Exercice 3 – TD 7
Restaurant (NuméroMenu, NomMenu, NuméroPlat, Nomplat, Typeplat)

D = {NuméroMenu → NomMenu, NuméroMenu → NuméroPlat;
NuméroPlat → NomPlat; NuméroPlat → TypePlat}

1. En quelle forme normale ?
première forme normale : attributs atomiques = oui

deuxième forme normale : première forme normale plus attributs non clés dépendent de la totalité de la clé : oui puisque la clé est composée d'un seul attribut (numéroMenu)

troisième forme normale : non :
car typePlat dépend indirectement de numéroMenu, par transitivité par le biais de numéroPlat

2. décomposition
par algorithme de synthèse

R1(NuméroMenu, NomMenu, NuméroPlat)
soumise à D1 = {NuméroMenu → NomMenu,NuméroPlat}

R2(NuméroPlat, NomPlat, TypePlat)
soumise à D2 = {NuméroPlat → NomPlat; NuméroPlat → TypePlat}

R1 et R2 sont en troisième forme normale et aussi en BCNF
(en troisième forme normale et toutes les parties gauches des Dfs sont clés)

vérification qu'il n'y a pas de perte d'informations par la matrice des alphas

	NuméroMenu	NomMenu	NuméroPlat	NomPlat	TypePlat
R1	a1	a2	a3	~~b14~~ a4	~~b15~~ a5
R2	b21	b23	a3	a4	a5

en faisant jouer
la DF NuméroPlat → NomPlat : comme j'ai même partie gauche (a3), j'unifie les parties droites en remplaçant b14 par a4

la DF NuméroPlat → TypePlat : comme j'ai même partie gauche (a3), j'unifie les parties droites en remplaçant b15 par a5

j'ai une ligne de a => donc sans perte d'information

–

Exercice 5
R(A, B, C) soumise à F = {A → C}

décomposition proposée :
R1 (A, B) et R2 (B,C)

perte de la DF (A→ C)

- première manière pour la perte d'information

	A	B	C
R1	a1	a2	b13
R2	b21	a2	a3

impossibilité d'avoir une ligne de a => donc perte d'information

- seconde manière

prendre une instance de R pour ensuite la projeter
sur les 2 sous-relations et la recomposer par jointure
sur les attributs communs (B ici)

R

A	**B**	**C**
d1	b1	c1
d1	b2	c1
d2	b1	c2
d3	b1	c3
d2	b3	c2

projection sur R1 et R2

R1

A	**B**
d1	b1
d1	b2
d2	b1
d3	b1
d2	b3

R2

B	C
b1	c1
b2	c1
b1	c2
b1	c3
b3	c2

par jointure sur les attributs communs B la relation de départ

A	B	C
d1	b1	c1
d1	b1	c2
d1	b1	c3
d1	b2	c1
d2	b3	c2
d2	b1	c1
d2	b1	c2
d2	b1	c3
d3	b1	c1
d3	b1	c2
d3	b1	c3

ici on a plus d’information et donc on une perte d’information (perte de la sémantique associée à la DF A → C)

Exercice 6 ;
Remboursement(Id_emprunteur, nom, adresse, montantemprunte, datedemande, daterembourssement, montantremboursement)

1. requête

delete from remboursement r1 where sum(montantremboursement) =
(select montantemprunté from remboursement r2 where r1.idEmprunteur =
r2.idEmprunteur and r1.dateDemande = r2.datedemande) ;

2. recherche de Dfs s'appliquant à la relation

idEmprunteur → Nom
idEmprunteur → Adresse
idEmprunteur, DateDemande → MontantEmprunté
IdEmprunteur, DateDemande, DateRemboursement -> MontantRemboursement

calculer les clés de la relations
seule clé
IdEmprunteur, DateDemande, DateRemboursement

(fermeture transitive de IdEmprunteur, DateDemande, DateRemboursement
=
IdEmprunteur, DateDemande, DateRemboursement, Nom, Adresse,
MontantEmprunté, MontantRemboursement

Forme normale ?
Première forme normale : oui – les attributs sont atomiques
Deuxième forme normale : non – des attributs non clés ne dépendent pas de la totalité de la clé : par Nom ne dépend que d'une partie de la clé qui est IdEmprunteur

Algorithme de synthèse :

Emprunteur (IdEmprunteur, Nom, Adresse) cidEmprunteur → Nom
idEmprunteur → Adresse}

Emprunt(idEmprunteur, DateDemande, MontantEmprunté)
avec F_Emprunt ={idEmprunteur, DateDemande → MontantEmprunté}

Remboursement(IdEmprunteur, DateDemande, DateRemboursement, MontantRemboursement)
avec
F_remboursement = {IdEmprunteur, DateDemande, DateRemboursement -> MontantRemboursement}

les relations décomposées sont en BCNF

— Eléments de TP en prévision de la séance de la semaine prochaine (orientée projet)

Exemple sur la table Abonné reprise sur un serveur MySQL (testé sur les machines de la fac)

– reprise de la création de la table et insertion de tuples

```
CREATE TABLE ABONNE (
        NUM_AB  integer primary key,
        NOM VARCHAR(12),
        PRENOM VARCHAR(10),
        VILLE VARCHAR(30),
        AGE integer,
        TARIF float,
        REDUC float,
        CONSTRAINT DOM_AGE CHECK (AGE BETWEEN 0 AND 120));
```

```
INSERT INTO ABONNE VALUES (901001,'LEVEQUE','PIERRE','MONTPELLIER',40,500,NULL) ;
INSERT INTO ABONNE VALUES (902043,'DUPONT','MARIE','MONTPELLIER',20,200,20);
```

```
INSERT INTO ABONNE VALUES (902075,'RENARD','ALBERT','MONTPELLIER',18,200,NULL);
INSERT INTO ABONNE VALUES (911007,'MARTIN','LOIC','BEZIER',35,500,20);
INSERT INTO ABONNE VALUES (911021,'DUPONT','ANTOINE','MONTPELLIER',38,200,NULL );
INSERT INTO ABONNE VALUES (911022,'DUPONT','SYLVIE','MONTPELLIER',35,300,NULL);
INSERT INTO ABONNE VALUES (911023,'DUPONT','JEAN','MONTPELLIER',22,100,20);
INSERT INTO ABONNE VALUES (922011,'MEUNIER','LUC','MONTPELLIER',14,90,NULL);
INSERT INTO ABONNE VALUES (921102,'LUCAS','PAUL','MONTPELLIER',48,300,20);
INSERT INTO ABONNE VALUES (922143,'REVEST','ANNIE','MONTPELLIER',12,100,NULL);
INSERT INTO ABONNE VALUES (932010,'ANTON','JEANNE','MONTPELLIER',10,100,NULL);

–- test d'un déclencheur

CREATE TABLE ERREUR (message varchar(50));

DELIMITER //

CREATE OR REPLACE TRIGGER TEST BEFORE INSERT
ON ABONNE FOR EACH ROW
BEGIN
   IF NEW.VILLE IS NOT NULL   -- la ville est renseignée mais est
   AND NEW.VILLE != 'MONTPELLIER'       -- ni MONTPELLIER
   AND NEW.VILLE != 'BEZIER'       -- ni BEZIER
   THEN
      INSERT INTO ERREUR (message) VALUES ('Erreur insertion');
   END IF;
END ; //

-- changer le délimiteur pour le remettre à ; (contexte SQL)
DELIMITER ;

INSERT INTO ABONNE VALUES (932011,'ANTOINE','ZOE','BEZIERS',10,100,NULL);
INSERT INTO ABONNE VALUES (932016,'ANTOINE','BALTHAZAR','ALES',10,100,NULL);


– Test d'une fonction

DROP FUNCTION IF EXISTS NIVEAU_TARIF;
DELIMITER $$
CREATE FUNCTION NIVEAU_TARIF (TARIF FLOAT)
RETURNS VARCHAR(20)
DETERMINISTIC
BEGIN
   DECLARE NIVEAU VARCHAR(20);

   IF TARIF > 420 THEN
      SET NIVEAU = 'HAUT';
   ELSEIF (TARIF >= 180 AND
         TARIF <= 420) THEN
      SET NIVEAU= 'NORMAL';
   ELSEIF TARIF < 180 THEN
      SET NIVEAU = 'BAS';
   END IF;
   RETURN (NIVEAU);
END ; $$
DELIMITER ;


select num_ab, niveau_tarif(tarif) from abonne;
```