

--	--	--

Introduction aux bases de données – LU2IN009

Examen du 21 Mai 2021

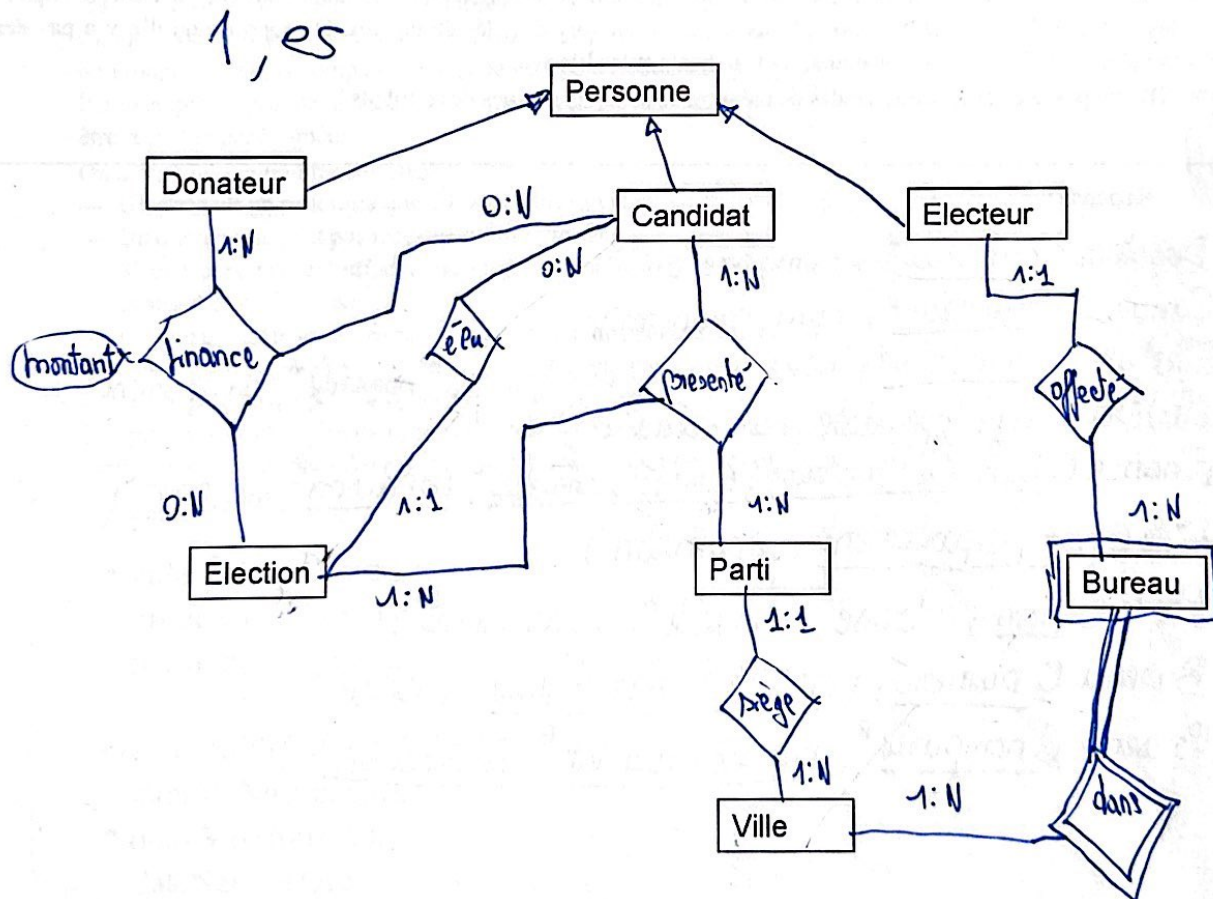
Durée : 2 heures

Documents autorisés

Les téléphones mobiles doivent être éteints et rangés dans les sacs. Le barème sur 20 points (14 questions) n'a qu'une valeur indicative.

1 Modélisation Entité-Association et Relationnel (5 pts)

On considère une base de données pour gérer les élections passées à l'aide d'un schéma entité-association dont **toutes** les entités apparaissent sur la figure suivante :



Les personnes sont identifiées par leur numéro de sécurité sociale et on connaît leurs nom et prénom. Les candidats sont présentés par un parti à une élection donnée (ils peuvent se présenter à plusieurs élections). Chaque élection est identifiée par un *id*, est d'un certain type et a lieu à une certaine date. On connaît pour chaque élection le candidat qui a été élu. Un parti politique a un nom unique et a son siège à une certaine adresse dans une ville donnée. Une ville a un nom, une population et est dans un département. Plusieurs villes peuvent avoir le même nom mais pas dans le même département. Chaque électeur est affecté à un bureau de vote. Un bureau de vote a une adresse et un numéro qui l'identifie pour une ville donnée (ex : le bureau de vote 3 de la ville Strasbourg dans le département du Bas-Rhin). Les donateurs sont des personnes qui peuvent financer pour un certain montant un candidat pour une élection. Ils peuvent financer plusieurs candidats. Un candidat peut ne pas être financé.

Question 1 (3 points)

Complétez le schéma Entité-Association précédent. Ne pas oublier de préciser les attributs et les identifiants de chaque entité. Ajoutez également les associations correspondantes entre les entités en précisant leurs cardinalités et leurs éventuels attributs. Rappel : les liens de généralisation/spécialisation (héritage) sont représentés par des flèches, les entités faibles par un double rectangle, leur association identifiante par un double losange. Attention : Vous ne pouvez pas rajouter des entités supplémentaires !

Question 2 (1 1/2 points)

Compléter le cadre suivant pour traduire le schéma Entité/Association en relationnel. Pour chaque table, soulignez les clés primaires et ajoutez une '*' aux attributs qui sont des clés étrangères. On suppose qu'il n'y a pas des personnes autres que les Donateurs, les Candidats et les Électeurs.

NB : ne pas donner les commandes de création de table et les types des attributs !

Réponse :

Donateur C numsecu*, nom, prenom)
 Candidat C numsecu*, nom, prenom)
 Electeur C numsecu*, nom, prenom, numibureau*, nomville*, departement*)
 Election C id, type, date, numsecuelu*)
 Finance C numsecuDonateur*, numsecucandidat*, idElection*, montant)
 Ville C nom, departement, population)
 Parti C nom, adresse, nomville*, departement*)
 Bureau C numero, nomville*, departement, adresse)
 Presente C nomparti*, numsecucandidat*, idElection*)

Question 3 (1/2 point)

Un donateur peut-il donner plusieurs sommes d'argent à un candidat pendant la même élection ? Entourer et justifier la bonne réponse.

Réponse : Oui ☒ Non

Justification :

Non, la clé de l'association est le triplet idCandidat, idDonateur, idElection

2 Contraintes et Triggers (7 pts)

Question 4 (3 points)

On veut créer un schéma relationnel qui permet de stocker des informations sur les donneurs et receveurs de sang, les dons et les transfusions. Les huit groupes sanguins (A+, A-, B+, B-, AB+, AB-, O+, O-) sont stockés dans une table Groupe. Les informations sur les donneurs et les receveurs sont stockées dans la table Personne. Chaque personne est identifiée par son numéro de sécurité sociale. On connaît aussi son nom, sa date de naissance et son groupe sanguin. Les donneurs et donneuses sont aussi stockés dans la table Donneur avec le nombre de dons (nbDons). Chaque don (pochette de sang) est stocké dans la table Don avec un identificateur unique (idDon), le numéro de sécurité sociale du donneur et la date du don. Chaque don utilisé dans une transfusion est enregistré dans la table Transfusion, avec le numéro de sécurité sociale du receveur et la date de transfusion (dateTrans). Chaque pochette de sang peut être transfusée à une seule personne. Enfin, la table Compatible contient les paires de groupes sanguins compatibles avec le groupe du donneur (groupeDon) et le groupe du receveur (groupeRec). Par exemple, le groupe donneur O- peut être reçu par tous les groupes, tandis que le groupe donneur AB+ ne peut être reçu que par lui-même.

On définit les contraintes suivantes :

- Un donneur ne peut faire que 70 dons dans sa vie.
- Un donneur ne peut pas recevoir son propre sang.
- Si une personne est effacée, on efface toutes les informations qui la concernent (en respectant les autres contraintes ci-dessous).
- Un don qui a été utilisé dans une transfusion, ne peut pas être effacé.
- Une personne qui a reçu une transfusion, ne peut pas être effacée.
- On ne peut pas effacer des groupes de sang référencés.

Complétez le schéma relationnel suivant en ajoutant les contraintes précédentes, les contraintes de clé et les contraintes référentielles. Vous n'êtes pas obligés de nommer les contraintes avec CONSTRAINT.

Réponse :

```
create table Groupe (
  groupe varchar(3),
  primary key (groupe)
);
create table Personne(
  numSS Number(6),
  nom Varchar(32),
  dateNaiss Date,
  groupe not null Varchar(3),
  primary key (numSS),
  foreign key (groupe) references Groupe (groupe)
```



```

);
create table Donneur(
  numSS Number(6),
  nbDons Number(2),
  primary key (numSS),
  foreign key (numSS) references Personne (numSS)
    on delete cascade
  check (nbDons <= 10)
);
create table Don(
  idDon number(12),
  numSS number(6),
  dateDon Date,
  primary key (idDon),
  foreign key (numSS) reference donneur (numSS)
    on delete cascade
);
create table Transfusion(
  idDon Number(12),
  numSS Number(6),
  dateTrans Date,
  primary key (idDon),
  foreign key (numSS) references Personne (numSS)
    on delete restrict, (par default),
  Foreign Key (idDon) references Don (idDon)
    on delete restrict, (par default),
  check ((numSS, idDon) not in (select numSS, idDon from Don))
);
create table Compatible(
  groupeDon Varchar(3),
  groupeRec Varchar(3),
  foreign key (groupeDon, groupeRec) reference
    table (groupe, groupe) ou
    Foreign Key (groupeDon) references Groupe (groupe)
    Foreign Key (groupeRec) references Groupe (groupe)
);

```

Question 5 (1 point)

Avec le schéma précédent, est-ce qu'on peut effacer une personne qui a fait des dons sanguins qui ont été utilisés dans une transfusion. Entourez la bonne réponse et justifiez.

Réponse : Oui / Non

Justification :

Si on efface la personne on effacerai aussi son don qui ne peut pas être effacé à cause de restrit. dans Transfusion

Question 6 (1/2 point)

Insérer dans la base de données les deux donneurs de sang, (1) Pierre avec le numéro de sécurité social 123456 né le 2 mars 1956 et le groupe A+ et (2) Annette, avec le numéro 890716 et née le 17 juillet 1969 et le groupe A-.

Réponse :

```
Insert into Personne Values (1, 'Pierre', parsedatetime('11-12-2017', 'dd-MM-yyyy'), 'A+');
Insert into Personne Values (2, 'Annette', parsedatetime('11-10-2019', 'dd-MM-yyyy'), 'A-');
Insert into Donneur Values (123456, 0);
Insert into Donneur Values (890716, 0);
```

Les deux donneurs ont fait des dons en 2017 et 2019 :

```
insert into Don values(6878, 123456, parsedatetime('11-12-2017', 'dd-MM-yyyy'));
```

```
insert into Don values(2818, 890716, parsedatetime('11-10-2019', 'dd-MM-yyyy'));
```

Question 7 (2 1/2 points)

On veut créer un trigger qui assure une distance minimale de 180 jours entre deux dons faits par le même donneur ou la même donneuse. Si cette distance est respectée, le nombre de dons du donneur est augmenté de 1. Sinon, le trigger déclenche une exception 'dates_top_proche'.

Par exemple, l'insertion suivante déclenche l'exception 'dates_top_proche' :

```
insert into Don values(5677, 890716, parsedatetime('19-10-2019', 'dd-MM-yyyy'));
```

Réponse :

```
create or replace trigger incDons
```

```
before insert on Don
```

```
for each row
```

```
declare
```

```
diffDays Number;
```

```
date_top_proche exception
```

```
begin
```

```
select :new.DateDon - max(DateDon) into diffDays
```



```

from Don
where num8 = new.num8;
if (diffDays is null or diffDays > 180) then
  update Donneur
  set nbDonc = nbDonc + 1
  where num8 = new.num8;
else
  raise date trop proche;
end if;
end;
/

```

3 Requêtes (8 pts)

On considère le schéma de la base suivante qui gère les consultations de patients par des médecins au sein de centres de santé. Les attributs de la clé primaire sont soulignés et ceux d'une clé étrangère portent le nom des attributs référencés et sont suivis d'un astérisque.

Medecin (idM, specialite, annees)

Patient (idP, dateNaiss, villeP, idM*)

Centre (idC, capacite, villeC, secteur)

Tarif (idM*, idC*, montant)

Consult (idP*, idM*, idC*, motif, date)

Chaque médecin intervient dans une spécialité donnée (ex. ophtalmo, chirurgie, ...) et possède une ancienneté exprimée en nombre d'années de service. Pour chaque patient, on connaît sa date de naissance, sa ville d'habitation et son médecin traitant (attribut *idM*). Les médecins consultent leurs patients dans des centres, pour chaque centre on connaît sa capacité d'accueil en nombre de box de consultations, la ville où il se situe et son secteur qui peut être soit 1 soit 2. Le tarif appliqué par un médecin dépend du centre où il consulte et est renseigné dans la table *Tarif*. La table *Consult* renseigne le motif (suivi, premier rdv, soins) et la date d'une consultation d'un patient par un médecin au sein d'un centre.

Exprimer dans le langage demandé (calcul ou SQL) les requêtes suivantes.

Question 8 (1 point)

Les identifiants des patients qui consultent leur médecin traitant dans un centre situé dans leur ville d'habitation.

Réponse : Calcul

$$\{p.idP \mid \text{Patient}(p) \wedge \exists c, t (\text{Consult}(c, t) \wedge \text{Centre}(c, t) \wedge p.idP = c.idP \wedge c.idC = t.idC \wedge p.villeP = t.villeC \wedge p.idM = c.idM)\}$$

Question 9 (1 point)

Les identifiants des médecins avec plus de 10 ans de service qui ont eu au moins une consultation et qui ne consultent que dans des centres en secteur 2.

Réponse : Calcul

$$\{ m.idM \mid Medecin(m) \wedge m.annee > 10 \wedge \exists c (Consult(c) \wedge m.idM = c.idM \wedge \neg (\exists c, t (Consult(c) \wedge Centre(t) \wedge m.idM = c.idM \wedge c.idC = t.idC \wedge t.secteur = 1))) \}$$

Question 10 (1 point)

Les identifiants des centres situés à Paris où il existe au moins une consultation pour chaque spécialité présente dans la base.

0,25

Réponse : Calcul

$$\{ t.idC \mid Centre(t) \wedge t.villeC = 'Paris' \wedge \exists sp (Medecin(sp) \wedge \neg \exists c, m (Consult(c) \wedge Medecin(m) \wedge c.idC = t.idC \wedge c.idM = m.idM \wedge m.specialite = sp.specialite)) \}$$

Question 11 (1 point)

Les identifiants des patients qui ont consulté leur médecin traitant à deux reprises à 3 mois d'intervalle.

0

Réponse : SQL

```

Select p.idP
From Patient p, consult c1, consult c2, medecin m
Where p.idP = c1.idP and c1.idP = c2.idP and c1.idM = m.idM and c2.idM = m.idM
and datediff(month, c1, c2) = 3

```

Question 12 (1 point)

Pour chaque identifiant de centre et chaque spécialité, retourner le nombre de consultations réalisées dans ce centre pour cette spécialité ainsi que le montant total de ces consultations.

0,5

Réponse : SQL

```

Select Count(c.idC), Count(m.specialite), sum(c.cons.taxifs)
From consult c, tarif t, medecin m
Where c.idM = m.idM and c.idM = m.idM and t.idC = c.idC
group by c.idC, m.specialite

```

Question 13 (1 ½ points)

①

Le motif pour lequel il y a le plus grand nombre de consultations impliquant des médecins de la spécialité 'ophtalmo'.

Réponse: SQL

```

Select c.motif
From Consult c, Medecin m
where m.idM = c.idM and m.specialite = 'ophtalmo'
group by c.motif
having count(*) >= all (select count(*) from medecin m, consult c
where m.idM = c.idM and m.specialite = 'ophtalmo'
group by c.motif);

```

Question 14 (1 ½ points)

0,25

Les identifiants des centres où tous les médecins avec plus de 10 années de service consultent au moins une fois un patient habitant 'Lyon'. La solution doit utiliser **GROUP BY**.

Réponse: SQL

```

Select c.idc
from Medecin m, consult c, patient p
where c.idM = m.idM and m.annee > 10 and p.idP = c.idP and p.ville = 'Lyon'
groupBy idc
having count(distinct c.idM) = (select count(distinct c2.idM)
from Medecin m2, consult c2
where m2.idM = c2.idM and c2.idc = c.idc and
m2.annee > 10)

```