quelle situation y a-t-il un conflit entre des verrous de transactions différentes ?
   (c) Quelle est la condition pour qu'une série de transactions soit "sérialisable" ?
3. Dépendances fonctionnelles :
   (a) Qu'est-ce qu'une fermeture transitive ?
   (b) Qu'est-ce qu'une dépendance élémentaire ?

Correction : *Voir le cours.*

---

2 Centre aéré (20 minutes)

Un centre aéré souhaite créer sa base de données afin de gérer ses activités et ses animateurs. Pour cela, on crée la relation :

R(NoEnfant, NomEnfant, PrenEnfant, NoGroupe, NomGroupe, NomAccomp, Jour, NoActivite, TypeActivite)

Un n-uplet (*noenfant*, *nomenfant*, *prenenfant*, *nogroupe*, *nomgroupe*, *nomaccomp*, *jour*, *noactivite*, *typeactivite*) a pour signification que l'enfant de numéro *noenfant*, de nom *nomenfant* et de prénom *prenenfant*, appartient au groupe de numéro *nogroupe*, de nom *nomgroupe*, encadré par l'animateur de nom *nomaccomp*. Ce groupe participe le jour *jour* à l'activité numéro *noactivite* qui est de type *typeactivite* (football, tir à l'arc, randonnée, ...).

L'analyse de la situation a conduit à l'ensemble F de dépendances fonctionnelles suivant :

(NoEnfant) → (PrenEnfant, NomEnfant)
(NoEnfant) → (NoGroupe, NomAccomp)
(NoGroupe) → (NomGroupe,)
(NoGroupe) → (NomAccomp)
(NoGroupe, Jour) → (NoActivite, TypeActivite)
(NoActivite) → (TypeActivite)
(NoEnfant, Jour) → (NomGroupe)

Toutes vos réponses doivent être **justifiées**.

1. Calculez la couverture minimale de ces dépendances fonctionnelles.

Correction : *(NoEnfant, Jour) → (NomGroupe) n'est pas élémentaire, puisque NomGroupe ne dépend que de NoEnfant. On la change donc en (noEnfant) → (NomGroupe)*

À partir de (NoEnfant) → (NoGroupe) et de NoGroupe → (NomAccomp), par transitivité on obtient (NoEnfant) → (NomAccomp), on peut donc supprimer cette dernière relation. Attention de nombreux étudiants ont appliqué cette règle et en ont conclu que l'on pouvait supprimer NoGroupe → (NomAccomp), ce qui est faux puisqu'on perd une information !

---

2. Trouvez la ou les clés de la relation R.

Correction : *Une façon simple de répondre est de prendre tous les attributs à gauche des DF et de supprimer ceux qui dépendent d'un autre de ces attributs. On obtient donc NoEnfant, Jour.*

---

3. Proposez une décomposition de R en troisième forme normale, qui préserve l'information et les dépendances fonctionnelles.

Correction : *Là encore tout part des DF :*
NoEnfant → (PrenEnfant, NomEnfant)
NoEnfant → (NoGroupe)
NoGroupe → (NomGroupe)
NoGroupe → (NomAccomp)
(NoGroupe, Jour) → (NoActivite)
(NoActivite) → (TypeActivite)

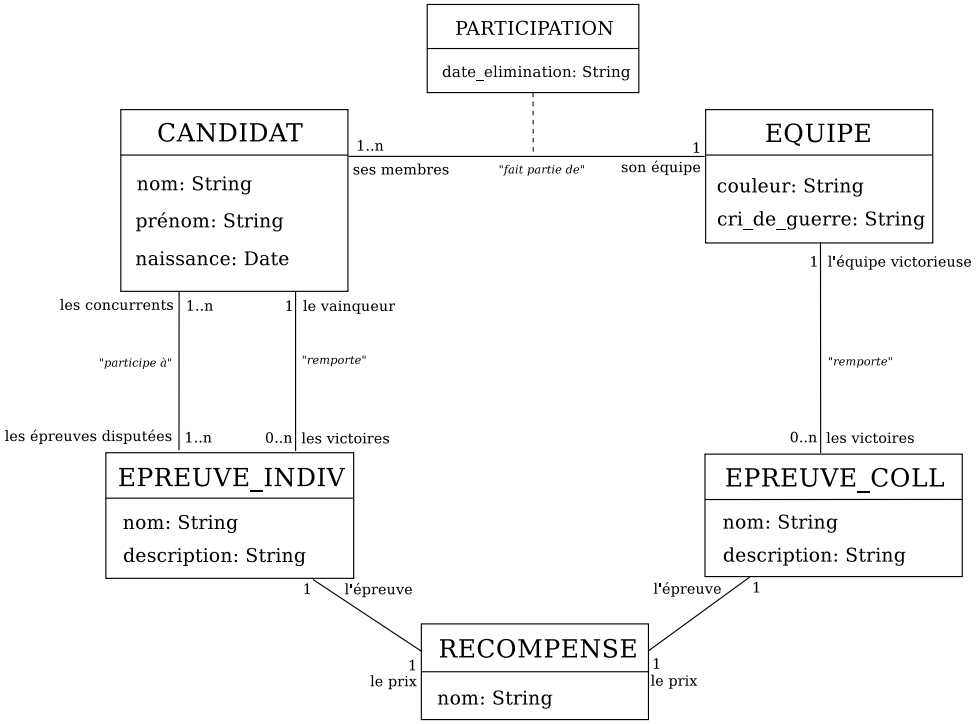
Chaque partie gauche doit être une clé d'une table, on a donc :
Enfant(NoEnfant, PrenEnfant, NomEnfant, NoGroupe)
Groupe(NoGroupe, NomGroupe, NomAccomp)
Activite(NoActivite, TypeActivite)
Activite_Enfant(NoGroupe, Jour, NoActivite)

---

3 Télé-réalité (15 minutes)

Une émission de télé-réalité regroupe des candidats répartis dans deux équipes. Au fil des jours, des épreuves ont lieu, soit individuelles (les candidats les uns contre les autres), soit collectives (les équipes l'une contre l'autre). Chaque épreuve est associée à une récompense et est remportée par une personne ou une équipe. Une fois par semaine, un candidat est éliminé, jusqu'à ce qu'il n'en reste qu'un.

Le diagramme de classes UML réalisé pour modéliser cette situation est le suivant (prenez-le tel quel, même s'il ne vous plaît pas).



Proposez un modèle relationnel correspondant à ce diagramme. Indiquez les clés primaires et étrangères.

Correction :

CANDIDAT(candidat_id, nom, prénom, naissance, date_elimination, equipe_id)
EQUIPE(equipe_id, couleur, cri_de_guerre)
PARTICIPANT_EPREUVE(nom_epreuve, candidat_id)
EPREUVE_INDIV(nom, description, recompense, candidat_id_victorieux)
EPREUVE_COLL(nom, description, recompense, equipe_id_victorieuse)

---

4 Commerce du vin (45 minutes)

On considère la base de données dont la structure est donnée par les commandes SQL ci-dessous :

```
CREATE TABLE viticulteur (n_viticulteur INTEGER NOT NULL AUTO_INCREMENT,
  nom VARCHAR(30), prenom VARCHAR(20),
  ville VARCHAR(30),
  PRIMARY KEY n_viticulteur);

CREATE TABLE vin (n_vin INTEGER NOT NULL AUTO_INCREMENT,
  cru VARCHAR(20), millesime INTEGER,
  region VARCHAR(15), n_viticulteur INTEGER, stock INTEGER,
  PRIMARY KEY n_vin,
  FOREIGN KEY n_viticulteur REFERENCES viticulteur(n_viticulteur),
  UNIQUE (cru, millesime, n_viticulteur));

CREATE TABLE buveur (n_buveur INTEGER NOT NULL AUTO_INCREMENT,
  nom VARCHAR(20), prenom VARCHAR(20), ville VARCHAR(30),
  PRIMARY KEY n_buveur);

CREATE TABLE commande (n_commande INTEGER NOT NULL AUTO_INCREMENT,
  n_buveur INTEGER, n_vin INTEGER, date DATE,
  quantite INTEGER,
  PRIMARY KEY n_commande,
  FOREIGN KEY n_buveur REFERENCES buveur(n_buveur),
```

```
FOREIGN KEY n_vin REFERENCES vin(n_vin),
UNIQUE (n_buveur, n_vin, date));
```

Point vocabulaire :

- Un viticulteur est une personne qui produit du vin.
- Le cru d'un vin est une sorte de catégorie associée au vin ; un vin est produit dans une région particulière.
- Le millésime est l'année de production d'un vin (le vin n'étant produit qu'une fois par an).
- Un buveur est une personne qui commande (achète) du vin.

Questions :

1. Dessinez un diagramme UML correspondant au modèle relationnel décrit par ce code SQL.
2. Répondez par vrai ou faux aux assertions suivantes. **Expliquez**.
   (a) Deux viticulteurs peuvent avoir la même valeur `n_viticulteur`.

   Correction : *Faux (c'est une clé primaire)*

   (b) Plusieurs viticulteurs peuvent produire un vin des mêmes cru et millésime.

   Correction : *Vrai (le triplet `(cru, millesime, n_viticulteur)` est UNIQUE mais pas `(cru, millesime)`)*

   (c) L'attribut `viticulteur.n_viticulteur` est une clé étrangère faisant référence à l'attribut `n_viticulteur` de la table `vin`.

   Correction : *Faux (c'est le contraire).*

   (d) Si on veut commander plusieurs vins différents, on passe plusieurs commandes d'affilée.

   Correction : *Vrai (`n_commande` est une clé primaire).*

3. Une contrainte supplémentaire est ajoutée au système : une commande ne peut concerner que 6, 12 ou 18 bouteilles. Que doit-on ajouter dans ou après la création de la table pour modéliser cette contrainte ? (réponse = le nom de la commande, soit un mot)

   Correction : *CHECK*

4. Expliquez ce que font les requêtes SQL suivantes (et n'oubliez pas de répondre à la question subsidiaire pour la R_7).

   R_1 `SELECT DISTINCT cru FROM vin;`

   Correction : *Renvoie tous les crus de la base.*

   R_2 `SELECT COUNT(*) FROM buveur;`

   Correction : *Renvoie le nombre de buveurs.*

R_3
```
SELECT nom, prenom, cru, millesime FROM viticulteur
  JOIN vin ON viticulteur.n_viticulteur = bin.n_viticulteur;
```

Correction : *Renvoie les crus et millésimes produit par chaque viticulteur.*

R_4
```
SELECT nom, prenom, cru, millesime FROM buveur
  JOIN commande ON buveur.n_buveur = command.n_buveur
  JOIN vin ON commande.n_vin = vin.n_vin;
```

Correction : *Renvoie les crus et millésimes commandés par chaque buveur.*

R_5
```
SELECT n_buveur FROM commande GROUP BY n_buveur HAVING COUNT(*) > 10;
```

Correction : *Renvoie les buveurs ayant passé plus de 10 commandes*

R_6
```
SELECT * FROM buveur WHERE prenom LIKE "Géra%" AND nom LIKE "%pardieu";
```

Correction : *Renvoie les buveurs dont le nom finit par "pardieu" et le prénom commence par "Géra".*

R_7
```
INSERT INTO buveur VALUES (5, 'Charles', 'Frédérique', 'Paris')
```
(Quelle sera la valeur de la cré primaire pour le nouvel élément inséré ?)

Correction : *Insère une nouvelle buveuse. Son identifiant est auto-incrémenté donc c'est le SGBD qui décide (ce ne sera pas 5).*

R_8
```
INSERT INTO commande
SELECT NULL, n_buveur, n_vin, NOW(), 6 FROM commande
WHERE n_vin = 12 GROUP BY n_buveur HAVING COUNT(*) > 10;
```
sachant que la fonction `NOW()` renvoie la date du jour

Correction : *Ajoute une commande de 6 bouteilles du vin 12 aux buveurs en ayant déjà commandé plus de 10 fois*