Examen juin 2015

Corrigé-type

Consignes

Livres fermés. Durée : 3 heures 1/2.
Répondez à chaque question sur une feuille séparée
sur laquelle figure vos nom, prénom, et section.

Les étudiants en géographie ne doivent pas répondre aux questions 2b, 4b et 5.

Soyez clairs, concis, et précis.

3h30 → pas de temps à perdre.

Pondération des 7 questions (juin uniquement):

Année	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Total
2017	20	20	20	10	15	15	10	110
2016	20	20	20	10	15	15	10	110
2015	20	20	20	15	10	10	10	105
2014	20	20	20	15	10	10	10	105
2013	20	20	20	15	15	10	10	110

Consignes

Livres fermés. Durée : 3 heures 1/2.
Répondez à chaque question sur une feuille séparée
sur laquelle figure vos nom, prénom, et section.

Les étudiants en géographie ne doivent pas répondre aux questions 2b, 4b et 5.

Soyez clairs, concis, et précis.

- **clairs** : utilisez le bon vocabulaire, pas d'ambiguïté, donnez des exemples si vous n'êtes pas sûrs (avec modération, voir point suivant).
- **concis** : le moins de phrases possibles, tout en restant clairs.
  - Avantage : moins de temps
- **précis** : Répondez à la question et uniquement à la question
  - Exemple :
    - *Q : Soit une relation R(A,B,C,D) avec F={A →B, C→A, BD→D}. La décomposition ρ(AB,ACD) est-elle sans perte?*
    - R : Oui
    - *Q : Soit une relation R(A,B,C,D) avec F={A →B, C→A, BD→D}. La décomposition ρ(AB,ACD) est-elle sans perte? Justifiez.*
    - R. Oui, car une décomposition en R1 et R2 est sans perte si R1 ∩ R2 → R1 \ R2 ou si R1 ∩ R2 → R2 \ R1. Or ici, on a bien A → B avec R1 ∩ R2 = A et R1 \ R2 = B.

Q1

Une société spécialisée dans la vente de vêtements au détail souhaite informatiser sa gestion. La base de données présente les propriétés suivantes :

— Un magasin est identifié par un code et décrit par son adresse (composée du code postal, de la localité, de la rue et du numéro dans la rue).

— Un patron (modèle qui sert à produire un vêtement) est identifié par un numéro unique et décrit par un label et un type (jupe, pantalon, veste, ...).

— Un vêtement est identifié par un numéro unique pour un patron donné et est décrit par sa taille (S,M,L,...), ainsi que le prix de vente minimum défini par la direction. Pour chaque vêtement, on souhaite savoir dans quel magasin il se trouve ou a été vendu.

— Les clients et les vendeuses sont tous deux des personnes qui sont identifiées par leur adresse e-mail et décrites par un nom et un prénom. Une vendeuse est, en outre, décrite par un numéro de registre national unique et par sa date de naissance. On souhaite également conserver son âge. Un client est également décrit par un numéro de carte de fidélité, s'il en possède une. On utilisera un client factice pour représenter dans la base de données un client ne souhaitant pas communiquer ses informations personnelles.

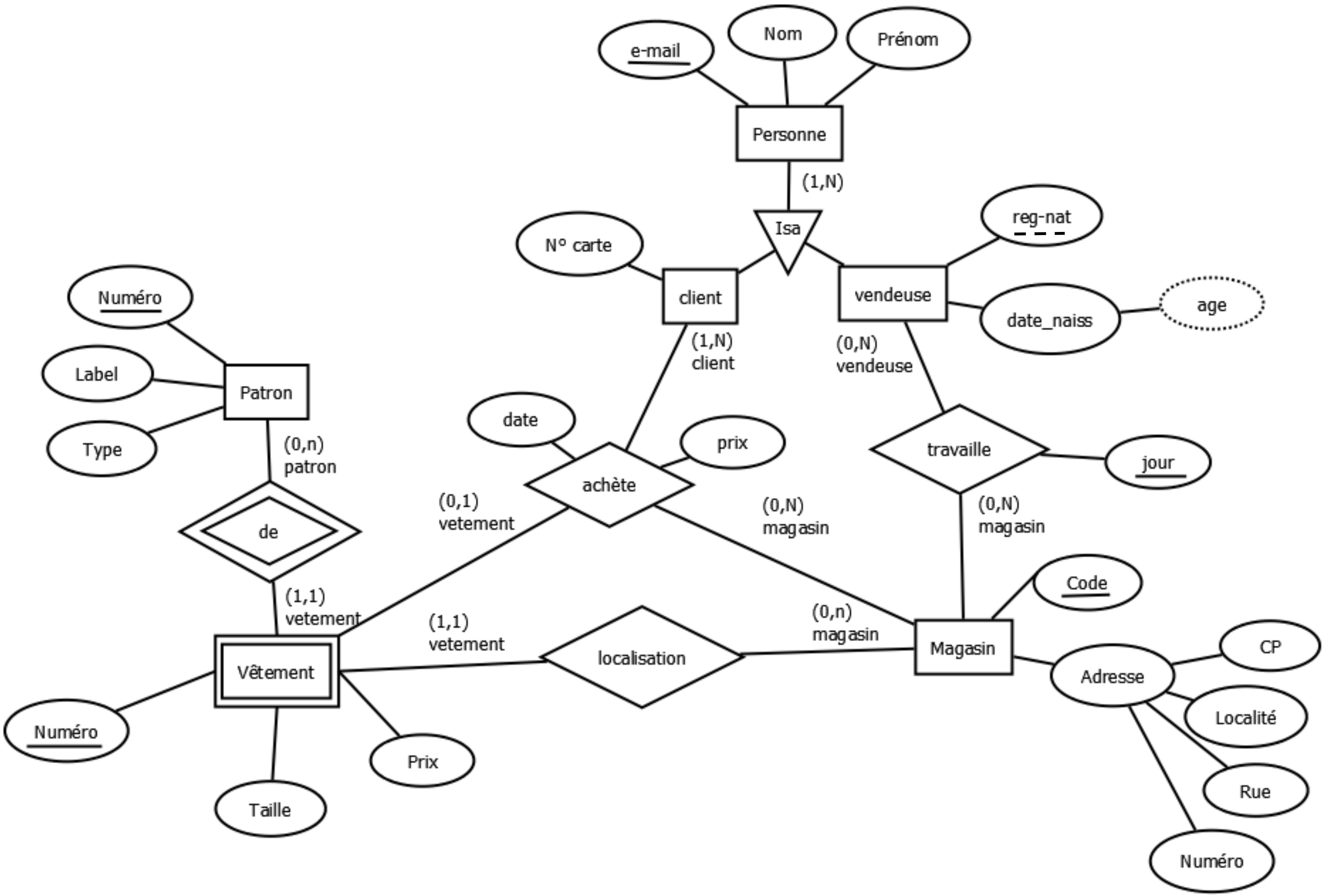
— Un client achète un vêtement dans un magasin. Pour chaque achat, on conserve la date de l'achat ainsi que le prix auquel le vêtement a été vendu.

— Une vendeuse travaille dans un seul magasin à un jour donné, mais peut travailler dans des magasins différents des jours différents. Plusieurs vendeuses peuvent travailler dans le même magasin un jour donné.

(a) Dessinez un diagramme entités-relations conforme à la description ci-dessus. Précisez les clés des ensembles d'entités et des relations, ainsi que les contraintes d'intégrité éventuellement non représentées dans le diagramme entités-relations.

(b) Effectuez la conversion de ce diagramme vers le modèle relationnel.

Q1a


e-mail
Nom
Prénom
Personne
(1,N)
Isa
N° carte
client
reg-nat
vendeuse
date_naiss
age
Numéro
Label
Patron
Type
(1,N)
client
(0,N)
vendeuse
date
prix
achète
travaille
jour
(0,n)
patron
de
(0,1)
vetement
(0,N)
magasin
(0,N)
magasin
Code
(1,1)
vetement
(1,1)
vetement
(0,n)
magasin
Vêtement
localisation
Magasin
CP
Adresse
Localité
Numéro
Prix
Rue
Taille
Numéro

Q1a (suite)

Clés des entités:

- Patron : Numéro
- Personne : e-mail
- Client : e-mail
- Vendeuse : e-mail
- Magasin : code
- Vêtement : Numéro + rôle patron de vêtement

Clés des relations :

- De : Rôle vêtement
- Localisation : Rôle vêtement
- Achète : Rôle vêtement
- Travaille : jour + rôle vendeuse

Contraintes d'intégrité :

- Email valide, taille = [XS, S, M, L, XL, XXL], prix > 0, age > 15 ans, …

Q1b

- Patron (Numéro, Label, type)
- Vêtement(Num_vet, #Numéro, Taille, Prix, #e-mail, #Code, Prix_vente,date)
- Personne(e-mail, nom, prénom)
- Client(e-mail,n°carte)
- Vendeuse(e-mail,reg-nat, date_naiss)
- Magasin(Code, Rue, Numéro, CP, localité)
- Travaille(#e-mail, jour, #Code)

Q2

2. *Théorie des dépendances.*

   (a) Soit $R(A, B, C, D, E)$ un schéma de relation et $F = \{A \rightarrow E, BC \rightarrow A, D \rightarrow B\}$ un ensemble de dépendances fonctionnelles associé à R.

      i. Donnez la (les) clé(s) de R. Justifiez.

      ii. Ce schéma est-il en BCNF ? Justifiez.

      iii. Donnez l'algorithme de décomposition en BCNF et, si nécessaire,appliquez-le à R. Cette décomposition est-elle sans perte ? Conserve-t-elle les dépendances ?

Q2ai

- C et D n'apparaissent jamais dans la partie de droite des dépendances
- Calculons CD+:
  - Avec D → B, j'ai BCD
  - Avec BC → A, j'ai ABCD
  - Avec A → E, j'ai ABCDE = R
- J'obtiens donc R à partir de CD, qui est une clé.
- Comme j'ai besoin de CD dans chacune de mes (super)clés, CD est la seule clé.

Q2aii

- Ce shéma n'est pas en BCNF, car j'ai, par exemple, A → E, et A n'est pas une clé.

Q2aiii

Algorithme de décomposition en BCNF

Données : Soit un schéma R et un ensemble de dépendances fonctionnelles F.

L'algorithme procède par décompositions successives pour obtenir une décomposition de R sans perte par rapport à F, mais sans garantie de conservation des dépendances.

- Si R n'est pas en BCNF, soit une dépendance non triviale $X \to A$ de F^+, où X n'est pas une super-clé.
- On décompose R en $R_1 = R - A$ et $R_2 = XA$ (sans perte vu le critère : $R_1 \cap R_2 = X$ et $R_2 - R_1 = A$).
- On applique l'algorithme à :

  $R_1,\ \pi_{R_1}(F) \qquad R_2,\ \pi_{R_2}(F)$

Puisque les relations deviennent de plus en plus petites, la décomposition doit s'arrêter.

Appliquons l'algorithme.
F = {A→E,BC→A,D→B}
F+ = {A→E,BC→AE,D→B, CD→R}

1. A→E pose problème, je décompose en
   1. R1 = (A,E) avec {A→E} OK
   2. R2 = (A,B,C,D) avec {BC → A, D → B, CD → AB}
2. BC → A pose problème, je décompose en
   1. R21 = (A,B,C) avec {BC → A} OK
   2. R22 = (B,C,D) avec {D → B, CD →B}
3. D → B pose problème, je décompose en
   1. R221 = (B,D) avec {D → B} OK
   2. R222 = (C,D) avec {CD → CD} OK

Solution : (A,E), (A,B,C), (B,D) et (C,D)
Décomposition sans perte : Oui, car application de l'algorithme

Conserve les dépendances :

- A+ = AE OK
- BC+ = BCA OK
- D+ = DB OK

Oui, la décomposition conserve les dépendances.

Q2

2. *Théorie des dépendances.*

   (b) Soit un schéma de relation $R(A, B, C, D)$.

      i. Expliquez, en utilisant la définition d'une DVM, pourquoi $A \twoheadrightarrow BCD$ est triviale.

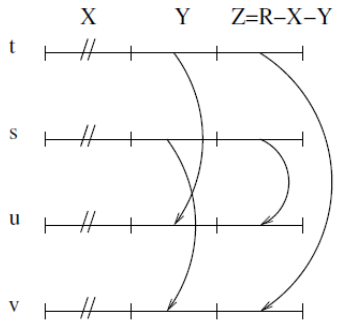
      ii. Donnez un exemple de relation r, de schéma R, qui satisfasse $d = \{C \rightarrow A, B \twoheadrightarrow CD\}$ mais pas $B \rightarrow A$ ou (dé)montrez que c'est impossible.

Q2bi

Les Dépendances à valeurs multiples : définition formelle

Une relation r de schéma R satisfait une *dépendance à valeurs multiples (DVM)* $X \twoheadrightarrow Y$ si pour toute paire de tuples $t, s \in r$ tels que $t[X] = s[X]$, il existe deux tuples $u, v \in r$ tels que :

1. $u[X] = v[X] = t[X] = s[X]$
2. $u[Y] = t[Y]$ et $u[R–X–Y] = s[R–X–Y]$
3. $v[Y] = s[Y]$ et $v[R–X–Y] = t[R–X–Y]$



- Dans le cas de A $\twoheadrightarrow$ BCD, c'est trivial, car Z ne contient aucun attribut.
- De ce fait, t = u et s = v et la définition est respectée.

Q2bii

- Essayons de trouver un exemple
  - 1 : Rendons faux $B \rightarrow A$
  - 2 : Rendons vrai $C \rightarrow A$
  - 3 : Rendons vrai $B \twoheadrightarrow CD$
  - Mais maintenant, $C \rightarrow A$ n'est plus vrai

A	B	C	D
A1	B1	C1	D1
A2	B1	C2	D1
A1	B1	C2	D1
A2	B1	C1	D1

- Essai 2
  - 1 : Rendons vrai $B \twoheadrightarrow CD$ et $C \rightarrow A$
  - Mais $B \rightarrow A$ est vrai, rendons-le faux
  - Alors $C \rightarrow A$ n'est plus vrai

A	B	C	D
A1	B1	C1	D1
A2	B1	C1	D1

- C'est impossible

Q2bii (suite)

On peut aussi remarquer que si B $\twoheadrightarrow$CD, alors, par complémentation,

B $\twoheadrightarrow$ A.

On nous demande de rendre B $\twoheadrightarrow$A et B $\rightarrow$ A faux.

On peut seulement y arriver à partir de (au moins) deux tuples, avec des valeurs identiques pour B et différentes pour A.

Si les deux tuples ont des valeurs identiques pour C, alors on n'a pas C$\rightarrow$A.

Si les deux tuples ont des valeurs différentes, je dois rajouter deux lignes pour croiser les valeurs de A et, je n'ai pas non plus C$\rightarrow$A

C'est donc bien impossible.

Q3

3. *Langages d'interrogation.*

Un club d'échecs dispose d'une base de données incluant les relations dont le schéma est donné ci-dessous.

— *membre*(<u>code</u>, nom, prénom, adresse) : relation reprenant les informations concernant les différents membres ;

— *partie*(<u>numéro</u>, #code1, #code2, date, vainqueur) : relation reprenant les différentes parties jouées entre deux membres. Vainqueur vaut soit 1, soit 2, soit 0 en cas de match nul. Le joueur représenté par *code*1 joue avec les pièces blanches ;

— *historique*(<u>numéro</u>, <u>ordre</u>, #id_mvt) : relation reprenant l'historique des parties, à savoir la liste des mouvements effectués lors de la partie, et l'ordre dans lequel ces mouvements ont été effectués (le premier mouvement est numéroté «1», les blancs jouent toujours en premier) ;

— *mouvement*(<u>id_mvt</u>, couleur, type, pos_dep, pos_arr) : relation reprenant les mouvements effectués, à savoir la couleur et le type (pion, fou, tour, ...) de pièce jouée, ainsi que sa case de départ et celle d'arrivée (représentées par une lettre et un chiffre).

Exprimez les requêtes suivantes en algèbre relationnelle étendue et en SQL :

(a) Rechercher le nom et le prénom des personnes ayant gagné au moins une partie en jouant avec les pièces noires.

(b) Rechercher, pour chaque membre, le nombre de parties qu'il a gagnées, et triez les résultats en fonction de ce nombre, par ordre décroissant.

(c) Rechercher le nom et le prénom des membres qui commencent toutes leurs parties en déplaçant un cavalier (qu'ils jouent les blancs ou les noirs).

Q3a

$$\Delta(\Pi_{nom,prénom}(\sigma_{vainqueur=2}(\delta_{code \leftarrow code2}(partie)) \bowtie membre))$$

```
SELECT DISTINCT nom, prénom
FROM membre
WHERE code IN
        (SELECT code2 AS code
        FROM partie
        WHERE vainqueur = 2)
;
```

Q3b

τ_{somme} $((\gamma_{code,\ somme\ \leftarrow\ COUNT(numéro)}$ $(\Pi_{numéro,\ code}$ $(\delta_{code\ \leftarrow\ code2}$ $(\sigma_{vainqueur=2}$ (partie))) $\cup$ $\Pi_{numéro,\ code}$ ($\delta_{code\ \leftarrow\ code1}$ $(\sigma_{vainqueur=1}$ (partie)))))

$\cup$

$(\Pi_{\ code,somme\ \leftarrow\ 0}$ $(\Pi_{\ code}$(membre) -

Π_{code} $(\gamma_{code,\ somme\ \leftarrow\ SUM(numéro)}$ $(\Pi_{numéro,\ code}$ $(\delta_{code\ \leftarrow\ code2}$ $(\sigma_{vainqueur=2}$ (partie))) $\cup$ $\Pi_{numéro,\ code}$ ($\delta_{code\ \leftarrow\ code1}$ $(\sigma_{vainqueur=1}$ (partie)))))))

Logique :
- Je crée une table contenant les numéros des parties et le code de la personne qui l'a gagnée (attention qu'il peut jouer les blancs ou les noirs)
- Je groupe par code, et je compte le nombre de parties.
- Certains codes pourraient ne jamais avoir joué, j'insère leur code avec la valeur 0
- Je trie le résultat final.

Q3b (suite)

```
SELECT code, somme
FROM
        (SELECT code, COUNT(numéro) AS somme
        FROM ((SELECT code2 as code, numéro
                FROM partie WHERE vainqueur = 2) UNION
                (SELECT code1 as code, numéro
                FROM partie WHERE vainqueur = 1))
        GROUP BY code
        ) UNION
        (SELECT code, 0 AS somme
        FROM membre
        WHERE code NOT IN
                (SELECT code2 as code FROM partie WHERE vainqueur = 2)
        AND code NOT IN
                (SELECT code1 as code FROM partie WHERE vainqueur = 1))
ORDER BY somme DESC;
```

Q3c

$\Pi_{nom,prénom}$
$((\Delta(\Pi_{code,type}\ (\delta_{code1 \leftarrow code}\ (\sigma_{couleur=blanc}\ (mouvement) \bowtie \sigma_{ordre=1}\ (historique) \bowtie partie))$
$\cup$
$\Pi_{code,type}\ (\delta_{code2 \leftarrow code}\ (\sigma_{couleur=noir}\ (mouvement) \bowtie \sigma_{ordre=2}\ (historique) \bowtie partie))) \div$
$\delta_{type \leftarrow 'cavalier'}\ )$
$\bowtie membre)$

Logique:

- Je crée une table avec, pour chaque partie, la première pièce que le joueur a déplacé (attention qu'il peut jouer les blancs ou les noirs)
- Je retire les doublons
  - J'ai donc, pour chaque personne, une liste de pièces qu'il a joué lors de son premier mouvement, quelles que soit les parties.
- Je divise avec une table fictive qui contient uniquement la valeur "cavalier".
  - S'il a joué autre chose que le cavalier dans au moins une partie, la division ne renverra pas son code
- Je croise avec membre pour récupérer son nom et son prénom.

Q3c (suite)

```
SELECT nom, prénom
FROM membre
WHERE NOT EXISTS (SELECT *
        FROM partie NATURAL JOIN historique NATURAL JOIN mouvement
        WHERE type NOT LIKE 'cavalier' AND ordre = 1 and membre.code = partie.code1)
AND NOT EXISTS (SELECT *
        FROM partie NATURAL JOIN historique NATURAL JOIN mouvement
        WHERE type NOT LIKE 'cavalier' AND ordre = 2 and membre.code = partie.code2);
```