Département Licence	ANNÉE : 2005/2006 SESSION DE JUIN 2006 **PARCOURS :** INF 3 **UE :** INF 109, Bases de données **Épreuve :** INF 109 EX **Date :** Mardi 20 juin 2006 **Heure :** 8 heures 30 **Durée :** 1 heure 30 Documents : non autorisés Épreuve de M. Alain GRIFFAULT

SUJET + CORRIGE

Avertissement

- La plupart des questions sont indépendantes.
- Le barème est (approximativement) proportionnel à la difficulté de la question.
- L'espace laissé pour les réponses est suffisant (sauf si vous utilisez ces feuilles comme brouillon, ce qui est fortement déconseillé).

Le sujet

Le sujet porte sur une gestion simplifiée des vainqueurs d'épreuves individuelles aux jeux olympiques d'été.

Soit $JO(Licence, Vainqueur, Annee, Ville, Pays, Epreuve, Engages, RecordMondial)$ une relation et ses dépendances fonctionnelles :

- $\{Annee\} \longrightarrow \{Ville, Pays\}$ qui indique que l'année détermine la ville et le pays organisateurs.
- $\{Ville\} \longrightarrow \{Pays\}$ qui indique que les villes ne changent jamais de pays.
- $\{Annee, Epreuve\} \longrightarrow \{Vainqueur, Engages, RecordMondial\}$ qui indique qu'à chaque Jeux Olympiques, pour chaque épreuve individuelle, il y a un seul vainqueur. On enregistre également le nombre d'engagés et si le record mondial de la discipline a été battu pendant les Jeux Olympiques.
- $\{Vainqueur, Epreuve\} \longrightarrow \{Annee\}$ qui indique que le vainqueur d'une épreuve détermine l'année. Une autre interprétation est qu'un athlète n'a pas le droit de participer à une épreuve qu'il a déjà gagné. *Cette règle n'existe que dans le cadre de ce sujet.*
- $\{Licence\} \longrightarrow \{Vainqueur\}$ qui indique que la licence est attribuée à vie à un athlète.
- $\{Vainqueur\} \longrightarrow \{Licence\}$ qui indique qu'il n'y a pas deux athlètes ayant le même nom.

Par exemple

Licence	Vainqueur	Annee	Ville	Pays	Epreuve	Engages	RecordMondial
254	BEAMON	1968	Mexico	Mexique	SautLongueur	35	Oui
915	LEWIS	1984	Los Angeles	USA	100 m	82	Non

spécifie que

- BEAMON possède la licence 254 ; qu'en 1968, à Mexico au Mexique, parmi 35 concurrents, il a gagné le saut en longueur et que le record du monde a été battu.
- LEWIS possède la licence 915 ; qu'en 1984, à Los Angeles aux USA, parmi 82 concurrents, il a gagné le 100 m et que le record du monde n'a pas été battu.

Exercice 1 (Requêtes SQL sur *JO* (6 points))

Question 1.1 (1 point) *Écrire une requête SQL qui caractérise les "Ville" dans lesquelles au moins un record du monde a été battu.*

Réponse :

```
SELECT DISTINCT Ville
FROM            JO
WHERE           RecordMondial = 'True'; -- ou 'Yes', 'T', ...
```

Question 1.2 (1 point) *Écrire une requête SQL qui caractérise les "Epreuve" de l'année 1968 pour lesquelles le record du monde a été battu cette année là.*

Réponse :

```
SELECT DISTINCT Epreuve
FROM            JO
WHERE           Annee = 1968
  AND           RecordMondial; -- = 'True' est inutile car c'est un booleen
```

Question 1.3 (1 point) *Écrire une requête SQL qui caractérise les "Vainqueur" d'au moins une épreuve en 1960 et d'au moins une épreuve en 1964.*

Réponse :

```
-- Une vision algèbrique
SELECT DISTINCT Vainqueur
FROM            JO
WHERE           Annee = 1960
INTERSECT
SELECT DISTINCT Vainqueur
FROM            JO
WHERE           Annee = 1964;

-- Une vision prédicative
SELECT DISTINCT JO1.Vainqueur
FROM            JO AS JO1, JO AS JO2
WHERE           JO1.Vainqueur = JO2.Vainqueur
  AND           JO1.Annee = 1960
  AND           JO2.Annee = 1964;
```

Question 1.4 (1 point) *Écrire une requête SQL qui caractérise les "Vainqueur" et le nombre d'épreuves différentes remportées. Le résultat sera ordonné par ce nombre.*

Réponse :

```
SELECT DISTINCT Vainqueur, COUNT(DISTINCT Epreuve)
FROM            JO
GROUP BY        Vainqueur
ORDER BY        COUNT(DISTINCT Epreuve) DESC;
```

Question 1.5 (2 points) *En supposant que* LEWIS *et l'"Annee" 1984 soient déjà dans la base ; écrire une requête SQL qui ajoute l'information qu'en 1984, parmi 27 compétiteurs,* LEWIS *a gagné le saut en longueur et le record du monde n'a pas été battu.*

Réponse : *La présence de* LEWIS *permet de récupérer sa licence, et celle de "Annee" de récupérer la ville et la pays.*

```
INSERT INTO JO
  SELECT Licence, Vainqueur, Annee, Ville, Pays, 'SautLongueur', 27, 'No'
  FROM   (SELECT          DISTINCT Licence, Vainqueur
          FROM            JO
          WHERE           Vainqueur = 'Lewis')
         AS TEMP1,
         (SELECT          DISTINCT Annee, Ville, Pays
          FROM            JO
          WHERE           Annee = 1984)
         AS TEMP2;
```

Exercice 2 (Normalisation (6 points))

Question 2.1 (1 point) *Donnez toutes les clefs candidates de la relation "JO".*

Réponse :
- $C_1 = \{Licence, Epreuve\}$
- $C_2 = \{Vainqueur, Epreuve\}$
- $C_3 = \{Annee, Epreuve\}$

Question 2.2 (1 point) *Même si l'on suppose qu'il n'y a aucun doublon dans "JO", justifiez pourquoi la relation "JO" n'est pas en troisième forme normale.*

Réponse : *Une seule des explications suivantes est suffisante.*

Non 2NF : $\{Licence\} \longrightarrow \{Vainqueur\}$ *alors que "Licence" appartient à une clef candidate.*

Non 2NF : $\{Vainqueur\} \longrightarrow \{Licence\}$ *alors que "Vainqueur" appartient à une clef candidate.*

Non 3NF : $\{Ville\} \longrightarrow \{Pays\}$ *alors que "Ville" et "Pays" n'appartiennent à aucune clef candidate.*

Question 2.3 (2 points) *Appliquez un algorithme de normalisation pour obtenir une décomposition, sans perte d'information et sans perte de dépendance fonctionnelle, de la relation "JO" en un ensemble de relations en troisième forme normale.*
Vous n'écrirez sur la copie que les nouvelles relations et les dépendances fonctionnelles qui sont à la base des projections effectuées.

Réponse :
- $Lieu(Ville, Pays)$ *qui provient de* $\{Ville\} \longrightarrow \{Pays\}$
- $Olympiade(Annee, Ville)$ *qui provient de* $\{Annee\} \longrightarrow \{Ville\}$
- $Athlete(Licence, Vainqueur)$ *qui provient de* $\{Licence\} \longrightarrow \{Vainqueur\}$ *ou de* $\{Vainqueur\} \longrightarrow \{Licence\}$
- $Competition(Epreuve, Annee, Vainqueur, Engages, RecordMondial)$ *qui provient de* $\{Epreuve, Annee\} \longrightarrow \{Vainqueur, Engages, RecordMondial\}$ *ou de* $\{Epreuve, Vainqueur\} \longrightarrow \{Annee, Engages, RecordMondial\}$

Question 2.4 (1 point) *Donnez une contrainte sur l'attribut "Annee" et une sur l'attribut "Engages". Indiquez comment il est possible de les décrire en SQL.*

Réponse : *Le nombre d'engagés est toujours strictement positif, et les années sont des multiples de 4. Dans la relation qui contient l'attribut "Engages", il suffit d'ajouter la contrainte* `CHECK (Engages > 0)` *; et dans la relation qui contient l'attribut "Annee", il suffit d'ajouter la contrainte* `CHECK (Annee % 4 = 0)`.

Question 2.5 (1 point) *Indiquez, pour vos relations, les éventuelles clefs étrangères. Pour chacune, précisez, s'il y a lieu, les conséquences (traitements, actions, ...) en cas de suppression de tuples.*

Réponse : *Il fallait parler des lignes* `REFERENCES` *présentes dans le schéma suivant.*

```
CREATE TABLE Athlete (
  Vainqueur       char(30) PRIMARY KEY,      -- ou UNIQUE NOT NULL
  Licence         integer  UNIQUE NOT NULL   -- ou PRIMARY KEY
);
CREATE TABLE Lieu (
  Ville           char(20) PRIMARY KEY,
  Pays            char(20) NOT NULL
);
CREATE TABLE Olympiade (
  Annee           integer  PRIMARY KEY CHECK (Annee % 4 = 0),
  Ville           char(20) NOT NULL REFERENCES Lieu(Ville) ON DELETE CASCADE
);
CREATE TABLE Competition (
  Annee           integer  REFERENCES Olympiade(Annee) ON DELETE CASCADE,
  Epreuve         char(20),
  -- Il faut soit Vainqueur soit Licence pour la jointure avec Athlete
  -- Il est préférable de choisir la clef primaire de Athlete
  Vainqueur       char(30) REFERENCES Athlete(Vainqueur) ON DELETE CASCADE,
  Engages         integer  NOT NULL CHECK (Engages > 0),
  RecordMondial   boolean  NOT NULL,
  PRIMARY KEY (Annee, Epreuve),              -- ou (Epreuve, Vainqueur)
  UNIQUE (Epreuve, Vainqueur)                -- ou (Annee, Epreuve)
);
```

Exercice 3 (Requêtes SQL SUR VOTRE DÉCOMPOSITION (9 points))

Question 3.1 (1 point) *Écrire une requête SQL qui caractérise les "Ville" dans lesquelles au moins un record du monde a été battu.*

Réponse :

```
SELECT DISTINCT Ville
FROM            Lieu NATURAL JOIN Olympiade NATURAL JOIN Competition
WHERE           RecordMondial = 'True';
```

Question 3.2 (1 point) *Écrire une requête SQL qui caractérise les "Epreuve" de l'année 1968 pour lesquelles le record du monde a été battu cette année là.*

Réponse :

```
SELECT DISTINCT Epreuve
FROM            Competition
WHERE           Annee = 1968
  AND           RecordMondial;
```

Question 3.3 (1 point) *Écrire une requête SQL qui caractérise les "Vainqueur" d'au moins une épreuve en 1960 et d'au moins une épreuve en 1964.*

Réponse :

```
-- Une vision algèbrique
SELECT DISTINCT Vainqueur
FROM            Competition
WHERE           Annee = 1960
INTERSECT
SELECT DISTINCT Vainqueur
FROM            Competition
WHERE           Annee = 1964;

-- Une vision prédicative
SELECT DISTINCT J01.Vainqueur
FROM            Competition AS J01, Competition AS J02
WHERE           J01.Vainqueur = J02.Vainqueur
  AND           J01.Annee = 1960
  AND           J02.Annee = 1964;
```

Question 3.4 (1 point) *Écrire une requête SQL qui caractérise les "Vainqueur" et le nombre d'épreuves différentes remportées. Le résultat sera ordonné par ce nombre.*

Réponse :

```
SELECT DISTINCT Vainqueur, COUNT(DISTINCT Epreuve)
FROM            Competition
GROUP BY        Vainqueur
ORDER BY        COUNT(DISTINCT Epreuve) DESC;
```

Question 3.5 (1 point) *En supposant que* LEWIS *et l'"Annee" 1984 soient déjà dans la base; écrire une requête SQL qui ajoute l'information qu'en 1984, parmi 27 compétiteurs,* LEWIS *a gagné le saut en longueur et le record du monde n'a pas été battu.*

Réponse :

```
INSERT INTO Competition VALUES (1984, 'SautLongueur', 'Lewis', 27, 'No');
```

Question 3.6 (2 points) *Après en avoir donné une écriture algébrique, écrire une requête SQL qui caractérise les athlètes qui ont été "Vainqueur" d'au moins une épreuve toutes les "Annee".*

Réponse : *La division algébrique s'écrit* $R^1 / \ R^2 = \pi[A_i](R^1) - \pi[A_i]((\pi[A_i](R^1) \times R^2) - R^1)$.
Dans cette formule, il suffit de remplacer :
– R^1 *par* $\pi[Vainqueur, Annee](Competition)$,

– R^2 par $\pi[Annee](Olympiade)$,
puis d'effectuer la substitution dans la requête SQL vue en cours.

```
-- les noms des athletes vainqueurs lors de toutes les olympiades
-- Division algebrique avec R1(Annee, Vainqueur) et R2 (Annee)
SELECT DISTINCT Vainqueur
FROM (SELECT DISTINCT Annee, Vainqueur
      FROM   Competition
     ) AS R1_1
  EXCEPT
SELECT Vainqueur
FROM (SELECT *
      FROM   (SELECT DISTINCT Vainqueur
              FROM (SELECT DISTINCT Annee, Vainqueur
                    FROM   Competition
                   ) AS R1_2
             ) AS PiR1,
             (SELECT DISTINCT Annee
              FROM   Olympiade) AS R2
        EXCEPT
      SELECT *
      FROM   (SELECT DISTINCT Vainqueur, Annee
              FROM   Competition
             ) AS R1_3
     ) AS NonEntierR1;

-- Ce qui donne après simplification
SELECT DISTINCT Vainqueur
FROM Competition
  EXCEPT
SELECT  Vainqueur
FROM (SELECT *
      FROM (SELECT DISTINCT Vainqueur
            FROM Competition) AS PiR1,
           (SELECT DISTINCT Annee
            FROM   Olympiade) AS R2
        EXCEPT
      SELECT DISTINCT Vainqueur, Annee
      FROM   Competition
     ) AS NonEntierR1;
```

Question 3.7 (2 points) *Après en avoir donné une écriture prédicative, écrire une requête SQL qui caractérise les athlètes qui ont été "Vainqueur" de toutes les "Epreuve".*

Réponse : *La division prédicative s'écrit* $R^1/\ R^2 = \{x\ /\ \ \nexists y \in R2,\ \ \nexists(a,b) \in R1\ \wedge\ (x = a \wedge y = b)\}$.
Dans cette formule, il suffit de remplacer :
– R^1 par $\pi[Vainqueur, Epreuve](Competition))$,
– R^2 par $\pi[Epreuve](Competition)$,
puis d'effectuer la substitution dans la requête SQL vue en cours.

```
-- les athletes ayant gagnés toutes les épreuves
-- Division prédicative avec R1(Vainqueur, Epreuve) et R2 (Epreuve)
SELECT DISTINCT Vainqueur
FROM   (SELECT Vainqueur, Epreuve
        FROM   Competition) AS R1
WHERE  NOT EXISTS
       (SELECT *
        FROM  (SELECT Epreuve
               FROM   Competition) AS R2
        WHERE  NOT EXISTS
               (SELECT Epreuve
                FROM   (SELECT Vainqueur, Epreuve
```

```
                        FROM   Competition) AS R11
                WHERE  R11.Vainqueur = R1.Vainqueur
                  AND  R11.Epreuve   = R2.Epreuve));

-- Ce qui donne après simplification
SELECT DISTINCT Vainqueur
FROM   Competition AS C1
WHERE  NOT EXISTS
       (SELECT Epreuve
        FROM   Competition AS C2
        WHERE  NOT EXISTS
               (SELECT Epreuve
                FROM   Competition AS C3
                WHERE  C3.Vainqueur = C1.Vainqueur
                  AND  C3.Epreuve   = C2.Epreuve));
```