Examen final

Master 1 Ingénierie Mathématique

Bases de données relationnelles

Benjamin Auder

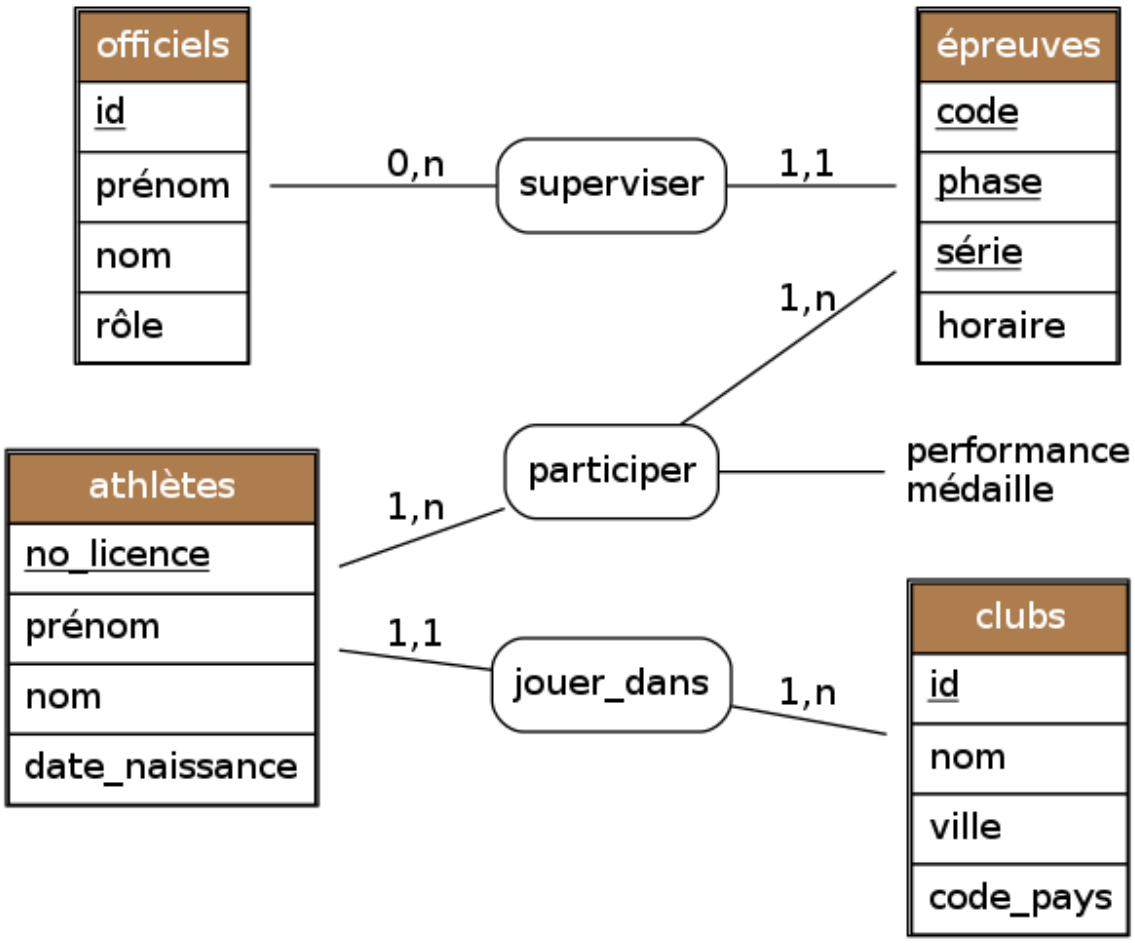
25 mars 2015

Exercice 0 [Environ 0% des points]

Tux peut-il voler ? Justifiez la réponse.

Exercice 1 [Environ 30% des points]

Cet exercice analyse une modélisation simplifiée du déroulement d'un meeting d'athlétisme. Comme l'indique le schéma entités/associations ci-dessous, on enregistre les performances des athlètes sur des épreuves (saut en longueur, lancer de poids, 800m ...), qui sont supervisées par des "officiels". Ce dernier terme désigne toute personne impliquée dans la compétition mais n'y participant pas ; les tâches sont diverses : chronométrage, contrôle anti-dopage, vérification du bon déroulement d'une épreuve ...



Une épreuve a un code, et est divisée en phases : de "1/8$^{\text{ème}}$ de finale" à "finale". Chaque phase comporte plusieurs séries (par exemple les séries de la demi-finale du 100m). Ces trois valeurs constitue la clé de l'entité.

(a) Répondez aux questions selon les indications du schéma.
Justifiez (brièvement mais précisément).

1. Un athlète peut-il être affilié à plusieurs clubs ?
2. Peut-il y avoir des clubs sans athlètes ?
3. Peut-il y avoir des officiels qui ne supervisent pas d'épreuves ?

(b) Comment étendre le schéma pour prendre en compte la participation d'équipes à certaines épreuves (le relais 4x100m par exemple) ? On ne prendra pas en compte cette extension pour les questions suivantes.

(c) Déterminez le schéma relationnel à partir du schéma E/A en commentant les opérations effectuées.

(d) Déduisez-en les instructions SQL nécessaires à la création des tables. Note : afin d'alléger la réponse *vous pouvez omettre les tables* `clubs`, `officiels` *et* `athlètes`, mais les autres tables utiles doivent être indiquées.

(e) Construisez une requête SQL renvoyant deux colonnes (nomClub, nbMédailles) correspondant respectivement aux noms des clubs et nombres de médailles (par club). La sortie doit être triée par valeurs de *nbMédailles* décroissantes, et exclure les clubs sans médailles.

Exercice 2 [Environ 40% des points]

Début d'une partie entre Magnus Carlsen et Levon Aronian au tournoi de Linares en 2007

Les questions de cet exercice portent sur la base de données "chess" contenant des résultats de parties d'échecs, ayant eu lieu dans des tournois. Vous n'avez pas besoin de

savoir jouer pour répondre, les commentaires ci-après suffisent.

Description des tables :

```
joueurs (
    id SERIAL PRIMARY KEY,
    nom VARCHAR(32) UNIQUE NOT NULL,
    prénom VARCHAR(32) NOT NULL,
    -- Année de naissance.
    année INTEGER NOT NULL
)

-- Lors d'un tournoi chaque participant a au moins une fois
-- les blancs, et au moins une fois les noirs.
tournois (
    id SERIAL PRIMARY KEY,
    -- Souvent le nom de la ville dans laquelle le tournoi a lieu.
    nom VARCHAR(32) NOT NULL,
    année INTEGER NOT NULL,
    UNIQUE (nom, année)
)

parties (
    id_tournoi INTEGER REFERENCES tournois(id),
    -- Un tournoi est divisé en rondes. Pendant une ronde,
    -- chaque participant joue au plus une partie.
    ronde INTEGER,
    -- "blancs" et "noirs" désignent les identifiants des joueurs
    -- s'affrontant (respectivement avec les pièces blanches et noires).
    blancs INTEGER REFERENCES joueurs(id),
    noirs INTEGER REFERENCES joueurs(id),
    -- "elo_b" (resp. "elo_n") ci-dessous désigne le classement
    -- du joueur des blancs (resp. des noirs).
    elo_b INTEGER NOT NULL,
    elo_n INTEGER NOT NULL,
    -- Le résultat est un type énuméré à trois valeurs :
    --   1-0 = les blancs gagnent,
    --   0-1 = les noirs gagnent,
    --   1/2 = match nul.
    résultat ChessResult NOT NULL,
    PRIMARY KEY (id_tournoi, ronde, blancs, noirs)
)
```

Rappel : il y a deux possibilités de connexion à la base pour tester vos réponses.

1. ssh login@192.168.31.236 puis psql -d chess, ou

2. naviguer à l'adresse http://192.168.31.236 depuis la salle d'examen.

Les réponses doivent être rendues au format électronique.

(a) Tournois (nom, année) dont le nom n'apparaît qu'une seule fois dans la table.

(b) Liste des participants (prénom, nom) au tournoi de Dortmund en 2005.

(c) Pourcentages de gains blancs, parties nulles et gains noirs sur l'ensemble des parties.

(d) Moyenne des âges des participants aux tournois de (nom =) 'Linares' − toutes années confondues −, *relativement à la date du tournoi*.

(e) Meilleure performance (joueur A gagnant contre B, avec A moins fort que B) réalisée au tournoi de 'WijkaanZee' en 2001. Renvoyez les noms des joueurs, leurs classements et le résultat.

(f) Nombre de points obtenus par Alexander Morozevich lors du tournoi de Sarajevo en 1999. On compte
1 pour les victoires,
1/2 pour les parties nulles,
0 pour les défaites.

(g) Joueurs (prénom, nom) ayant battu Garry Kasparov.

(h) On définit le "nombre de Kasparov" comme suit :
0 pour Garry Kasparov lui-même,
1 pour les joueurs l'ayant battu,
2 pour les gens ayant battu quelqu'un qui a battu Kasparov (mais n'ayant pas réalisé cette performance eux-mêmes),
...etc,
$+\infty$ si aucun entier n'est valide.
Écrivez une requête qui calcule le nombre de Kasparov.

Exercice 3 [Environ 10% des points]

Cet exercice porte sur une petite base de données (non construite : à vous de copier-coller les instructions de création ci-dessous dans votre base), contenant des résultats d'étudiants à un examen dans plusieurs matières.

Voici les instructions de création des tables :

```sql
-- Les etudiants.
CREATE TABLE students(
    id INTEGER PRIMARY KEY,
    firstname VARCHAR(32) NOT NULL,
    lastname VARCHAR(32) NOT NULL
);

-- Les cours.
CREATE TABLE courses(
    code INTEGER PRIMARY KEY,
    name VARCHAR(32) UNIQUE NOT NULL
);

-- Les notes, comprises entre 0 et 20.
-- **********************************
-- Remarque : on considère que la note peut être nulle, pour distinguer
-- par exemple un étudiant autorisé à compenser la matière d'un autre
-- oblige de repasser le module.
CREATE TABLE grades(
    student_id INTEGER REFERENCES students(id),
    course_code INTEGER REFERENCES courses(code),
    grade NUMERIC CHECK (grade >= 0 AND grade <= 20),
    PRIMARY KEY (student_id, course_code)
);
```

Quelques instructions possibles pour insérer des données :

```sql
INSERT INTO students VALUES
    (1, 'Maria', 'McConnell'),(2, 'Bernard', 'Young'),
    (3, 'Anthony', 'McCann'),(4, 'Lauren', 'Hill');

INSERT INTO courses VALUES
    (1, 'History'),(2, 'Mathematics'),
    (3, 'Chinese'),(4, 'Chemistry');

INSERT INTO grades VALUES
    -- Je laisse quelques lignes non remplies,
    -- pour tester le trigger de la premiere question.
    (1, 1, 10.0),(1, 2, 15.0),
    (2, 2, 12.0),(2, 3, 7.0),
    (3, 3, 18.0),(3, 4, 9.0),
    (4, 4, 13.0),(4, 1, 14.0);
```

(J'utilise des termes anglais pour éviter les accents, qui passent mal lors du copier-coller depuis un document PDF.)

Les réponses doivent être rendues au format électronique.

(a) Requête SQL renvoyant deux colonnes : les notes moyennes par matière, et les noms des matières correspondantes. Sur l'exemple elle doit retourner 12/20 en histoire, 13.5 en maths, 12.5 en chinois et 11.0 en chimie.

(b) On ajoute – comme indiqué ci-après – une colonne à la table *courses*, contenant les moyennes obtenues par matières.

```
ALTER TABLE courses
    ADD COLUMN mean NUMERIC;
```

Implémentez une fonction trigger nommée "computeMean" qui, à l'insertion/suppression ou mise à jour d'une note, (re)calcule la moyenne de la matière concernée. Après l'opération, la valeur moyenne reportée dans la table *courses* doit être correcte.

Exercice 4 [Environ 20% des points]

Cette exercice considère une table unique T contenant des données relatives à des cours de ski ou surf, dans une station de sports d'hiver. La table T est constituée de 6 attributs :

nom_moniteur	[Nm]	: le nom d'un moniteur
niveau	[N]	: le niveau d'un cours,
materiel	[M]	: le type de matériel utilisé (ski ou surf),
lieu_depart_arrivee	[L]	: lieu du cours,
heure_debut	[Hd]	: horaire de début,
heure_fin	[Hf]	: horaire de fin.

On suppose les dépendances fonctionnelles suivantes :

Nm $\rightarrow$ M
N $\rightarrow$ L
Nm, N $\rightarrow$ L, Hd
N, Hd $\rightarrow$ Hf

(a) Donnez en justifiant une clé minimale de la relation. Y en a-t-il d'autres ?
On considère cette clé pour la question suivante.

(b) Quelles formes normales vérifie la table T ? Peut-on aller plus loin (par application des deux théorèmes vus au second cours) ? Décrivez les étapes nécessaires.

(c) Lors de l'insertion d'un cours, il faut s'assurer qu'il est compatible avec les autres donnés par le moniteur. En particulier les plages horaires ne peuvent pas se chevaucher (les cours ont lieu tous les jours de la semaine). On considère que :

— deux cours peuvent s'enchaîner sans pause s'ils partent du même endroit,
— mais doivent être espacés d'au moins 30min dans le cas contraire.

L'objectif de cette question est de vérifier ces contraintes, qui devront être ajoutés directement dans la table (pas besoin d'un trigger).

Voici l'instruction de création de la table T :

```
CREATE TABLE cours_ski(
    nom_moniteur VARCHAR(32),
    niveau INTEGER,
    PRIMARY KEY(nom_moniteur, niveau),
    materiel VARCHAR(8) NOT NULL
    CONSTRAINT checkMateriel CHECK
        (materiel = any(ARRAY['ski', 'surf'])),
    lieu_depart_arrivee VARCHAR(32) NOT NULL,
    heure_debut TIME NOT NULL,
    heure_fin TIME NOT NULL,
    -- Un cours doit durer au moins 30 minutes.
    CONSTRAINT checkTime CHECK (heure_fin - heure_debut > '0:30')
);
```

Et quelques instructions d'insertion de données possibles, pour les tests :

```
INSERT INTO cours_ski VALUES(
  'Jade', 1, 'surf', 'pistes debutants', '11:00', '12:00');
INSERT INTO cours_ski VALUES(
  'Milla', 2, 'ski', 'telesiege chamois', '10:00', '12:00');
INSERT INTO cours_ski VALUES(
  'Andrew', 1, 'ski', 'pistes debutants', '13:00', '14:00');

INSERT INTO cours_ski VALUES(
  'Jade', 2, 'surf', 'teleski dromadaire', '8:45', '10:45'); -- echec
INSERT INTO cours_ski VALUES(
  'Jade', 1, 'surf', 'pistes debutants', '10:00', '11:00'); -- OK
INSERT INTO cours_ski VALUES(
  'Milla', 3, 'ski', 'telesiege chamois', '9:30', '11:00'); -- echec
INSERT INTO cours_ski VALUES(
  'Andrew', 2, 'ski', 'teleski dromadaire', '14:45', '16:45'); -- OK
```

Rappel : la réponse à cette question **doit être rendue au format électronique**.