Corrigé BDD I – 2ème Session – EISTI 2009-2010

Cours

Q1 : 1pt / 0.5 pt (2 groupes sur 3 bons)

Personne(idP, nom, prénom) : idP -> nom ; idP -> prénom

Réalise(*idP*, idF, année) : idF -> idP ; idF -> année

Joue(*idP, idF*, rôle, titre) : *idP, idF* -> rôle ; idF -> titre

Q2 : 0.5 pt (1FN) / 0.5 pt (BCNF)

⇨ Modèle en 1FN / Nouveau modèle proposé en BCNF :

Personne(idP, nom, prénom)

Film(idF, titre, année, *idP*)

Joue(*idP, idF*, rôle)

Q3 : 0.5 pt (Personne) / 0.5 pt (Musée)

1 personne visite au plus un musée (éventuellement aucun)

1 musée est visité par au moins une personne (éventuellement plusieurs)

Q4 : 1 pt $\Pi_{idCoureur,idCourse}(Courir) \div \Pi_{idCourse}(Course)$

Q5 : 1 pt ALTER

Q6 : 1 pt (4/4) / 0.5 pt (3/4) a) oui (nom) b) non c) oui (age) d) non

Q7 : 1 pt

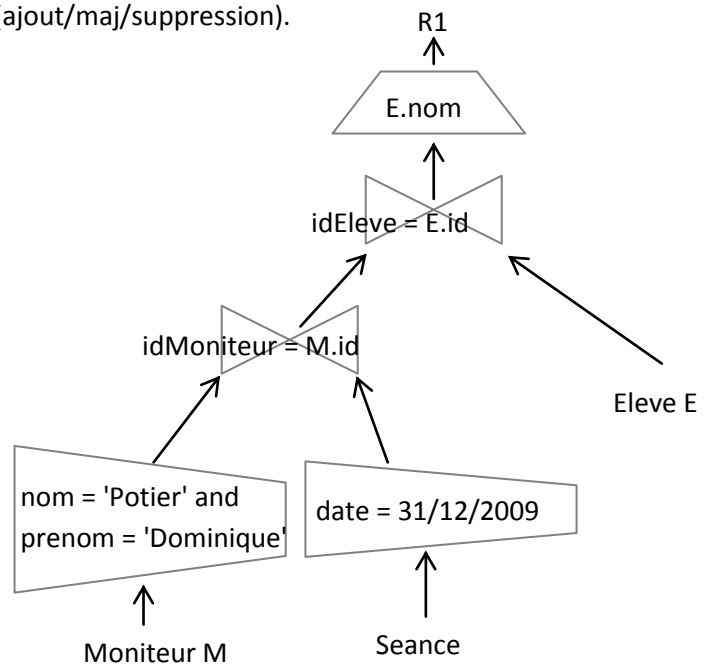
a	b
3	4
5	6
15	16

Q8 : 1 pt

La table Etudiant permet la saisie/modification/lecture de toutes les données concernat les étudiants par la secrétaire. Le professeur ne peut que consulter la liste des étudiants ordonnée par classe et groupe ; il peut ainsi connaître les étudiants qui doivent assister à ses cours sans forcément connaître les détails personnels et sans pouvoir modifier les données (ajout/maj/suppression).

Exercice

Q1. 1pt



Q2. 1 pt

```
SELECT distinct E.nom FROM (Moniteur M JOIN Seance ON M.id = idMoniteur) JOIN Eleve E ON E.id = idEleve
        WHERE M.nom = 'Potier' AND M.prenom = 'Dominique'
        AND to_char(dateSeance, 'DD/MM/YYYY') = '31/12/2009';
SELECT distinct E.nom FROM Moniteur M, Seance, Eleve E WHERE M.id = idMoniteur AND E.id = idEleve
        AND M.nom = 'Potier' AND M.prenom = 'Dominique'
        AND to_char(dateSeance, 'DD/MM/YYYY') = '31/12/2009';
```

NB : en SQL, date renommé en dateSeance

```
SELECT nom FROM Eleve WHERE id IN (
        SELECT idEleve FROM Moniteur M JOIN Seance ON M.id = idMoniteur
        WHERE M.nom = 'Potier' and M.prenom = 'Dominique'
        AND to_char(dateSeance, 'DD/MM/YYYY') = '31/12/2009');
```

Q3. 1 pt

$F_{sum(tarifHeureConduite * duree)}(Seance \bowtie_{idEleve = Eleve.id} \sigma_{prenom = 'Angelique' AND E.nom = 'Pascal'}(Eleve) \bowtie_{AutoEcole.id = idAutoEcole} AutoEcole)$

Q4. 1 pt

```
SELECT sum(tarifHeureConduite * duree) FROM (Seance JOIN Eleve E ON idEleve = E.id)
JOIN AutoEcole A ON A.id = idAutoEcole WHERE  E.prenom = 'Angelique' AND E.nom = 'Pascal';
```

Q5. 1 pt

```
SELECT A.nom, A.adresse, count(M.id) FROM AutoEcole A JOIN Moniteur M WHERE A.id = idAutoEcole GROUP BY
A.id, A.nom, A.adresse HAVING count(M.id) >= 5;
```

Q6. 1 pt

$R_1 = {}_{id,nom,prenom}F_{nom,\ prenom,\ sum(duree)}\ (Seance \bowtie_{idEleve = id} Eleve))$

Sol 1 : en posant sum(NULL) = 0

$R = TRI(\rho_{(nom,prenom,total\_heure)}(R_1), total\_heure\downarrow)$

Sol 2 : introduction fonction de remplacement de valeur NULL nvl ou isnull : nvl(x,v) = v if x IS NULL else x

$R = TRI(\rho_{(nom,prenom,total\_heure)}(\Pi_{nom,prenom,nvl(total\_heure\_non\_null,0)}(\rho_{(nom,prenom,total\_heure\_non\_null)}(R1))), total\_heure\downarrow)$

Sol 3 :

$R_2 = P_{id,nom,prenom,duree}(Seance \bowtie_{idEleve = id} Eleve)$

$R_3 = \rho_{(id,nom,prenom,duree)}(\Pi_{id,nom,prenom,0}(Eleve))$

$R = TRI(\rho_{(nom,prenom,total\_heure)}({}_{id,nom,prenom}F_{nom,\ prenom,\ sum(duree)}(R_2 U R_3)), total\_heure\downarrow)$

Q7. 1,5 pt. En SQL, sum(NULL) = NULL ! On pourra quand même accepter solution papier qui suppose sum(NULL) = 0

```
        SELECT nom, prenom, sum(duree) FROM (
                (SELECT id, nom, prenom, 0 as duree, NULL, NULL  FROM Eleve)
                UNION
                (SELECT id, nom, prenom, duree, dateSeance, heureDebut FROM Eleve JOIN Seance ON id = idEleve))
        GROUP BY id, nom, prenom ORDER BY sum(duree) DESC;
SELECT nom, prenom, NVL(total_heure_non_null, 0) as total_heure FROM (
                SELECT nom, prenom, sum(duree) as total_heure_non_null FROM Eleve LEFT JOIN Seance
                ON id = idEleve GROUP BY id, nom, prenom)
        ORDER BY total_heure DESC;
```

Q8. 1 pt

$R_1 = \rho_{(idMoniteur)}(\Pi_{Moniteur.id}(Moniteur \bowtie_{AutoEcole.id=idAutoEcol} \sigma_{nom='Rally\ Auto'}(AutoEcole)))$

$R_2 = PidEleve,idMoniteur(Seance)$

$R_3 = R_2 \div R_1$

$R = \Pi_{nom,prenom}(Eleve \bowtie_{Eleve.id=idEleve} R3)$

Q9. 1,5 pts

Sol1 : nom et prénom des élèves tels qu'il n'existe pas 1 moniteur de Rally Auto avec qui l'élève n'a pas pris de leçon(s).

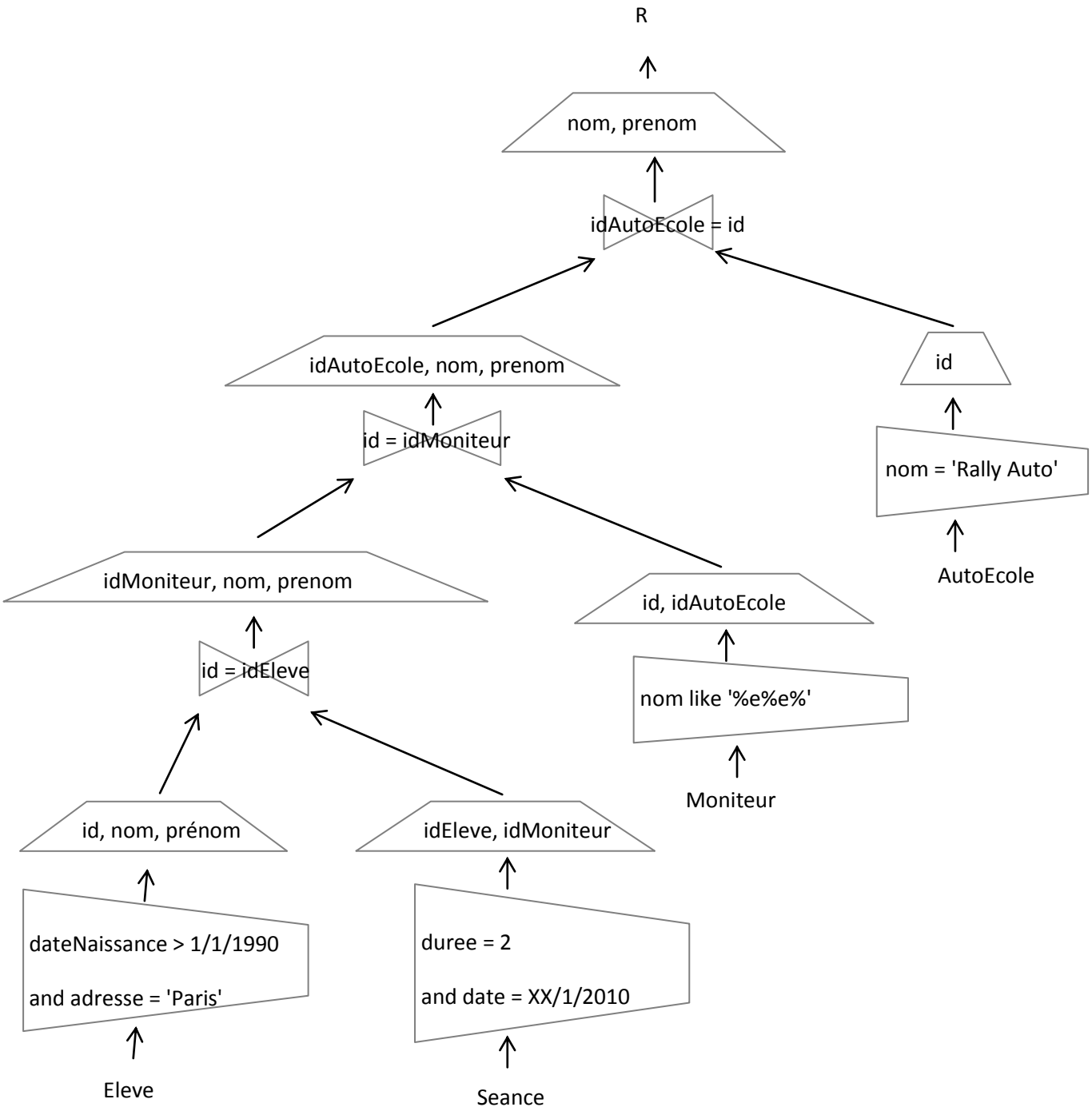
```
SELECT nom, prenom FROM Eleve eleve_tous_moniteurs WHERE NOT EXISTS (
        SELECT * FROM (Seance JOIN Moniteur moniteur_rally ON idMoniteur = moniteur_rally.id)
        JOIN AutoEcole AE ON AE.id = idAutoEcole WHERE nom = 'Rally Auto' AND NOT EXISTS (
                SELECT * FROM Seance JOIN Moniteur M ON idMoniteur = M.id
                WHERE M.id = moniteur_rally.id AND eleve_tous_moniteurs.id = idEleve));
```

Sol 2 : nom et prénom des élèves qui ont pris des leçons à l'auto-école avec un nombre de moniteurs différents égal au nombre de moniteurs de cette auto-école

```
SELECT E.nom, E.prenom FROM Eleve E JOIN Seance ON E.id = idEleve JOIN Moniteur M ON M.id = idMoniteur
        JOIN AutoEcole A ON M.idAutoEcole = A.id WHERE A.nom = 'Rally Auto'
```

```
GROUP BY E.id, E.nom, E.prenom HAVING COUNT(DISTINCT M.id) =
        (SELECT COUNT(M.id) FROM Moniteur M JOIN AutoEcole A ON idAutoEcole = A.id
                WHERE A.nom = 'Rally Auto');
```

Q10. 2 pts



Attention, AutoEcole peut être joint avec la table Moniteur ou la table Eleve. Dans tous les cas, il faut absolument faire la jointure sur le critère Eleve.id = Seance.idEleve pour être sûr que l'élève a suivi une séance sélectionnée

Notation :

- Q1 à9 : pénalité de 0.5 si opérateurs OK et dans le bon ordre mais détail erroné (distinct, ambigüité champ, critère de jointure non précisé)
- Q10 : 1pt pour restrictions descendus en bas + regroupées suivies de projections / 1pt pour jointures suivies de projections