DISVE Licence	Année Universitaire 2007/2008 Session 1 de Printemps **PARCOURS :** CSB4 & CSB6 **UE :** INF 159, Bases de données **Épreuve :** INF 159 EX **Date :** Mercredi 23 avril 2008 **Heure :** 8 heures 30 **Durée :** 1 heure 30 Documents : non autorisés Épreuve de M. Alain Griffault	

Code d'anonymat :

Avertissement

- La plupart des questions sont indépendantes.
- Le barème total est de 22 points car le sujet est assez long.
- Le barème de chaque question est (approximativement) proportionnel à sa difficulté.
- L'espace pour répondre est suffisant (sauf si vous l'utilisez comme brouillon, ce qui est fortement déconseillé).

Le sujet

Le sujet porte sur une gestion simplifiée d'un tournoi de tennis de table. Un tournoi comprend plusieurs tableaux. Chaque tableau est un ensemble de parties. Chaque partie d'un tableau oppose deux joueurs inscrits. Un joueur inscrit est un joueur licencié dans un pays.
Soit `Tournoi(Jo1_Licence, Jo1_Nom, Jo1_Points, Jo1_Ne_le, Jo1_Iso_pays, Jo1_Pays, Jo2_Licence, Jo2_Nom, Jo2_Points, Jo2_Ne_le, Jo2_Iso_pays, Jo2_Pays, Tableau, Tab_Intitule, Tab_PointsMax, Tab_Ne_apres, Jo1_Gagne)` une relation. Un exemple est donné dans la table suivante, où vu le nombre important d'attributs, les colonnes et lignes sont inversées.

Attributs	Commentaires	partie 1	partie 2	partie 3
Jo1_Licence integer	Numéro du joueur 1	41	48	51
Jo1_Nom text	Son nom	Eloi	Karakasevic	Gardos
Jo1_Points integer	Sa valeur	11833	11776	11755
Jo1_Ne_le date	Sa date de naissance	1975-01-01	1975-01-01	1975-01-01
Jo1_Iso_pays char(3)	Le code de son pays	FRA	SRB	AUT
Jo1_Pays text	Son pays	France	Serbie	Autriche
Jo2_Licence integer	Numéro du joueur 2	58	58	62
Jo2_Nom text	Son nom	Chila	Chila	Legout
Jo2_Points integer	Sa valeur	11701	11701	11693
Jo2_Ne_le date	Sa date de naissance	1975-01-01	1975-01-01	1975-01-01
Jo2_Iso_pays char(3)	Le code de son pays	FRA	FRA	FRA
Jo2_Pays text	Son pays	France	France	France
Tableau char(1)	Un identificateur	C	C	A
Tab_Intitule text	Son intitulé	Qualif JO	Qualif JO	Qualif JO
Tab_PointsMax integer	Valeur limite	12000	12000	12000
Tab_Ne_apres date	Naissance limite	1900-01-01	1900-01-01	1900-01-01
Jo1_Gagne boolean	Vrai=(joueur1 gagne)	False	False	True

Interprétation : La première partie indique que le français `Eloi` a perdu contre le français `Chila` dans le tableau `C`, la seconde indique que le serbe `Karakasevic` a perdu contre le français `Chila` dans le tableau `C` et la troisième indique que l'autrichien `Gardos` a gagné contre le français `Legout` dans le tableau `A`.

Remarque : Les données sont réelles sauf pour les dates de naissance des joueurs.

Pour une gestion correcte, la relation est munie des contraintes d'intégrité suivantes :

C1 : Tous les attributs sont déclarés `NON NULL`.

C2 : `PRIMARY KEY (Jo1_Licence, Jo2_Licence, Tableau)` qui indique la clef primaire.

C3 : `CONSTRAINT Jo1_age_points CHECK(Jo1_Ne_le > Tab_Ne_apres AND Jo1_Points < Tab_PointsMax)` qui indique que le joueur1 n'est pas trop âgé pour pouvoir participer à ce tableau et qu'il n'est pas trop fort pour pouvoir participer à ce tableau.

C4 : `CONSTRAINT Jo2_age_points CHECK(Jo2_Ne_le > Tab_Ne_apres AND Jo2_Points < Tab_PointsMax)` qui indique que le joueur2 n'est pas trop âgé pour pouvoir participer à ce tableau et qu'il n'est pas trop fort pour pouvoir participer à ce tableau.

C5 : `CONSTRAINT Jo1_Jo2 CHECK(Jo1_Licence < Jo2_Licence)` qui indique que (conjugué avec la clef primaire) deux mêmes joueurs ne peuvent pas se rencontrer deux fois dans le même tableau.

Exercice 1 (Requêtes SQL sur `Tournoi` (9 points))

Question 1.1 (1 point) *Écrire une requête SQL qui calcule pour chaque* **`tableau`** *le nombre de parties disputées dans celui ci.*

Question 1.2 (1 point) *Soit l'expression algébrique*

$$\alpha[\mathit{Jo1_Licence}:\mathit{Licence},\ \mathit{Jo1_Nom}:\mathit{Nom}](\pi[\mathit{Jo1_Licence},\ \mathit{Jo1_Nom}](\sigma[\mathit{Jo1_Gagne}](\mathit{Tournoi}))) \cup \alpha[\mathit{Jo2_Licence}:\mathit{Licence},\ \mathit{Jo2_Nom}:\mathit{Nom}](\pi[\mathit{Jo2_Licence},\ \mathit{Jo2_Nom}](\sigma[\neg\ \mathit{Jo1_Gagne}](\mathit{Tournoi})))$$

Écrivez la requête SQL correspondante, puis expliquez ce qu'elle calcule.

Question 1.3 (1,5 point) *Écrire une expression algébrique qui calcule les licences et les noms des joueurs ayant gagné toutes leurs parties.*

Question 1.4 (1,5 point) *Écrire une requête SQL qui permet de vérifier que les contraintes C3, C4 et C5 sont satisfaites.*

Question 1.5 (2 points) *Écrire une requête SQL qui retourne les licences et les noms des joueurs ayant participé à tous les tableaux. Vous pouvez créer des vues intermédiaires.*

Question 1.6 (2 points) *Écrire une requête SQL qui classe pour chaque tableau les joueurs en fonction de leurs résultats (nombre de parties gagnées moins nombre de parties perdues). Vous pouvez créer des vues intermédiaires.*

Exercice 2 (Normalisation (9 points)) *Pour améliorer la gestion des tournois, un informaticien (ayant entendu parler de formes normales) propose de découper la relation* `Tournoi` *en deux relations* `Joueurs(Licence, Nom, Points, Ne_le, Iso_pays, Pays, Tableau, Tab_Intitule, Tab_PointsMax, Tab_Ne_apres)` *et* `Parties(Licence1, Licence2, Tableau, Jo1_Gagne)`*. L'exemple est alors représenté par les deux tables.*

Attributs	*Joueur 1*	*Joueur 2*	*Joueur 3*	*Joueur 4*	*Joueur 5*
Licence integer	*41*	*48*	*51*	*58*	*62*
Nom text	*Eloi*	*Karakasevic*	*Gardos*	*Chila*	*Legout*
Points integer	*11833*	*11776*	*11755*	*11701*	*11693*
Ne_le date	*1975-01-01*	*1975-01-01*	*1975-01-01*	*1975-01-01*	*1975-01-01*
Iso_pays char(3)	*FRA*	*SRB*	*AUT*	*FRA*	*FRA*
Pays text	*France*	*Serbie*	*Autriche*	*France*	*France*
Tableau char(1)	*C*	*C*	*A*	*C*	*A*
Tab_Intitule text	*Qualif JO*	*Qualif JO*	*Qualif JO*	*Qualif JO*	*Qualif JO*
Tab_PointsMax integer	*12000*	*12000*	*12000*	*12000*	*12000*
Tab_Ne_apres date	*1900-01-01*	*1900-01-01*	*1900-01-01*	*1900-01-01*	*1900-01-01*

Attributs	*Commentaires*	*partie 1*	*partie 2*	*partie 3*
Licence1 integer	*Numéro du joueur 1*	*41*	*48*	*51*
Licence2 integer	*Numéro du joueur 2*	*58*	*58*	*62*
Tableau char(1)	*Un identificateur*	*C*	*C*	*A*
Jo1_Gagne boolean	*Vrai=(joueur1 gagne)*	*False*	*False*	*True*

Question 2.1 (1 point) *Donnez pour chacune des contraintes d'intégrité (cf page 2) la nouvelle contrainte équivalente. Précisez la (ou les) relations concernées.*

Un second informaticien trouve pertinent le découpage, mais explique qu'il faut maintenant lister les dépendances fonctionnelles et les contraintes d'intégrité afin de finir la normalisation. La liste des dépendances fonctionnelles trouvées est la suivante.

DF1 : $\{$`Licence`$\} \rightarrow \{$`Nom, Points, Ne_Le, Iso_pays, Pays`$\}$

DF2 : $\{$`Iso_pays`$\} \rightarrow \{$`Pays`$\}$

DF3 : $\{$`Tableau`$\} \rightarrow \{$`Tab_Intitule, Tab_PointsMax, Tab_Ne_apres`$\}$

DF4 : $\{$`Licence1, Licence2, Tableau`$\} \rightarrow \{$`Jo1_Gagne`$\}$

Question 2.2 (1 point) *Donnez toutes les clefs candidates de la relation* `Joueurs`*.*

Question 2.3 (1 point) *Donnez toutes les clefs candidates de la relation* `Parties`*.*

Question 2.4 (1 point) *Même si l'on suppose qu'il n'y a aucun doublon dans* `Joueurs`*, justifiez pourquoi la relation* `Joueurs` *n'est pas en troisième forme normale.*

Question 2.5 (1 point) *Justifiez que la relation* `Parties` *est BCNF.*

Question 2.6 (2 points) *Appliquez un algorithme (ou une technique) de normalisation pour obtenir une décomposition, sans perte d'information, de la relation* `Joueurs` *en un ensemble de relations BCNF. Vous n'écrirez sur la copie que les nouvelles relations et les dépendances fonctionnelles utilisées lors des projections effectuées.*

Question 2.7 (1 point) *Donnez pour chaque relation de votre normalisation, les contraintes d'intégrité et les clefs étrangères présentes.*

Question 2.8 (1 point) *Créez une vue* `TournoiEquiv` *à partir de votre décomposition qui contiennent exactement les tuples de la relation* `Tournoi` *initiale.*

Exercice 3 (Gestion des accès concurrents (4 points))

Question 3.1 (2 points) *Expliquez pourquoi une solution basée sur :*

1. *Une base accessible en lecture uniquement.*
2. *Une base pour les modifications.*
3. *Une recopie périodique de la base modifiée vers la base accessible en lecture.*

est sans inter-blocage et sans attente infinie.

Question 3.2 (2 points) *Soit une relation* `Reservation(jour date, train text, wagon integer, place integer, reservee boolean)` *pour la réservation de place dans les trains. Les applications pour la réservation sont disponibles dans des agences (un seul terminal par agence). Expliquez pourquoi la solution précédente n'est pas adaptée, puis proposez une mise en oeuvre sans verrou, sans inter-blocage, sans attente infinie, mais pas forcément optimale.*