Ing1 - Examen de Base de Données – EISTI 2008-2009

CORRIGE

Mardi 20 janvier 2009

1ère Partie – QCM – 8,5 points

Répondre aux questions suivantes. Chacune rapporte 0,5 point si elle est entièrement juste.

Q1 – Que signifie SGBD ? système de gestion de base de données

Q2 – Qu'est-ce un SGBD permet de faire ? Cochez les cases correspondantes :

□ conception d'un MCD
√□ manipulation de données
□ élaboration d'un dictionnaire de données
√□ intégrité des données
□ recueil des besoins client
√□ sécurité des données
√□ concurrence d'accès
□ passage d'un MCD à un MLD
√□ résistance aux pannes
√□ définition de données
□ tracé de graphiques

Q3 – Une base de données relationnelle est basée sur :

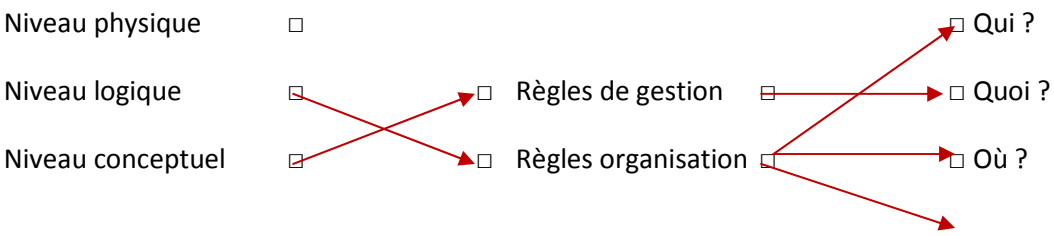
□ les espaces hilbertiens

√□ l'algèbre relationnelle

□ la méthode MERISE

□ la théorie de la relativité

Q4 – Mettre en correspondance les termes des colonnes suivantes :

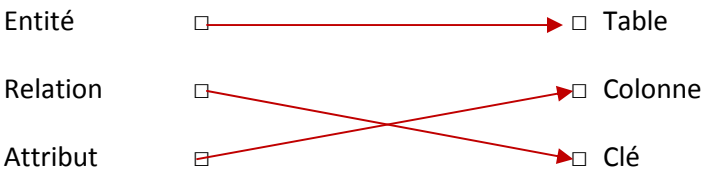


□ Quand ?

Q5 – Une clé qui identifie un enregistrement est une clé | PRIMAIRE |

Une clé qui permet de lier un enregistrement d'une table à un enregistrement d'une autre table est une clé | ETRANGERE |

Q6 – Associez les concepts du modèle de données avec ceux de la structure physique :



Q7 – Quels sont les éléments du LDD de SQL ?

□ SELECT

√□ CREATE

□ INSERT

√□ DROP

√□ ALTER

□ UPDATE

□ DELETE

Q8 – Quels sont les éléments du LMD de SQL ?

√□ SELECT

□ CREATE

√□ INSERT

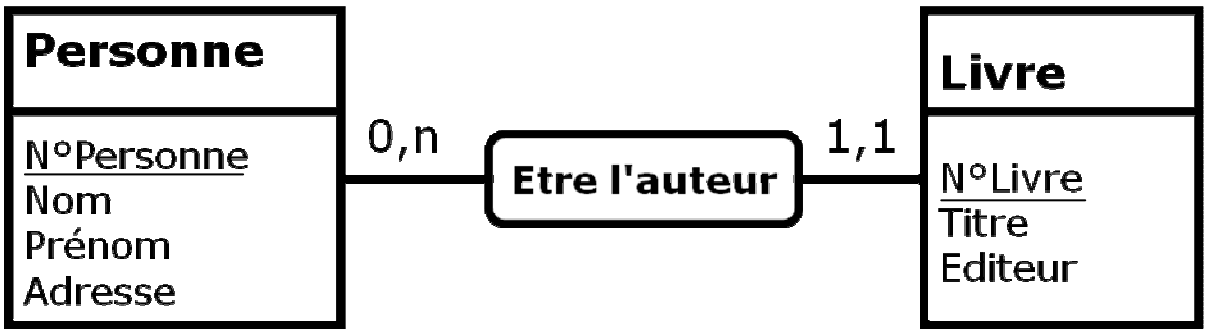
□ DROP

□ ALTER

√□ UPDATE

√□ DELETE

Q9 – Soit le MCD suivant :



Quelles sont :

- les entités : Personne, Livre
- les relations : Etre l'auteur
- les propriétés : N°Personne, Nom, Prénom, Adresse, N°Livre, Titre, Editeur
- les identifiants : N°Personne, N°Livre

Q10 – Dans le MCD précédent, interpréter les différentes cardinalités :

0,n :

1 Personne peut ne pas être l'auteur d'un Livre ;

1 Personne peut être l'auteur de plusieurs livres

1,1 :

1 Livre a pour auteur 1 seule Personne (au minimum et au maximum).

Q11 – Soit la relation suivante :

CRU	TYPE	CLIENT	REMISE
CHENAS	A	C1	3%
MEDOC	A	C2	5%
JULIENAS	B	C1	4%
CHENAS	A	C3	6%

Les dépendances fonctionnelles suivantes sont-elles vraies ou fausses ? Si oui, sont-elles élémentaires ?

a) CRU -> TYPE √□ vrai □ faux √□ élémentaire

b) CLIENT -> REMISE □ vrai √□ faux □ élémentaire
c) CRU -> REMISE □ vrai √□ faux □ élémentaire
d) CRU, CLIENT -> REMISE √□ vrai □ faux √□ élémentaire

Q12 – En quelle forme normale sont les relations suivantes ?

R1 = Commande(id_client, id_produit, quantité, nom_produit, prix_unitaire)

R2 = Commande(num_commande, id_client, id_produit, quantité, nom_produit, prix_unitaire)

R3 = Commande(num_commande,id_client,id_produits)

R4 = Commande(id_client,id_produit,date,num_commande,quantité) en considérant qu'un client ne peut passer qu'une commande par jour.

Relations	1FN	2FN	3FN	BCNF
R1	Oui	Non	non	Non
R2	Oui	Oui	non	Non
R3	non	Non	non	Non
R4	Oui	Oui	oui	Non

Q13 – Je veux effacer uniquement les données de la table Personne. Quelle requête utiliser ?

□ DROP TABLE Personne ;

√□ DELETE FROM Personne ;

Q14 – Soit la table Film suivante :

Num_film	Titre	Genre	Année
5	Dogville	Drame	2002
4	Breaking the waves	Drame	1996
3	Pulp Fiction	Policier	1994
2	Faux-Semblants	Epouvante	1988
1	Crash	Drame	1996
6	Alamo	Western	1960
7	Dangereusement vôtre	Espionnage	1985

Soient les vues suivantes :

a) CREATE VIEW Film_90_1 AS SELECT * FROM Film WHERE année BETWEEN 1990 AND 1999;
b) CREATE VIEW Film_90_2 AS SELECT * FROM Film WHERE année BETWEEN 1990 AND 1999 WITH CHECK OPTION CONSTRAINT const_film_90;

Les mises-à-jour suivantes modifient-elles la table sous-jacente Film ?

UPDATE Film_90_1 SET année = 2012 WHERE num_film = 3; √□ oui □ non

UPDATE Film_90_1 SET année = 1992 WHERE num_film = 7; □oui √□ non

UPDATE Film_90_2 SET année = 2012 WHERE num_film = 1; □ oui √□ non

UPDATE Film_90_2 SET année = 1992 WHERE num_film = 6; □ oui √□ non

Q15 – Les deux expressions algébriques suivantes donnent-elles le même résultat ?

a) $\sigma_{a=b}$ (T1 X T2)

b) T1 $\bowtie_{a=b}$ T2

Réponse : √□ oui □ non

Q16 – Dans les requêtes suivantes, un index pourrait-il être utilisé ?

a) SELECT * FROM Commande; □ oui √□ non

b) SELECT * FROM Commande WHERE id_commande > 1000; √□ oui □ non

c) SELECT * FROM Commande where la_date IS NULL; □ oui √□ non

d) SELECT * FROM LigneCommande lc JOIN Produit p ON lc.id_produit = p.id

WHERE prix_unitaire * quantité > 1000; (pour la restriction) □ oui √□ non

e) SELECT * FROM Produit WHERE nom = 'Clé USB'; √□ oui □ non

f) SELECT * FROM Client WHERE nom like 'M%'; √□ oui □ non

g) SELECT * FROM Produit WHERE prix_unitaire BETWEEN 100 and 1000; √□ oui □ non

Q17 –Soit la table initialement vide suivante :

CREATE TABLE T (a : INTEGER ,b : INTEGER) ;

On a la séquence d'ordres suivante qui proviennent de 2 sessions S1 et S2 :

S1 : INSERT INTO T VALUES (1,2) ;

S1 : INSERT INTO T VALUES (2,4) ;

S1 : COMMIT ;

S1 : UPDATE T SET b = 7 WHERE a = 1;

S2 : UPDATE T SET b = 4 WHERE a = 1;

S1 : INSERT INTO T VALUES (3,8) ;

S1 : DELETE FROM T WHERE b = 4 ;

S1 : ROLLBACK ;

S2 : COMMIT ;

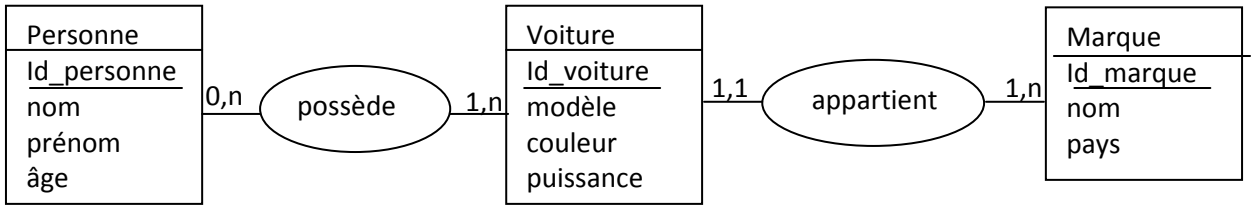
Quel est le contenu de la table T ?

a	b
1	**4**
2	**4**

2èmes Partie – Exercices – 11,5 points

Exercice 1 – 2,5 points

Donner le MLD correspondant au MCD suivant :



Personne(Id_Personne, nom, prénom, âge)

Voiture(Id_voiture, modèle, couleur, puissance,*#id_marque*)

Marque(Id_marque, nom, pays)

Possède(*#Id_personne, #Id_voiture*)

0,5 points par Entité et par relation traduite

Exercice 2

Une base de données relationnelle sur les activités de différents orchestres a été créée, le schéma de la base est donné par les relations ci-dessous :

ORCHESTRE(orch_id, nom_orch, date_creat, nb_mus, *chef_id*)

MUSICIEN(mus_id, nom_mus, prenom_mus, tel, adresse, *inst_id*)

INSTRUMENT(inst_id, nom_inst, categorie)

REPRESENTATION(rep_id, lieu, ville, date, *orch_id*)

FAIT_PARTIE(*orch_id, mus_id*)

La table ORCHESTRE contient le code (orch_id), le nom (nom_orch), la date de création (date_creat), le nombre de musiciens et le code du musicien qui dirige l'orchestre répertorié.

La table MUSICIEN contient le code (mus_id), le nom (nom_mus), le prénom (prenom_mus), le téléphone (tel), l'adresse de chaque musicien ainsi que le code de l'instrument.

La table INSTRUMENT contient le code (inst_id), le nom de l'instrument (nom_inst), la catégorie de l'instrument (categorie).

La table REPRESENTATION donne la liste des représentations données par les orchestres, pour chacune d'elles on enregistre un code (rep_id), le nom de la salle (lieu), le nom de la ville (ville), la date (date) ainsi que le code de l'orchestre qui a joué.

La table FAIT_PARTIE indique quels sont les musiciens qui font partie d'un orchestre.

Question 2.1 – 3 points

Pour chaque question, proposer une requête en algèbre relationnelle, l'arbre algébrique correspondant et une requête équivalente en SQL.

a) Quel est le nom des musiciens jouant d'un instrument à vent ?

Algèbre relationnelle :

$$\Pi_{nom\_mus}(MUSICIEN \bowtie (\sigma_{categorie = 'Instrument\ à\ vent'}(INSTRUMENT)))$$

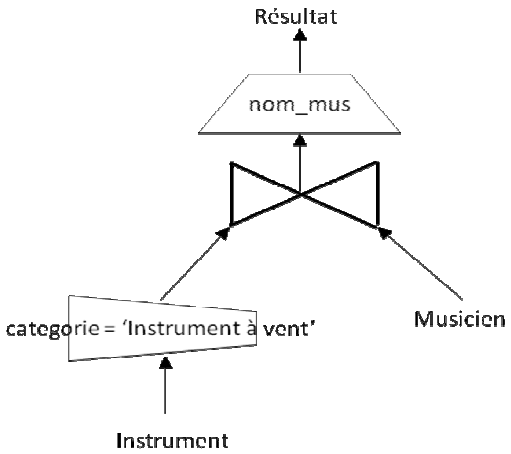
ou :

R1 = RESTRICT(INSTRUMENT, categorie = 'Instrument à vent')

R2 = JOIN(R1, MUSICIEN)

R3 = PROJECT(R2,nom_mus)

Arbre algébrique :



Requête SQL :

```
SELECT nom_mus
FROM MUSICIEN NATURAL JOIN INSTRUMENT
WHERE categorie = 'Instrument à vent';
```

ou :

```
SELECT nom_mus
FROM MUSICIEN M, INSTRUMENT I
WHERE categorie = 'Instrument à vent'
AND M.inst_id = I.inst_id ;
```

ou :

```
SELECT nom_mus
FROM MUSICIEN M
WHERE inst_id     IN     (SELECT inst_id
                          FROM INSTRUMENT
                          WHERE categorie = 'Instrument à vent') ;
```

b) Donner le nom des musiciens dirigeant au moins un orchestre de plus de 10 musiciens.

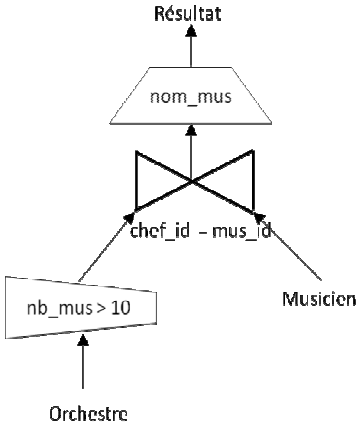
Algèbre relationnelle :

$$\Pi_{nom\_mus}(MUSICIEN \bowtie_{chef\_id=mus\_id} (\sigma_{nb\_mus>10}(ORCHESTRE)))$$

ou :

```
R1 = RESTRICT(ORCHESTRE, nb_mus > 10)
R2 = JOIN(R1, MUSICIEN, chef_id = mus_id)
R3 = PROJECT(R2,nom_mus)
```

Arbre algébrique :



Requête SQL :

```
SELECT DISTINCT nom_mus
FROM ORCHESTRE JOIN MUSICIEN ON chef_id = mus_id
WHERE nb_mus > 10;
```

Ou :

```
SELECT DISTINCT nom_mus
FROM ORCHESTRE, MUSICIEN
WHERE nb_mus > 10
AND chef_id = mus_id ;
```

c) Donner le nombre de représentation par ville dont au moins une a eu lieu après le 1er janvier 2000 ?

Algèbre relationnelle :

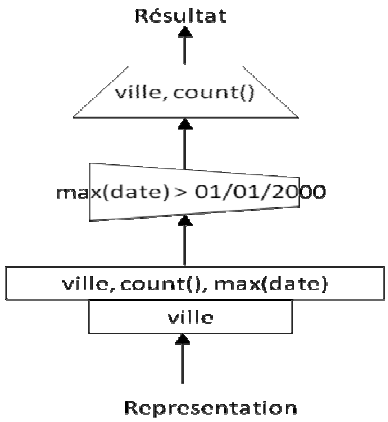
$$\Pi_{ville,nb}(\sigma_{max\_date>01/01/2000}(\rho_{(ville,nb,max\_date)}({}_{ville}F_{ville,count(),max(date)}(REPRESENTATION))))$$

Note : ρ = renommage de variables

ou plus simple :

$$\Pi_{ville,count()}(\sigma_{max(date)>01/01/2000}({}_{ville}F_{ville,count(),max(date)}(REPRESENTATION)))$$

Arbre algébrique :



Requête SQL :

SELECT ville, count(*)

FROM REPRESENTATION

GROUP BY ville

HAVING MAX(date) > TO_DATE('01/01/2000', 'DD/MM/YYYY');

Question 2.2 – 4 points

Pour chaque question, proposer une requête en algèbre relationnelle, l'arbre algébrique correspondant et l'équivalent en SQL.

a) Afficher le nom des musiciens et le nombre d'orchestres dont ils font partie (éventuellement nul).

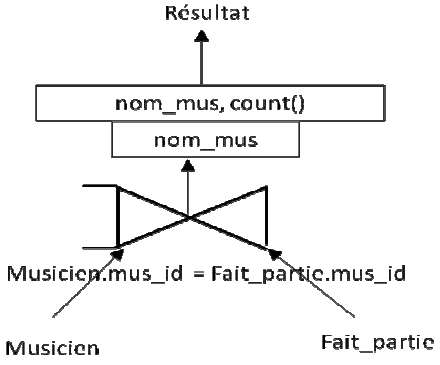
Algèbre relationnelle :

$$\Pi_{nom\_mus,nb}(\rho_{(id,nom\_mus,nb)}({}_{MUSICIEN.mus\_id}F_{MUSICIEN.mus\_id,nom\_mus,count()}(MUSICIEN \; |X|_{[MUSICIEN.mus\_id=FAIT\_PARTIE.mus\_id]} \; FAIT\_PARTIE)))$$

ou plus simple (sans renommage, group by nom_mus (pas une clé primaire)) :

$${}_{nom\_mus}F_{nom\_mus,count()}(MUSICIEN \; |X|_{[MUSICIEN.mus\_id=FAIT\_PARTIE.mus\_id]} \; FAIT\_PARTIE)$$

Arbre algébrique :



Requête SQL :

```
SELECT nom_mus, count(*)

FROM MUSICIEN m LEFT JOIN FAIT_PARTIE fp ON m.mus_id = fp.mus_id

GROUP BY m.mus_id, nom_mus
```

b) Afin de détecter les informations manquantes ou erronées, on souhaite trouver les noms des orchestres dont le nombre de musiciens est incohérent par rapport à aux musiciens enregistrés pour cet orchestre (musiciens non entrés dans la base ou information nombre de musicien non mise à jour lors de l'arrivée de nouveaux musiciens dans un orchestre).

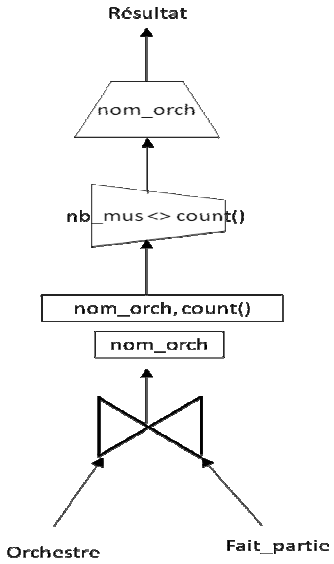
Algèbre relationnelle :

$$\Pi_{nom\_orch}(\sigma_{nb\_mus<>nb}(\rho_{(nom\_orch,nb\_mus,nb)}({}_{orch\_id,nb\_mus,nom\_orch}F_{nom\_orch,nb\_mus,count()}(ORCHESTRE \bowtie FAIT\_PARTIE)))))$$

ou plus simple (sans renommage, group by nom_orch (pas une clé primaire)) :

$$\Pi_{nom\_orch}(\sigma_{nb\_mus<>count()}({}_{nom\_orch}F_{nom\_orch,count()}(ORCHESTRE \bowtie FAIT\_PARTIE)))$$

Arbre algébrique :



Requête SQL :

```
SELECT nom_orch

FROM FAIT_PARTIE  NATURAL JOIN ORCHESTRE

GROUP BY orch_id, nb_mus, nom_orch HAVING count(*) <> nb_mus;
```

ou :

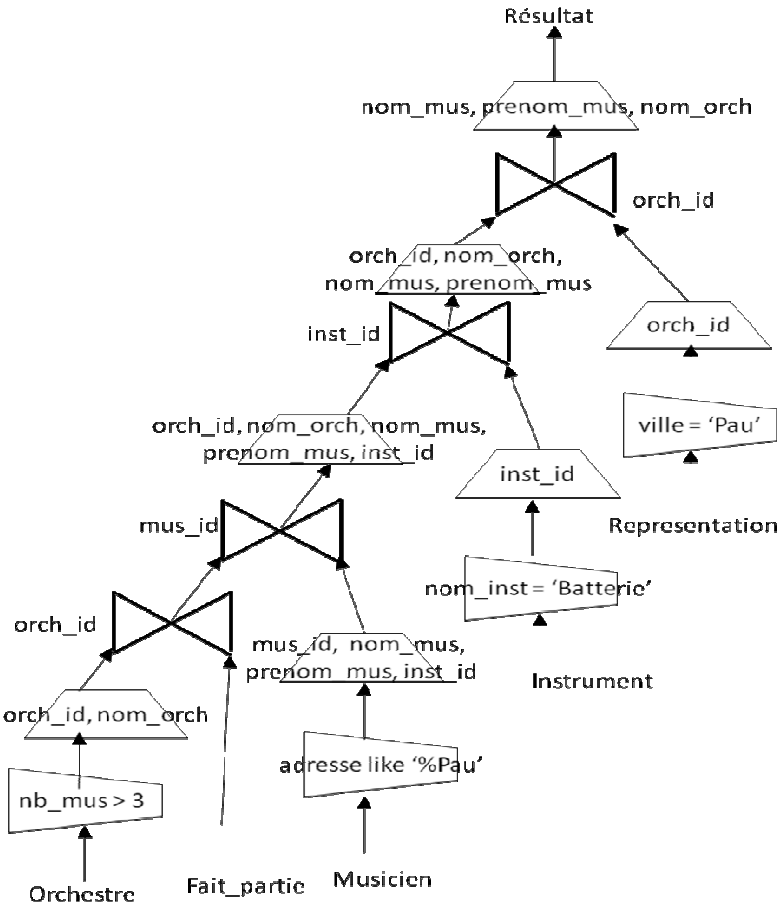
```
SELECT nom_orch
FROM FAIT_PARTIE F, ORCHESTRE O
WHERE F.orch_id = O.orch_id
GROUP BY nom_orch
HAVING count(*) <> nb_mus;
```

Question 2.3 – 2 points

Donner un arbre algébrique optimisé en suivant l'heuristique du cours pour la requête SQL suivante qui permet de trouver les joueurs de batterie habitant à Pau s'étant produit dans leur ville dans des orchestres de plus de 3 musiciens ainsi que le nom de leur groupe.

```
SELECT nom_mus, prenom_mus, nom_orch FROM
        ORCHESTRE NATURAL JOIN FAIT_PARTIE NATURAL JOIN MUSICIEN
        NATURAL JOIN INSTRUMENT NATURAL JOIN REPRESENTATION
WHERE nom_inst = 'Batterie' AND ville = 'Pau' AND adresse like '%Pau'  AND nb_mus > 3;
```

Voici l'arbre le plus optimisé possible : on effectue des restrictions d'abord et puis des projections intermédiaires qui permettent de réduire le nombre de colonnes de chaque table avant de faire des jointures.


Résultat
nom_mus, prenom_mus, nom_orch
orch_id
orch_id, nom_orch,
nom_mus, prenom_mus
inst_id
orch_id
orch_id, nom_orch, nom_mus,
prenom_mus, inst_id
ville = 'Pau'
inst_id
Representation
mus_id
nom_inst = 'Batterie'
orch_id
mus_id, nom_mus,
prenom_mus, inst_id
Instrument
orch_id, nom_orch
adresse like '%Pau'
nb_mus > 3
Orchestre
Fait_partie
Musicien