Exercice 1 – Normalisation :

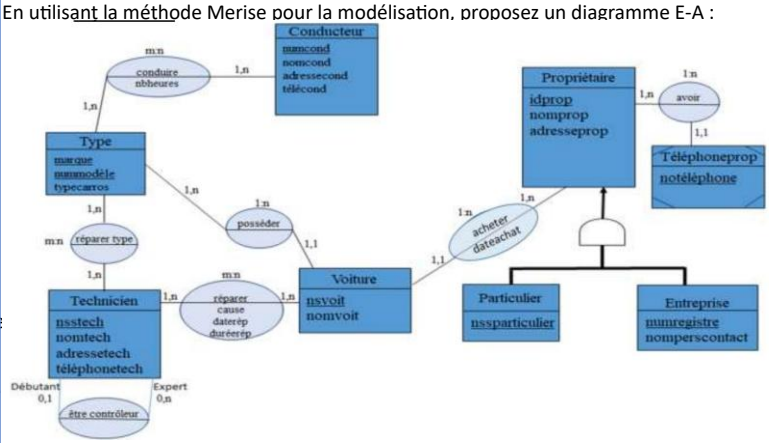
A18 : Considérons le schéma relationnel suivant pour une compagnie de vente de produits pour des clients selon des factures déterminées.
Client (idclient, nomclient, adresseclient, nofacture)
Facture (nofacture, idclient, datefacture, idproduit, quantité, prix)
Produit (idproduit, nomproduit, idtype, descriptiontype)
Type (idtype, descriptiontype)
Les contraintes logiques sont les suivantes :
• Chaque facture est relative à un seul client et doit contenir plusieurs produits. • Chaque produit doit exister une seule fois dans la facture elle-même.
• Chaque produit doit avoir un seul type.
1.1. Vérifiez chacune des relations mentionnées ci-dessus si elle est en forme normale en précisant le niveau. Justifiez votre réponse dans le cas où une fonction n'est pas en forme normale pour le niveau trouvé.
La relation « Client » n'est pas en 1FN, parce qu'il y a plusieurs valeurs de « nofacture » pour la même valeur de la clé « idclient ». La relation « Facture » n'est pas en 1FN, parce qu'il y a plusieurs valeurs de « idproduit », « quantité » et « prix » pour la même valeur de la clé « nofacture ». La relation « Produit » est en 1FN, en 2FN, mais elle n'est pas en 3FN, parce que la DF (idtype ◊ descriptiontype) n'est pas directe. La relation « Type » est en 1FN, en 2FN, en 3FN et en FNBC.
1.2. Décomposez les relations qui ne sont pas en forme normale.
La relation « Client » sera décomposée en deux relations :
R1 = Client (idclient, nomclient, adresseclient)
R2 = ClientFacture (idclient, nofacture)
La relation « Facture » sera décomposée en deux relations :
R3 = Facture (nofacture, idclient, datefacture)
R4 = FactureProduit (nofacture, idproduit, quantité, prix)
La relation « Produit » sera composée en deux relations :
R5 = Produit (idproduit, nomproduit, idtype)
R6 = Type (idtype, descriptiontype)
1.3. Quel sera le schéma relationnel après normalisation des relations.
R1 = Client (idclient, nomclient, adresseclient)
R3 = Facture (nofacture, idclient, datefacture)
R4 = FactureProduit (nofacture, idproduit, quantité, prix)
R5 = Produit (idproduit, nomproduit, idtype)
Type (idtype, descriptiontype)

A19 : Un service de documentation technique dans une entreprise voudrait créer une base de données pour répertorier tous ses documents et gérer les prêts aux employés de l'entreprise. Dans cette application, un document revient à un département et un seul. Il peut être écrit par plusieurs auteurs et décrit par plusieurs mots clés. Un employé travaille dans un seul département. Le schéma relationnel proposé pour cette base de données est le suivant :
DOCUMENT (iddoc, typedoc, titredoc, dateparution, iddept, idauteur, idmotclé)
AUTEUR (idauteur, nomauteur, prénomauteur); MOTCLE (idmotclé, motclé)
EMPLOYE (idemp, nomemp, prenomemp, fonction, adresse, tel, iddept, nomdept);
EMPRUNT (idemp, iddoc, titredoc, dateemprunt, dateretourprévue, dateretoureffective);
DEPARTEMENT (iddept, nomdept, idemp, nomemp)
1.1- Précisez la forme normale de chacune des relations. Justifiez votre réponse.
DOCUMENT n'est pas en 1FN, parce que idauteur et idmotclé sont multivalués pour la même valeur de la clé (iddoc).
AUTEUR est en 1FN, 2FN, 3FN et FNBC.
MOTCLE est en 1FN, 2FN, 3FN et FNBC.
EMPLOYE est en 1FN, en 2FN, mais pas en 3FN, parce que la DF (idemp -> nomdept) n'est pas directe. Elle est déduite par transitivité des 2 DF (idemp -> iddept et iddept -> nomdept).
EMPRUNT est en 1FN, mais pas en 2FN parce que la DF (idemp, iddoc -> titredoc) n'est pas élémentaire à cause de la DF (iddoc -> titredoc).
DEPARTEMENT n'est en 1FN, parce que idemp et nomemp sont multivalués pour la même valeur de la clé (iddept) et il existe une relation (idemp, nomemp) dans la relation DEPARTEMENT.
1.2- Pour les relations qui présentent toujours de la redondance, proposez une correction.
La relation DOCUMENT sera décomposée en 3 relations :
R1 = DOCUMENT (iddoc, typedoc, titredoc, dateparution, iddept)
R2 = DOCAUTEUR (iddoc, idauteur)
R3 =DOCMOTCLE (iddoc, idmotclé)
La relation EMPLOYE sera décomposée en 2 relations :
R4 = EMPLOYE (idemp, nomemp, prenomemp, fonction, adresse, tel, iddept)
R5 = DEPARTEMENT (iddept, nomdept) (redondante à éliminer)
La relation EMPRUNT sera décomposée en 2 relations :
R6 = EMPRUNT (idemp, iddoc, dateemprunt, dateretourprévue, dateretoureffective)
R7 = DOCUMENT (iddoc, titredoc) (redondante à éliminer)
La relation DEPARTEMENT sera décomposée en 3 relations :
R8 = DEPARTEMENT (iddept, nomdept)
R9 = DEPEMP (iddept, idemp)
R10 = EMPLOYE (idemp, nomemp) (redondante à éliminer)
1.3- Donnez le schéma relationnel final.
R1 = DOCUMENT (iddoc, typedoc, titredoc, dateparution, iddept)
R2 = DOCAUTEUR (iddoc, idauteur)
R3 =DOCMOTCLE (iddoc, idmotclé)
AUTEUR (idauteur, nomauteur, prénomauteur)
MOTCLE (idmotclé, motclé)
R4 = EMPLOYE (idemp, nomemp, prenomemp, fonction, adresse, tel, iddept)
R6 = EMPRUNT (idemp, iddoc, dateemprunt, dateretourprévue, dateretoureffective)
R8 = DEPARTEMENT (iddept, nomdept)
R9 = DEPEMP (iddept, idemp)
(puisque iddept existe au niveau de Employe, alors cette relation n'est pas obligatoire de l'avoir au niveau du résultat final et +0.25 au niveau de Employe)

Exercice 2 – Diagramme E-A

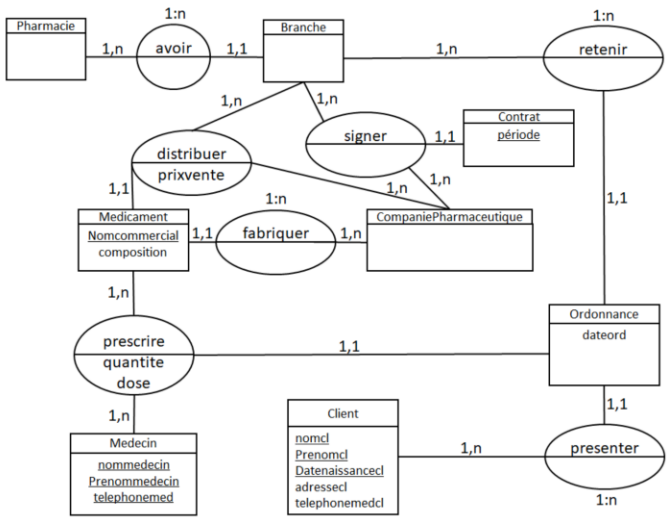
A19 : Gestion de parc de voitures

Un parc de voitures décide de réorganiser son système d'information en rassemblant les informations suivantes dans une base de données comme suit :
Chaque voiture est identifiée par un numéro de série et caractérisée par un nom. Une voiture est achetée par un seul propriétaire : soit par un particulier, soit par une entreprise. Chaque propriétaire est caractérisé par un identifiant propriétaire, un nom, une adresse et téléphones. Un particulier est identifié par un numéro de sécurité sociale (NSS). Une entreprise est identifiée par un numéro de registre. Si le propriétaire est une entreprise, nous devons connaître le nom de la personne à contacter. Nous devons également connaître la date d'achat de la voiture par un propriétaire. Chaque voiture a un type. Ce type spécifie la marque de la voiture (BMW, Volkswagen, etc.), son numéro de modèle et le type de carrosserie (Berline, Wagon, Limousine, etc.). Un certain nombre de conducteurs participent au parc. Chaque conducteur est caractérisé par un nom, une adresse, un numéro de téléphone et le type de voiture qu'il conduit, ainsi que le nombre total d'heures de conduite qu'il a effectué sur chacun de ces types de voiture. L'entretien des voitures est fait par des techniciens appartenant au parc. La réparation peut être effectuée par un ou plusieurs techniciens. Pour toute réparation, nous gardons trace de la cause de sa réparation, de sa date et de sa durée. Une voiture peut être réparée plusieurs fois par les mêmes techniciens. Chaque technicien est identifié par un NSS et caractérisé par un nom, une adresse, un numéro de téléphone et les types de voitures qu'il peut réparer. Un technicien expert peut contrôler un ou plusieurs techniciens débutants. Un technicien débutant doit être contrôlé par un seul expert.
En utilisant la méthode Merise pour la modélisation, proposez un diagramme E-A :



A18 : Gestion du Système d'Information de la chaîne de pharmacies

Une chaîne de pharmacies vous demande de concevoir une base de données représentant des informations sur les pharmacies, les compagnies pharmaceutiques, les médicaments, les clients des pharmacies ou patients et leurs médecins.La chaîne comporte plusieurs pharmacies dont chacune peut avoir différentes branches. Ces branches peuvent signer, indépendamment les unes des autres, des contrats, pour des périodes déterminées, avec des compagnies pharmaceutiques qui fabriquent et distribuent des médicaments. Notez qu'un médicament est fabriqué par une seule compagnie pharmaceutique qui le distribue sur différentes branches de pharmacies ayant signé des contrats avec la compagnie pharmaceutique qui fixe le prix de vente de ce médicament. Pour acheter un médicament, un client doit se présenter avec une ordonnance. Pour chaque ordonnance, la branche de la pharmacie retient le nom, prénom, date de naissance, adresse et téléphone du patient (ou client de la pharmacie) ; le nom, prénom et téléphone du médecin ayant prescrit l'ordonnance ; la date de l'ordonnance, les médicaments prescrits avec une quantité et une dose pour chacun. Concernant les médicaments, outre le prix de vente, chacun est caractérisé par un nom commercial. Un médicament est divisé en deux catégories : sirop et comprimé. Le sirop est caractérisé par le nombre de jours que le patient puisse utiliser le même flacon. Le comprimé est caractérisé par sa dose maximale par jour.
Proposez un schéma conceptuel dans le modèle "entité-association" (MCD).



Pour le reste des identifiants, des attributs et des types d'entités, ça dépend de la pensée de chaque étudiant; même pour les relations tertiaires.

Exercices SQL & Algèbre Relationnelle

Préparation pour Examen Final

Soit le schéma relationnel suivant concernant la gestion des personnels :

Departement (nodep, nomdep, nolocal)

Employe (nss, nomemp, adremp, datenaissanceemp, telemp, salaireemp, nodep)

Local (nolocal, localadresse)

I- Écrivez les expressions suivantes en algèbre relationnelle :

1- Déterminez les noms de tous les employés qui habitent à la même adresse que le département 'Comptabilité'.

$R1 = \pi_{[nodep, nomemp]} (Employe) \div_{[adremp = localadresse]} (\pi [nodep] (\sigma [nomdep = 'Comptabilité'] (Departement \bowtie Local)))$

Ou bien

$R1 = \pi_{[nodep, nomemp, [\delta[adremp \rightarrow adresse]]]} (Employe) \div (\pi [nodep, {}_{[\delta[localadresse \rightarrow adresse]]}] (\sigma [nomdep = 'Comptabilité'] (Departement \bowtie Local)))$

2- Déterminez le numéro et l'adresse du local de tous les départements qui contiennent des employés dont leur salaire est plus grand que 900$.

$\pi [nolocal, localadresse] (\sigma [salaireemp > 900] (Local \bowtie Departement \bowtie Employe))$

3- Déterminez les noms des employés qui ont un salaire inférieur à celui de l'employé 'Jean'.

$\pi [nomemp] (Employe \bowtie_{[salaireemp < salaireemp]} (\sigma [nomemp = 'Jean'] (Employe)))$

II- Écrivez les requêtes suivantes en SQL :

4- Affichez les noms des départements et le total des salaires des employés par département; pour tous les départements qui se trouvent à Paris et à Nice; pour tous les employés dont leur nom contient la lettre 'a' et contient à la deuxième position de la fin la lettre 'e', et leur date de naissance n'est ni en 1984 ni en 1986; triés par ordre descendant par le nom du département.

```
Select D.nomdep, sum(E.salaireemp)
From Departement D, Employe E, Local L
Where D.nodep = E.nodep
And D.nolocal = L.nolocal
And L.localadresse in ('Paris', 'Nice')
Group By D.nomdep
Having E.nomemp like '%a%'
And E.nomemp like '%e_'
And E.datenaissance not between '1984-01-01' and '1984-12-31'
And E.datenaissance not between '1986-01-01' and '1986-12-31'
Order by D.nomdep desc;
```

5- Créez une vue V_local composée du numéro du local 'V_nolocal', de l'adresse du local 'localadresse', et du nombre de département par local 'V_totnbdep'; pour tous les locaux qui ne se situent pas à paris et à Nice; et pour tous les départements dont leur nom ne contient pas la lettre 'o' à la deuxième position.

```
Create view V_local (V_nolocal , localadresse, V_totnbdep) As
Select L.nolocal, L.localadresse, count(D.nodep)
From local L, Departement D
Where L.nolocal = D.nolocal
And L.localadresse not in ('paris', 'Nice')
Group by L.nolocal, L.localadresse
Having D.nomdep not like '_o%';
```

6- Incrémentez le salaire de 5% de tous les employés dont leur nom se termine par la lettre 'x', leur date de naissance n'est pas en 1985 et 1986, et qui ne travaillent pas dans les départements de Comptabilité et de Vente.

```
Update Employe
Set salairemep = salaireemp * 1.05
Where nomemp like '%x'
And datenaissance not between '1985-01-01' and '1986-12-31'
And nodep not in (Select nodep
                  From Departement
                  Where nomdep = 'Comptabilité'
                  OR nomdep = 'Vente');
```

7- Supprimez les départements dont leur nom contient la lettre 'p' à la quatrième position et qui sont situés à Lyon.

```
Delete from Departement
Where nomdep like '___p%'
And nolocal in (Select nolocal
                From Local
                Where localadresse = 'Lyon');
```

8- Affichez les noms de tous les employés qui habitent à la même adresse que le département 'Comptabilité'.

```
Select nomemp
From Employe
Where adremp in (Select L.localadresse
                 From Local L, Departement D
                 Where L.nolocal = D.nolocal
                 And D.nomdep = 'Comptabilité');
```

9- Affichez le numéro et l'adresse du local de tous les départements qui contiennent des employés dont leur salaire est plus grand que 900$.

```
Select L.nolocal, L.localadresse
From Local L, Departement D, Employe E
Where L.nolocal = D.nolocal
And D.nodep = E.nodep
And E.salaireemp > 900;
```

10- Affichez les noms des employés qui ont un salaire inférieur à celui de l'employé 'Jean'.

```
Select nomemp
From Employe
Where salairemep < all (select salairemep
                        From Employe
                        Where nomemp = 'Jean');
```

Exercice 2 – Algèbre relationnelle et SQL (10 points)

On considère le schéma relationnel suivant :

Œuvre (idœuvre, titre)

Morceau (idmorceau, durée, iddisque, idœuvre)

Disque (iddisque, marque, type, dateparution, prix)

Execute (idmorceau, idinterprète)

Interprete (idinterprète, nom, adresse)

2.1- Écrivez les requêtes suivantes en algèbre relationnelle. (3 points)

a) Affichez l'identifiant de l'interprète (idinterprète) et le nom de l'interprète (nom) pour les interprètes qui ont exécuté des morceaux de durée de 2 minutes. **(1.5 pt)**

Réponse 2.1-a :

$\pi_{[idinterprète, nom]}$ (0.5 pt) $(\sigma_{[durée = 2]}$ (0.5 pt) (Interprete $\bowtie$ Execute $\bowtie$ Morceau) (0.5 pt))

b) Affichez l'identifiant du morceau (idmorceau), la durée du morceau (durée) et la marque du disque (marque); pour les morceaux qui sont enregistrés sur les disques dont leur date de parution (dateparution) est comprise entre février 2019 et juillet 2019. **(1.5 pt)**

Réponse 2.1-b :

$\pi_{[idmorceau, durée, marque]}$ (0.5 pt) $(\sigma_{[dateparution >= '2019\text{-}02\text{-}01'}$ (0.25 pt) $\wedge$ (0.25 pt) dateparution <= '2019-07-31' (0.25 pt)] (Morceau $\bowtie$ Disque) (0.25 pt))

2.2. Écrivez les requêtes suivantes en SQL. (7 points)

a) Créez une vue v_marque_disque qui contient la liste des marques (vmarque) et le nom des interprètes (vnom), pour toutes les marques qui produisent des disques qui contiennent des morceaux interprétés par des interprètes dont le nom contient la lettre 'a' et la lettre 'r' à la quatrième position de la fin et qui résident à Montréal et à Toronto; triez par ordre alphabétique selon le nom de l'interprète. **(2.5 pts)**

Réponse 2.2-a :

Create View v_marque_disque (vmarque, vnom) As (0.25 pt)

Select D.marque, I.nom (0.25 pt)

From Disque D, Morceau M, Execute E, Interprete I (0.25 pt)

Where D.iddisque = M.iddisque (0.25 pt)

And M.idmorceau = E.idmorceau (0.25 pt)

And E.idinterprète = I.idinterprète (0.25 pt)

And I.nom Like '%a%' (0.25 pt)

And I.nom Like '%r...' (0.25 pt)

And I.Adresse in ('Montréal', 'Toronto') (0.25 pt)

Order By I.nom; (0.25 pt)

b) Affichez pour chaque interprète son nom, le nombre de ses morceaux avec la moyenne de leurs durées. **(2 pts)**

Réponse 2.2-b :

Select I.nom, (0.25 pt) **Count(E.idmorceau)** (0.25 pt), **AVG(M.durée)** (0.5 pt)

From Interprete I, Execute E, Morceau M (0.25 pt)

Where I.idinterprète = E.idinterprète (0.25 pt)

And E. idmorceau = M. idmorceau (0.25 pt)

Group By I.nom; (0.25 pt)

c) Parmi les interprètes qui ont exécuté des morceaux de durée supérieure à 5 minutes, affichez les identifiants des interprètes (idinterprète) et le nombre de morceaux de ceux qui ont exécuté plus de 5 morceaux et qui ne sont pas résidents à Montréal. La minute est l'unité de mesure de la durée. **(2.5 pts)**

Réponse 2.2-c :

Select I.idinterprète, Count(E.idmorceau) (0.25 pt)

From Interprete I, Execute E, Morceau M (0.5 pt)

Where I.idinterprète = E.idinterprète (0.25 pt)

And E.idmorceau = M.idmorceau (0.25 pt)

And M.durée > 5 (0.25 pt)

And I.adresse <> 'Montréal' (0.25 pt)

Group By I.idinterprète (0.25 pt)

Having Count(E.idmorceau) > 5; (0.5 pt)

Exercice 5 – Déclencheurs (5 points)

Supposons que nous avons les deux tables suivantes :
Etudiant (matricule, nom_etudiant, adresse_etudiant, courriel_etudiant)
BulletinNote (matricule, session, cours, crédits, note)

5.1. Écrivez un déclencheur qui supprime tous les bulletins de notes dans la table « BulletinNote » de l'étudiant après la suppression des informations de cet étudiant dans la table « Etudiant ».

```
Create or Replace Function deleteBulletinNoteTrigger() Returns trigger AS $$
Begin
        Delete from BulletinNote
        Where BulletinNote.matricule = New.matricule;
        Return New;
End;
$$ language plpgsql;

Create Trigger af_del_etudiant
        After Delete On Etudiant
        For each row
        When (Etudiant.matricule = New.matricule)
        Execute procedure deleteBulletinNoteTrigger ();
```

Exercice 5- Déclencheurs (4 points)

Supposons que nous avons les deux tables suivantes :
Produit (idproduit, descriptionpr, quantitepr, prixunitairepr)
Commande (nocommande, idproduit, qtycommande, prix, montant)

5.1- Écrivez un déclencheur qui supprime toutes les lignes de commandes dans la table « Commande » du produit après la suppression des informations de ce produit dans la table « Produit ». (4 pts)
Réponse 5.1 :

```
CREATE OR REPLACE FUNCTION deleteCommandeProduit() (0.25 pt) RETURNS TRIGGER AS
$$ (0.25 pt)
BEGIN (0.25 pt)
    DELETE FROM Commande (0.25 pt)
    WHERE Commande.idproduit = :Old.idproduit (0.5 pt)
    RETURN Old; (0.25 pt)
END;
$$ languageplpgsql; (0.25 pt)

CREATE TRIGGER af_del_produit (0.25 pt)
    AFTER DELETE ON Produit (0.5 pt)
FOR EACH ROW (0.25 pt)
WHEN (Produit.idroduit = Old.idproduit) (0.5 pt)
EXECUTE PROCEDURE deleteCommandeProduit(); (0.5 pt)
```

Ou bien

```
CREATE OR REPLACE TRIGGER af_del_produit (0.5 pt)
    AFTER DELETE ON Produit (0.5 pt)
FOR EACH ROW (0.5 pt)
WHEN (Produit.idroduit = Old.idproduit) (0.5 pt)
Begin
    DELETE FROM Commande (0.5 pt)
    WHERE Commande.idproduit = :Old.idproduit (1 pt)
End; (0.5 pt)
```

erreur Adresselog ?

2.2. Écrivez la requête suivante en algèbre relationnelle et en SQL. (6.5 points)

Affichez le nom du passager (nompassager), le numéro du vol (numvol), la classe du vol (classevol), la ville de départ (idresselog) et la date de réservation (datéréservation) pour tous les vols dont la date de réservation (datedatéréservation) n'est pas dans l'année 2023 et le montant de réservation (montantréservation) n'est pas compris entre 700$ et 1000$ inclusivement.

Réponse 2.2 : Requête en algèbre relationnelle (3.5 pts)

π[nompassager, numvol, classevol, villedépart, datéréservation]

σ[datéréservation < '2023-01-01' $\vee$ datéréservation > '2023-12-31'
$\wedge$ montantréservation < 700 $\vee$ montantréservation > 1000]

(Passager $\bowtie$[Passager.idpassager = Réservation.idpassager]
Réservation $\bowtie$[Réservation.numvol = Vol.numvol] Vol)

Réponse 2.2 : Requête en SQL (3 pts)

```
Select  P.nompassager, V.numvol, V.classevol, V.ville départ, R.datéréservation
From    Passager P, Réservation R, Vol V
Where   P.idpassager = R.idpassager
AND     R.numvol     = V.numvol
AND     R.datéréservation  Not  Between  '2023-01-01'  and  '2023-12-31'
AND     R.montantréservation Not Between 700 and 1000
```

A23 : Gestion de formations continues.

Considérons un centre de formations continues dans lequel des participants suivent des cours pour obtenir des certificats dans divers domaines. Le centre de formation souhaite informatiser le système d'information en développant une base de données pour la gestion de formations continues.
Chaque participant est caractérisé par un identifiant, un nom complet composé d'un prénom et d'un nom, une adresse, un courriel, plusieurs numéros de téléphones propres à lui, et ses formations. Chaque formation est caractérisée par un nom unique, et ses propres cours. Chaque formation sera donnée sur différentes périodes. Chaque période est caractérisée par un identifiant, une date de début et une date de fin de la formation. Notons que plusieurs formations peuvent être données en parallèle pour une période déterminée.
Chaque cours est caractérisé par un sigle, une description, un prix et sa formation.
Chaque participant doit participer à un examen à la fin de la formation. Dans le cas où le participant passe l'examen, il obtient un certificat. Chaque certificat est caractérisé par une description, une date, un domaine et son participant.
Présentez le modèle Entité-Association avec la méthode Merise en intégrant :

Exercice 2 – Algèbre relationnelle & SQL (34 points)

Considérons le schéma relationnel suivant pour gérer la réservation d'un vol pour la période estivale :

Passager (idpassager, nompassager, villepassager, courrielpassager, notéléphonepassager)
Vol (numvol, classevol, datedépart, villedépart, datearrivée, villearrivée, tarif)
Réservation (numvol, idpassager, dateréservation, montantréservation)

À noter que le format de datedépart, datearrivée et dateréservation est 'AAAA-MM-JJ'.

2.1- Écrivez les requêtes suivantes en algèbre relationnelle. (12.5 points)

a) **Affichez l'identifiant du passager (idpassager) et le nom du passager (nompassager) pour tous les passagers qui habitent (villepassager) à Montréal ou à Laval et qui ont réservé un vol ayant une classe (classevol) 'Économique'. (3.5 pts)**

Réponse 2.1-a :

π[idpassager, nompassager] σ[villepassager = 'Montréal' $\vee$ villepassager = 'Laval'
$\wedge$ classevol = 'Économique']
(Passager $\bowtie$[Passager.idpassager = Réservation.idpassager] Réservation
$\bowtie$[Réservation.numvol = Vol.numvol] Vol)

b) **Affichez le numéro du vol (numvol), la ville de départ (villedépart) et la ville d'arrivée (villearrivée) pour tous les vols dont la ville de départ est la même que celle (villepassager) du passager (nompassager) 'Martin'. (4 pts)**

Réponse 2.1-b :

π[numvol, δ[villedépart $\rightarrow$ ville], villearrivée]
(Vol) $\div$ π[δ[villepassager $\rightarrow$ ville]] (σ[nompassager = 'Martin'](Passager $\bowtie$ Vol))

c) **Affichez le numéro du vol (numvol), la classe du vol (classevol) et le tarif du vol (tarif) pour tous les vols qui ont un tarif (tarif) compris entre 500$ et 900$ inclusivement, une date de départ (datedépart) en mars 2023, une date d'arrivée (datearrivée) qui n'est pas en juin 2023, et qui n'ont pas été réservés par des passagers. (5 pts)**

Réponse 2.1-c :

π[numvol, classevol, tarif] σ[tarif $\geq$ 500 $\wedge$ tarif $\leq$ 900
$\wedge$ datedépart $\geq$ '2023-03-01' $\wedge$ datedépart $\leq$ '2023-03-31'
$\vee$ datearrivée < '2023-06-01' $\vee$ datearrivée > '2023-06-31']
(Vol) $\div$ (Passager $\bowtie$[Passager.idpassager = Réservation.idpassager]
Réservation $\bowtie$[Réservation.numvol = Vol.numvol] Vol)

A23 :

Exercice 3 – Normalisation des relations

Considérez le schéma relationnel suivant non normalisé et comportant des clés primaires potentiellement incorrectes pour la gestion de vente d'articles dans un magasin :

Magasin (idmagasin, nommagasin, adressemagasin, notéléphonemagasin, courrielmagasin, idclient)

Client (idclient, nomclient, notéléphoneclient, courrielclient, idmagasin, nommagasin)

Article (idarticle, descriptionarticle, unitémesurearticle, prixunitairearticle, idmagasin, nommagasin)

Facture (nofacture, idclient, datefacture, montantfacture, nomclient)

Lignefacture (nofacture, idarticle, quantitéfacturée, prixdevente)

Notez que :

- Chaque magasin possède plusieurs téléphones et plusieurs clients.
- Chaque client possède plusieurs téléphones et peut acheter ses articles de plusieurs magasins.
- Chaque article existe dans un seul magasin.
- Chaque facture est pour un seul client.
- Chaque lignefacture peut contenir plusieurs articles.

3.1 - Présentez les dépendances fonctionnelles possibles et valides de toutes les relations mentionnées ci-dessus.

- DF Magasin
  - idmagasin -> nommagasin, adressemagasin, courrielmagasin
- DF Client
  - idclient -> nomclient, courrielclient, idmagasin
- DF Article
  - idarticle -> descriptionarticle, unitémesurearticle, prixunitairearticle, idmagasin
- DF Facture
  - nofacture, idclient -> datefacture, montantfacture
- DF Lignefacture
  - nofacture, idarticle -> quantitéfacturée, prixdevente

3.2 - Précisez la forme normale de chacune des relations dans le schéma relationnel ci-dessus. Justifiez votre réponse.

Magasin

- N'est pas en 1FN en raison des attributs non-clé `notéléphonemagasin` et `idclient` qui sont sont multivalués pour la même valeur de la clé idmagasin et aucune relation ne lie les attributs entre eux.

Client

- N'est pas en 1FN en raison des attributs non-clés `notéléphoneclient`, `idmagasin`, et `nommagasin` qui sont multivalués pour la même valeur de la clé `idclient` et il existe une relation entre les attributs `idmagasin` et `nommagasin`.

Article

- Est en 1FN, en 2FN mais pas en 3FN car la dépendance fonctionnelle `idarticle -> nommagasin` est indirecte et déduite par transitivité (idarticle -> idmagasin et idmagasin -> nommagasin).

Facture

- Est pas en 1FN mais pas en 2FN en raison de la DF (nofacture, idclient -> nomclient) non élémentaire vue la DF (idclient -> nomclient).

Lignefacture

- N'est pas en 1FN en raison de l'attribut non-clé `idarticle` est multivalué pour la même valeur de la clé `nofacture` et aucune relation existe dans la relation Ligne Facture.

3.3 - Proposez une correction des relations précédentes pour arriver à une normalisation en FNBC.

On divise la relation Magasin en deux relations.

R1 = Magasin(idmagasin, nommagasin, adressemagasin, courrielmagasin)

R2 = MagasinTelephone (idmagasin, notéléphonemagasin)

On divise la relation Client en trois relation.

R3 = Client (idclient, nomclient, courrielclient)

R4 = ClientTelephone (idclient, notéléphoneclient)

R5 = ClientMagasin (idclient, idmagasin)

R6 = Magasin(idmagasin, nommagasin, adressemagasin, courrielmagasin) *- Redondante à éliminer*

On divise la relation Article en deux relation.

R7 = Article(idarticle, descriptionarticle, unitémesurearticle, prixunitairearticle, idmagasin)

R8 = Magasin(idmagasin, nommagasin, adressemagasin, courrielmagasin) *- Redondante à éliminer*

On divise la relation Facture en deux relation.

R9 = Facture (nofacture, idclient, datefacture, montantfacture)

R10 = Client (idclient, nomclient, courrielclient) *- Redondante à éliminer*

On divise la relation Lignefacture en deux relation.

R11 = Lignefacture(nofacture, idarticle, quantitéfacturée, prixdevente)

R12 = Article(idarticle, descriptionarticle, unitémesurearticle, prixunitairearticle, idmagasin) *- Redondante à éliminer*

3.4 – Donnez le schéma relationnel final après normalisation FNBC.

R1 = Magasin(idmagasin, nommagasin, adressemagasin, courrielmagasin)

R2 = MagasinTelephone (idmagasin, notéléphonemagasin)

R3 = Client (idclient, nomclient, courrielclient)

R4 = ClientTelephone (idclient, notéléphoneclient)

R5 = ClientMagasin (idclient, idmagasin)

R6 = Article(idarticle, descriptionarticle, unitémesurearticle, prixunitairearticle, idmagasin)

R7 = Facture (nofacture, idclient, datefacture, montantfacture)

R8 = Lignefacture(nofacture, idarticle, quantitéfacturée, prixdevente)

Exercice 4 - Organisation physique et méthodes d'accès aux données.

4.1 - Énumérez les méthodes d'accès auz données

- Accès séquentiel aux rangées d'une table : Parcourir succesivement toutes les rangées d'une table, on commence par la première et finit par la dernière.
- Accès direct à une rangée : L'id de rangée permet d'identifeir chaque rangée de la BD de manière unique parmi toutes les autres rangées (id global).
- Accès séquentiel indexé : Les données sont organisées avec des index permettant d'accéder rapidement à des enregistrements spécifiques en utilisant des clés d'index.

4.2- Encerclez la lettre qui correspond à la bonne réponse dans les questions suivantes :

La méthode d'accès unidimensionnelle est basée sur :

a. Une combinaison de la clé primaire et de la clé étrangère
b. Une super-clé
c. Une clé d'accès ✓
d. Une clé étrangère
e. Une combinaison de la clé candidate et de la clé étrangère

La structure d'index est conservée :

a. Avec les données
b. Séparément des données ✓
c. Au niveau de la mémoire

Dans un arbre-B, les nœuds du dernier niveau représentent :

a. La racine
b. Les clés d'index
c. Les enregistrements
d. Aucune de ces options ✓

4.3- Considérons la table Produit (idproduit, descriptionproduit, unitémesure, prix):

Qu'est-ce qui se passera pour la requête suivante?

```
Select *
From produit
Where idproduit like '%c%"
And unitémesure = 'kg'
And prix >= 5;
```

La requête sélectionnera tous les produits dont l'ID contient la lettre "c", l'unité de mesure est "kg" et le prix est supérieur ou égal à 5. Notez que l'utilisation de l'opérateur LIKE avec un joker '%' au début rend l'indexation sur idproduit inutile si elle existe, car le moteur de base de données doit effectuer une recherche complète de la table pour trouver les correspondances. Cela peut ralentir considérablement la performance.

4.4- Indiquez quelle est la différence entre un index primaire et secondaire. Soyez clair.e.s et précis.e.s.

Un index primaire sert à ordonner physiquement en mémoire tandis qu'un index secondaire sert accélérer les opérations.