L2 INF245 — Examen
19 mai 2016, session 1

Durée : 2H
Documents autorisés : une feuille A4, recto-verso, manuscripte

1 À propos de relations

Soient R et S deux relations dont les schémas et les valeurs sont donnés dans les tableaux suivants :

R

A	B
1	11
2	20
4	21
3	10

S

B	C
10	100
8	110
11	101
10	101
11	100
12	100

T

A	C
1	110
8	100
1	101
0	102
1	100
2	100

Question 1 (4 points) :
Donner, sous forme de tableaux, le schéma et la valeur de la relation construite par chacune des requêtes ci-dessous appliquées aux deux relations R et S (il peut s'agir d'un message d'erreur Oracle).

1.
```
select C, B
from S where B <> 11;
```

2.
```
select R.A, R.B, S.B, S.C
from R join S on (R.B < S.B);
```

3.
```
select A, count(distinct B) as nbB,
          count(C) as nbC
from R natural join S group by A;
```

4.
```
select A, B, count(C) as nbC
from R natural join S group by A;
```

5.
```
select A,B from R natural join S
minus
select A,C from T
```

6.
```
select A,B from R
minus
select  R1.A, R1.B
from R R1 join R R2 on (R1.B > R2.B)
```

2 À propos de JO

On reprend, en le simplifiant un peu, le sujet JO étudié pendant le TP2.

2.1 Position du problème

Une *compétition* rassemble des sportifs de divers pays, hommes et femmes, qui se mesurent dans des épreuves de diverses *disciplines* (par exemple, ski nordique, sports de glace, etc.). Dans une discipline, il peut y avoir plusieurs *épreuves :* par exemple, en ski alpin, épreuves de descente, de slalom géant, etc.
Pour chaque **sportif**, identifié par son numéro, on connaît notamment son nom, son prénom, sa date de naissance, sa catégorie (genre), et le pays qu'il représente. On suppose que deux sportifs se distinguent par leur nom ou par leur prénom s'ils ont le même nom. Les sportifs d'un même pays forment la *délégation de ce pays*[1].

[1]Hypothèse simplificatrice : seuls les sportifs sont considérés dans cette version de l'application. En particulier, on ne tient pas compte de l'encadrement, ni des équipes médicales, etc.

Chaque **épreuve** a une *forme* et une *catégorie*. La forme d'une épreuve est *individuelle* ou par *équipe* (une équipe comporte au moins deux sportifs), ou par *couple* (un couple comporte exactement deux sportifs, une femme et un homme). Tous les sportifs d'une même équipe représentent le même pays. La plupart des épreuves par équipe est associée à un nombre fixé de sportifs, le même pour toutes les équipes. Le nombre de sportifs des équipes engagées est égal au nombre ainsi fixé. La *catégorie* d'une épreuve est *féminine*, *masculine* ou *mixte*. Les épreuves masculines confrontent des hommes et les épreuves féminines confrontent des femmes. Dans une épreuve par équipe, tous les sportifs d'une même équipe sont des hommes s'il s'agit d'une épreuve masculine ou des femmes s'il s'agit d'une épreuve féminine. Les épreuves par *couple*, confrontent des équipes *mixtes* composées d'exactement 2 participants : une femme et un homme. On prend comme hypothèse que chaque épreuve se déroule sur un seul jour. En particulier, on ne distingue pas d'épreuves préliminaires (des séries par exemple, ou 8e de finale, etc.).

Une même **équipe** ou un même **sportif** peut être engagé dans plusieurs épreuves de la même discipline. Un même sportif peut appartenir à plusieurs équipes engagées dans des épreuves de la même discipline, et être engagé aussi dans une ou plusieurs épreuves individuelles, de la même discipline.

A l'issue d'une épreuve des médailles sont attribuées : *or* pour le premier, *argent* pour le second, *bronze* pour le troisième. On suppose qu'il n'y a pas d'ex-aequo et que toutes les médailles sont attribuées (une épreuve comporte au moins trois inscrits).

Des billets peuvent être vendus pour le compte d'un utilisateur que l'on ne modélise pas dans cette version de l'application (on ne considère que des numéros d'utilisateur). Un billet permet à une personne d'assister à une épreuve donnée. Le prix d'un billet est fixé pour l'épreuve associée. Les billets vendus pour un utilisateur au cours de la même transaction sont associés à un dossier avec une date d'émission égale à celle de la transaction (à la granularité de la seconde). On prend pour hypothèse que le nombre de places que l'on peut vendre pour une épreuve n'a pas de limite.

2.2 Schéma de la base de données

LesSportifs (nSportif, nomS, prenomS, pays, categorieS, dateNais)
/ nomS,prenomS est une autre clef candidate. */*
/ (s, n, p, d, c, b) ∈ LesSportifs ⟺ le sportif dont le numéro est s a pour nom n et pour prénom p. Il représente le pays d, il est de la catégorie c et b est son jour de naissance. */*

LesEquipes (nSportif, nEquipe)
/ (s,e) ∈ LesEquipes ⟺ le sportif de numéro s est membre de l'équipe de numéro e. */*

LesDisciplines (discipline)
/ (d) ∈ LesDisciplines ⟺ d est le nom d'une discipline. */*

LesEpreuves (nEpreuve, nomE, forme, discipline, categorieE, nbs, dateEpreuve, prix)
/ (e, n, f, d, c, s, j, p) ∈ LesEpreuves ⟺ l'épreuve de numéro e a pour nom n et pour forme f, sa discipline est d, sa catégorie c et s, lorsqu'il est fixé, est le nombre de membres des équipes inscrites. Elle se déroule le jour j. Le prix d'une place pour assister à cette épreuve est p. */*

LesInscriptions (nInscrit, nEpreuve)
/ (i, e) ∈ LesInscriptions ⟺ l'équipe ou le sportif dont le numéro est i, participe à l'épreuve e. */*

LesResultats (nEpreuve, gold, silver, bronze)
/ (e, g, s, b) ∈ LesResultats ⟺ pour l'épreuve de numéro e, g (resp. s ou b) est le numéro (équipe ou sportif) ayant gagné la médaille d'or (resp. argent ou bronze). */*

LesDossiers_base (nDossier, nUtil, dateEmission)
/ (n, u, d) ∈ LesDossiers_base ⟺ le dossier de numéro n a été créé à la date d, pour le compte de l'utilisateur de numéro u. */*

LesBillets (nBillet, nDossier, nEpreuve)
/ (b, d, e) ∈ LesBillets ⟺ le billet b est associé au dossier d. Il correspond à l'achat de une place pour l'épreuve e. */*

Une vue LesDossiers qui donne pour chaque dossier le prix total de tous les billets associés :

LesDossiers (nDossier, nUtil, dateEmission, prix)
/ (n, u, d, p) ∈ LesDossiers ⟺ le dossier de numéro n a été créé pour le compte de l'utilisateur dont le login est u à la date d. Le prix total de tous les billets associés à n est p. */*

Les contraintes de domaine :

domain(nEquipe) = domain(nInscrit) = domain(nSportif) = domain(nUtil) = domain(gold) = domain(silver) = domain(bronze) = domain(nDossier) = domain(nBillet) = domain(nEpreuve) = domain(nbs) = domain(capacite) = integer > 0
domain(prix) = real ≥ 0
domain(nomE) = domain(pays) = domain(nomS) = domain(prenomS) = chaînes de caractères.
domain(categorieS) = {"féminin", "masculin"}
domain(categorieE) = {"féminin", "masculin", "mixte"}
domain(forme) = {"par équipe", "individuelle", "par couple" }
domain(dateNaissance) = domain(dateEpreuve) = date(jour)
domain(dateEmission) = date(seconde)
domain(discipline) = {"Ski alpin", "Bob", "Patinage artistique", ...}

2.3 Expression de requêtes

Indications : Les fonctions habituelles de comparaison sont étendues au type date. La fonction sysdate retourne un instant (à la granularité de la seconde), dont la valeur est observée au moment de son exécution. Exemples :

to_char (sysdate, 'Month YYYY') = 'Mai 2016'.

to_char(to_date('23/05/2015', 'DD/MM/YY'), 'DD-MM-YY HH:MI:SS') = '23-05-15 12:00:00'

Question 2 (6 points) :
Exprimer en SQL les requêtes ci-dessous.
Les requêtes devront construire des résultats sans répétition de valeurs, la clause distinct ne sera utilisée que lorsque nécessaire. Les produits de relation seront exprimés dans la clause from des requêtes. Toute sous-requête sera soigneusement spécifiée.

1. Quelles sont les épreuves (numéro et date de l'épreuve) pour lesquelles aucune place ne plus être vendue ? C'est-à-dire les épreuves passées.

2. Quelles sont les numéros des épreuves dans lesquelles le pays Italie a gagné au moins une médaille d'or par équipe ou en couple ?

3. Pour chaque épreuve (donnée par son numéro), combien y a-t-il de billets vendus ?

4. Donner pour chaque pays le nombre de médailles d'or, le nombre de médailles d'argent, et le nombre de médailles de bronze (lorsqu'une équipe gagne une médaille, celle-ci compte 1 pour le pays de l'équipe).

   Indication : introduire une relation T(pays, nInscrit) telle que <p,n> ∈ T ⟺ l'inscrit n, une équipe ou un sportif, est du pays p. Puis utiliser T dans la solution et fournir l'expression SQL qui construit T.

5. Chaque mois (du passé et du futur), pour chaque épreuve (donnée par son numéro), combien y a-t-il de billets vendus ? (un mois est donné par le nom du mois et le numéro de l'année : par exemple Mai 2016)

6. Chaque mois (du passé et du futur), quelle est l'épreuve (donnée par son numéro et son nom) pour laquelle il y a le plus de billets vendus ?

   Indication : introduire une relation construite par la requête 5, puis l'utiliser.

2.4 Mise en œuvre de l'application

Question 3 (3 points) :
Fournir le code SQL Oracle qui permet de définir les relations LesEpreuves, LesResultats, et LesSportifs.

Les autres relations sont définies dans l'annexe A.

3 A propos de comptes bancaires

On considère la base de données définie par le code SQL donné ci-dessous :

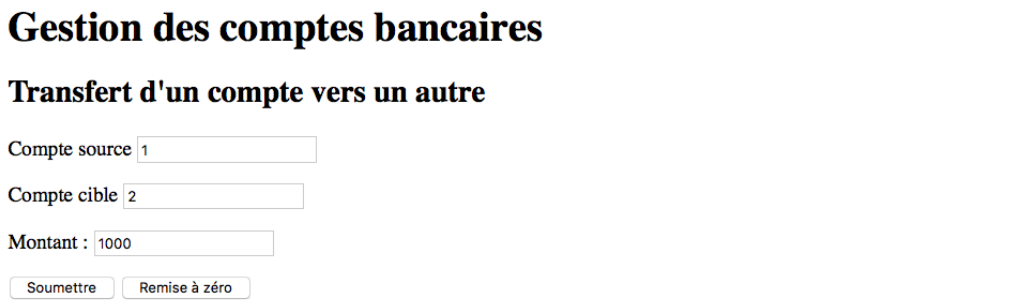
```
create table LesComptes (
   noC number, solde number,
   constraint comptes_pk primary key (noC),
   constraint comptes_ck check (solde >=0) );
```

Question 4 (4 points) :
Indiquer dans chacun des cas d'interaction ci-dessous, la valeur de la relation LesComptes et la page retournée par l'exécution du programme Transferer-action.php (voir annexe B) après que l'utilisateur ait cliqué sur le bouton Soumettre, et interprétée par un navigateur compatible html. Pour la première interaction, considérer la base de données dans l'état initial ci-dessous. Puis, considérer la base de données obtenue à l'issue de l'interaction précédente.

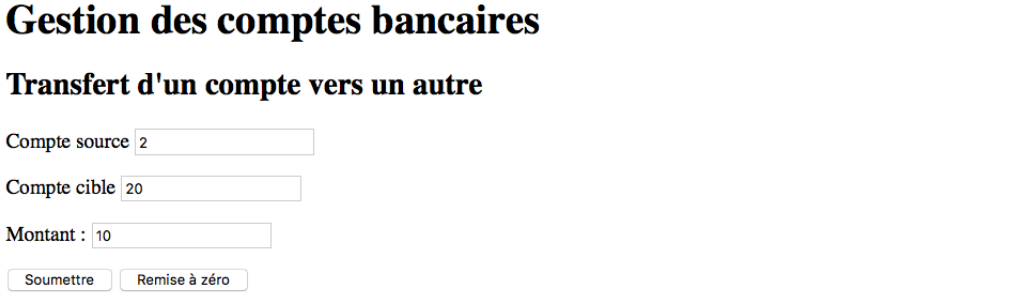
Valeur initiale de la relation LesComptes :

NOC	SOLDE
1	100
2	1
3	4701
4	5000

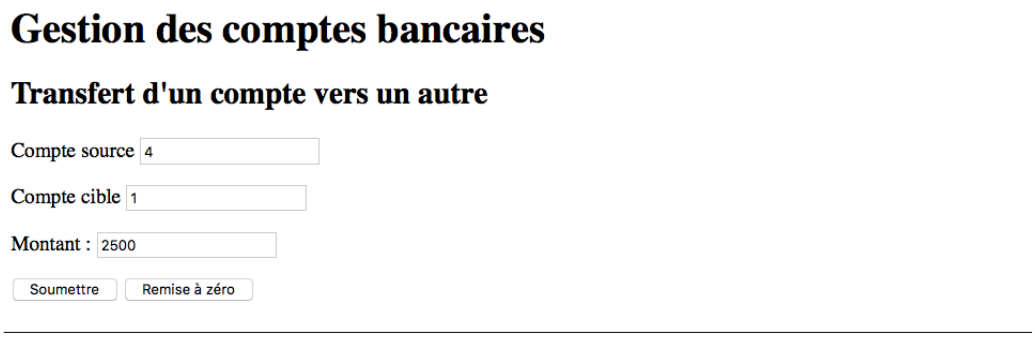
1.



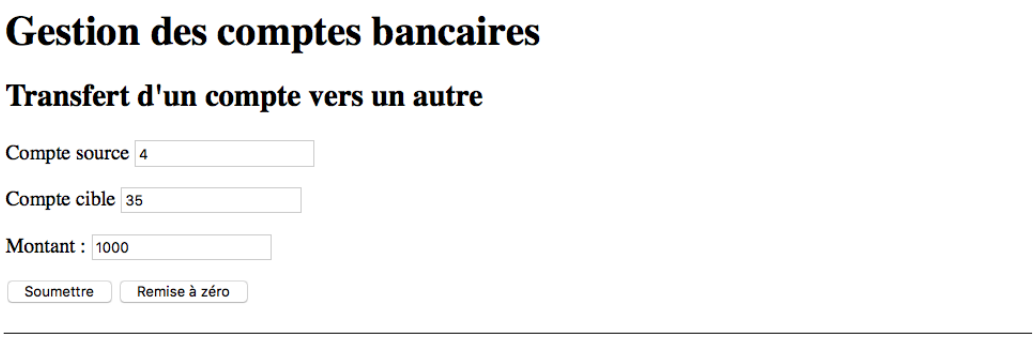
2.



3.



4.



Question 5 (3 points) :
Le programme php Transférer-action pose t-il problème ? Si oui, expliquer lequel et proposer une solution qui le corrige.

A Code SQL de création de certaines relations de la BD JO

```
create table LesDisciplines (discipline varchar(25),
    constraint di_pk primary key (discipline)
);
create table LesEquipes (nEquipe number(4,0), nSportif number(4,0),
    constraint eq_pk primary key (nEquipe, nSportif),
    constraint eq_fk foreign key (nSportif) references LesSportifs (nSportif),
    constraint eq_ck1 check (nEquipe > 0)
);
create table LesInscriptions (nInscrit number(4,0), nEpreuve number(4,0),
    constraint i_pk primary key (nInscrit, nepreuve),
    constraint i_fk foreign key (nEpreuve) references LesEpreuves(nEpreuve),
    constraint i_ck1 check (nInscrit > 0)
);
create table LesDossiers_base (nDossier integer, nUtil integer, dateEmission date,
    constraint d_pk primary key (nDossier)
);
create table LesBillets (nBillet integer, nDossier integer, nEpreuve number(3,0),
    constraint bi_pk primary key (nBillet, nDossier),
    constraint bi_fk1 foreign key (nDossier) references LesDossiers_base(nDossier),
    constraint bi_fk2 foreign key (nEpreuve) references LesEpreuves(nEpreuve)
);
```

B Transferer-action.php

```
<?php
   session_start();
    require_once ("utils.php") ;
   $login = $_SESSION['login'];
   $motdepasse = $_SESSION['motdepasse'];
?>
<!DOCTYPE HTML>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
<head> <meta charset="utf-8" />
       <title>Gestion des comptes bancaires : les comptes </title> <meta name="GENERATOR">
</head>
<body>
<h1> Gestion des comptes bancaires </h1>
<h2> Transfert d'un compte vers un autre </h2>
<?php
$cible=$_POST['cible'];
$source=$_POST['source'];
$montant=$_POST['montant'];
if (!isset ($login) or !isset ($motdepasse) or !isset($cible) or
   !isset($source) or !isset($montant)) {
       echo "<BR>Problemes avec les variables de session. ";
} else {
  Connexion ($login, $motdepasse, $lien, $codeerreur) ;
  if ($lien) {
     $req1 = "update LesComptes set solde = solde - :montant where noC = :source";
     $curs1 = oci_parse ($lien, $req1);
     oci_bind_by_name ($curs1, ":source", $source);
     oci_bind_by_name ($curs1, ":montant", $montant);
     $res1 = oci_execute ($curs1,OCI_NO_AUTO_COMMIT);
     if ($res1) {
        $req2 = "update LesComptes set solde = solde + :montant where noC = :cible";
        $curs2 = oci_parse ($lien, $req2);
        oci_bind_by_name ($curs2, ":cible", $cible);
        oci_bind_by_name ($curs2, ":montant", $montant);
        $res2 = oci_execute ($curs2,OCI_NO_AUTO_COMMIT);
        if ($res2) {
           oci_commit ($lien);
           echo "<br> Transfert effectué.";
        } else {
           $e = oci_error ($curs2);
           oci_rollback($lien);
           echo "<br> Transfert refusé : ".$e['message']." (code : ".$e['code'].")";
        }
     } else {
        $e = oci_error ($curs1);
        oci_rollback ($lien);
        echo "<br> Transfert refusé : ".$e['message']." (code : ".$e['code'].")";
     }
     Deconnexion ($lien);
  } else {
    echo LeMessage ($codeerreur);
  }
}
?>
</body> </html>
```