

TD1 BDD

Cahier des charges

↓ Analyse

Modèle Conceptuel < MCD
PCT

Modèle Logique < PLD
LOT

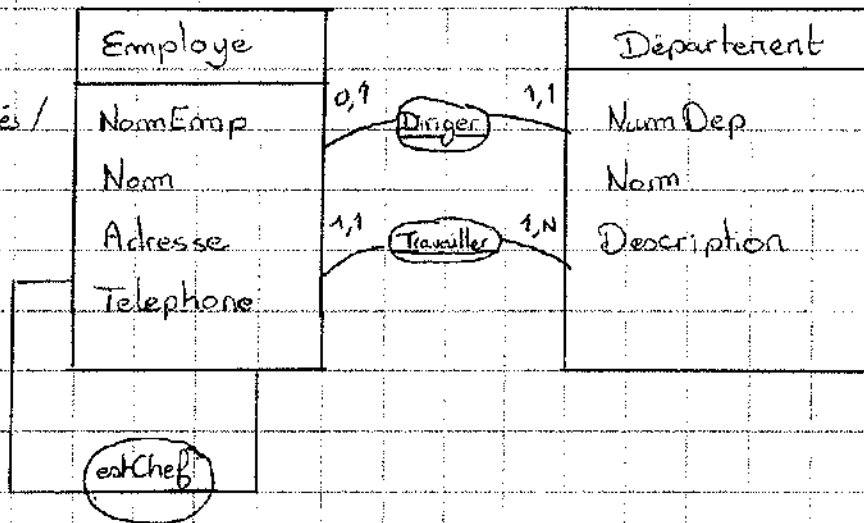
Modèle physique < PLD
MPT

MERISE / E-A / ER / EER < Architecture 3 niveaux
Séparer les données de traitement

* Entité

+ attributs / propriétés /
caractéristiques

* Association /
liens



° Rôle

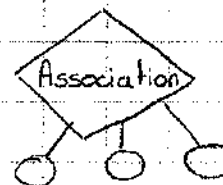
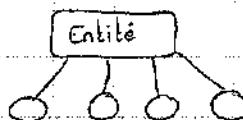
° Acté

° Cardinalité

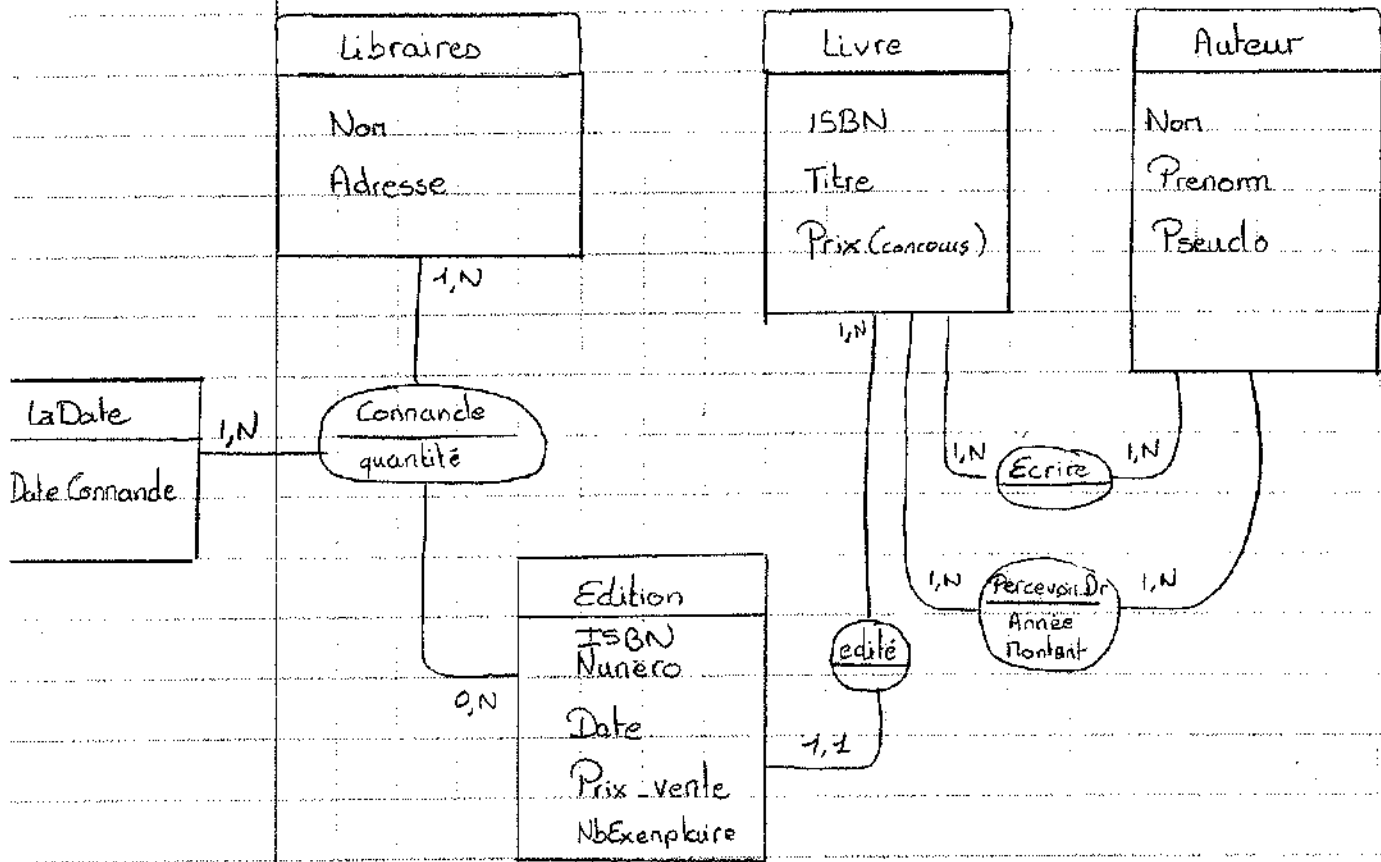
0,1 ou 0:1

0,N 0:N

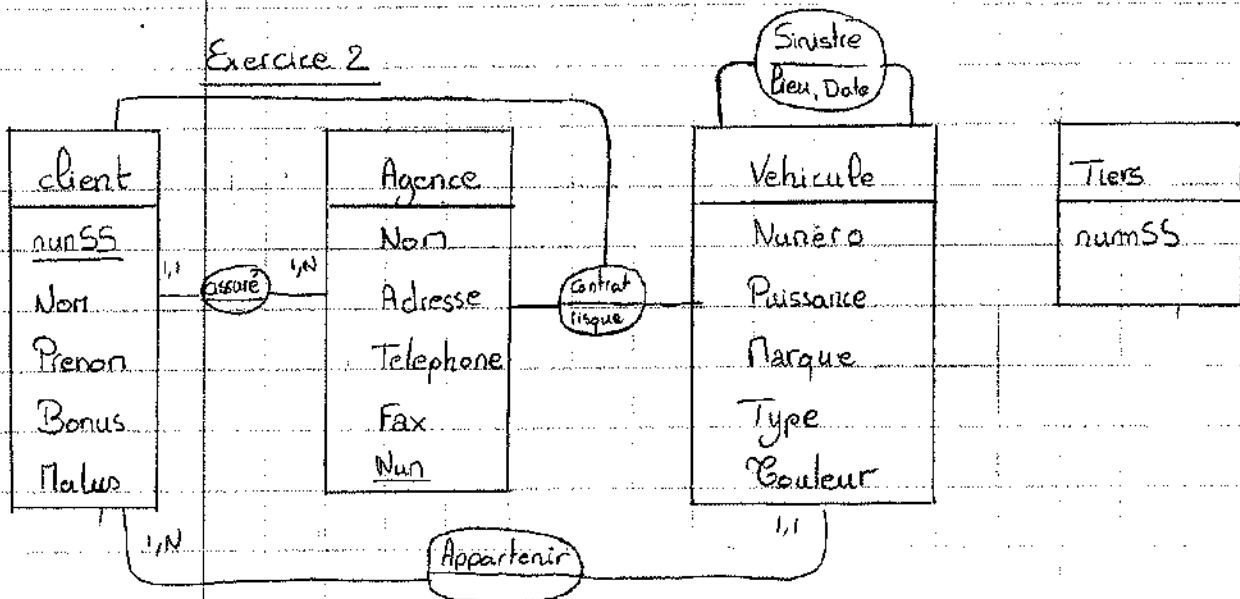
1,N 1:N



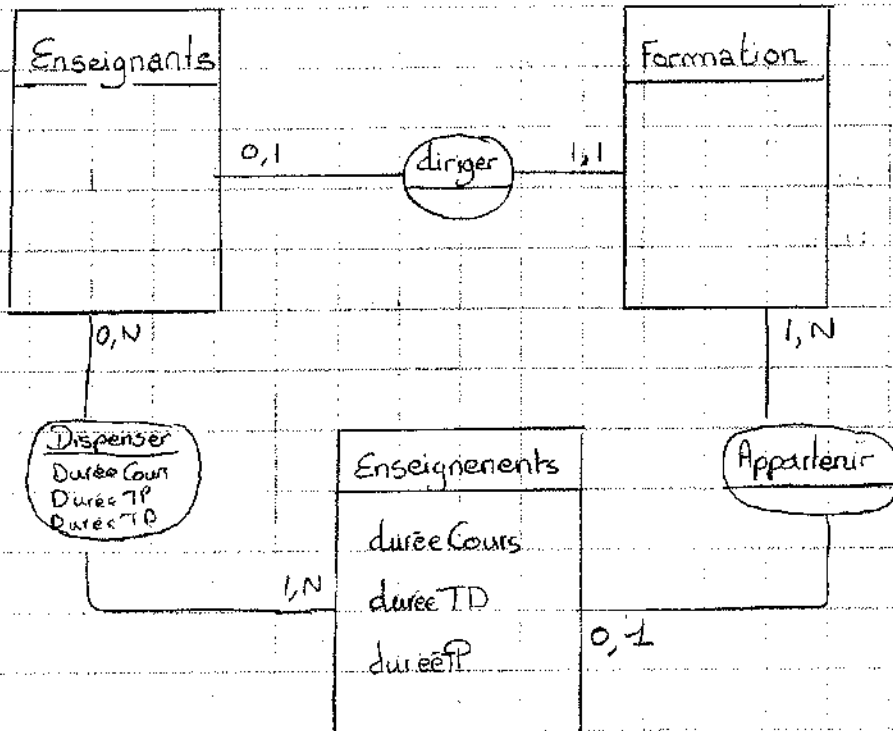
Exercice 1



Exercice 2

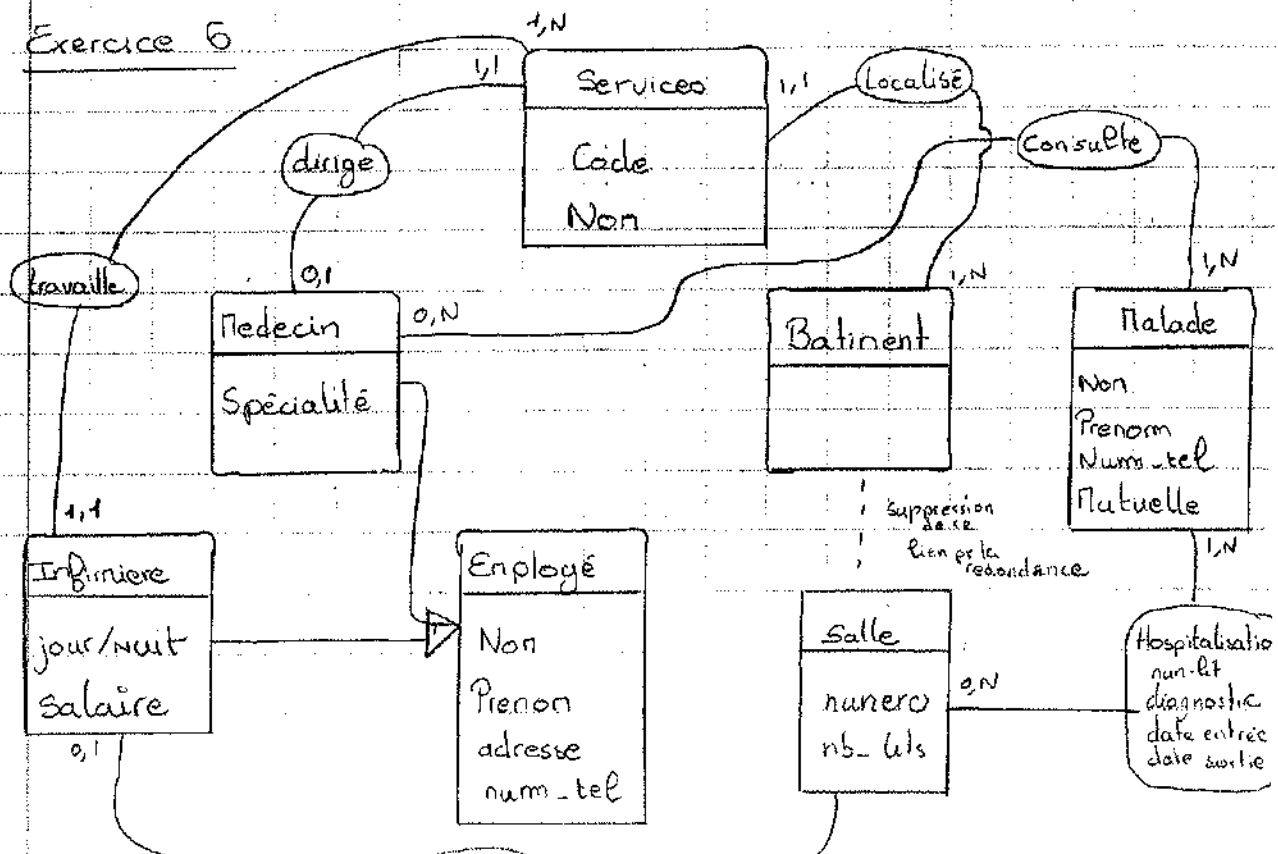


Exercice 3 MOD

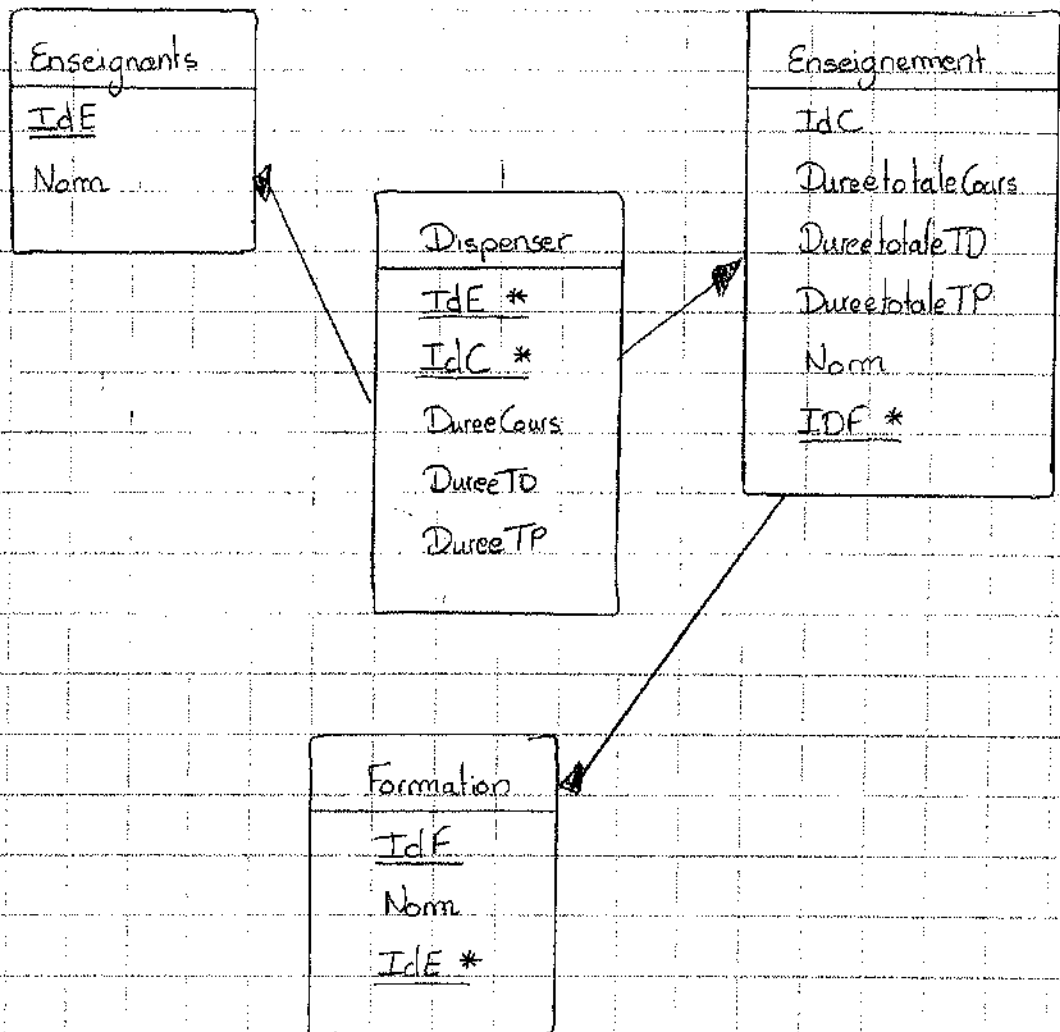


Exercice 4 cf. poly

Exercice 6

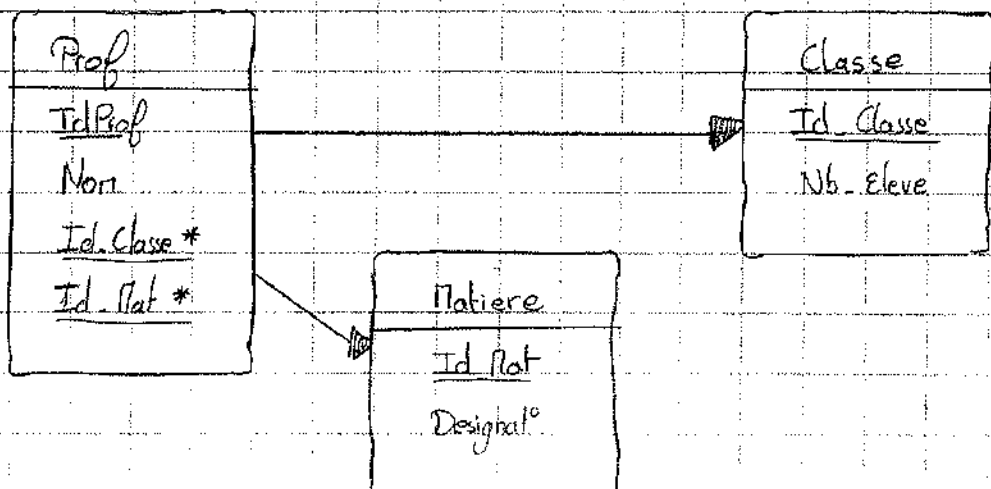


MLD de l'exercice 3



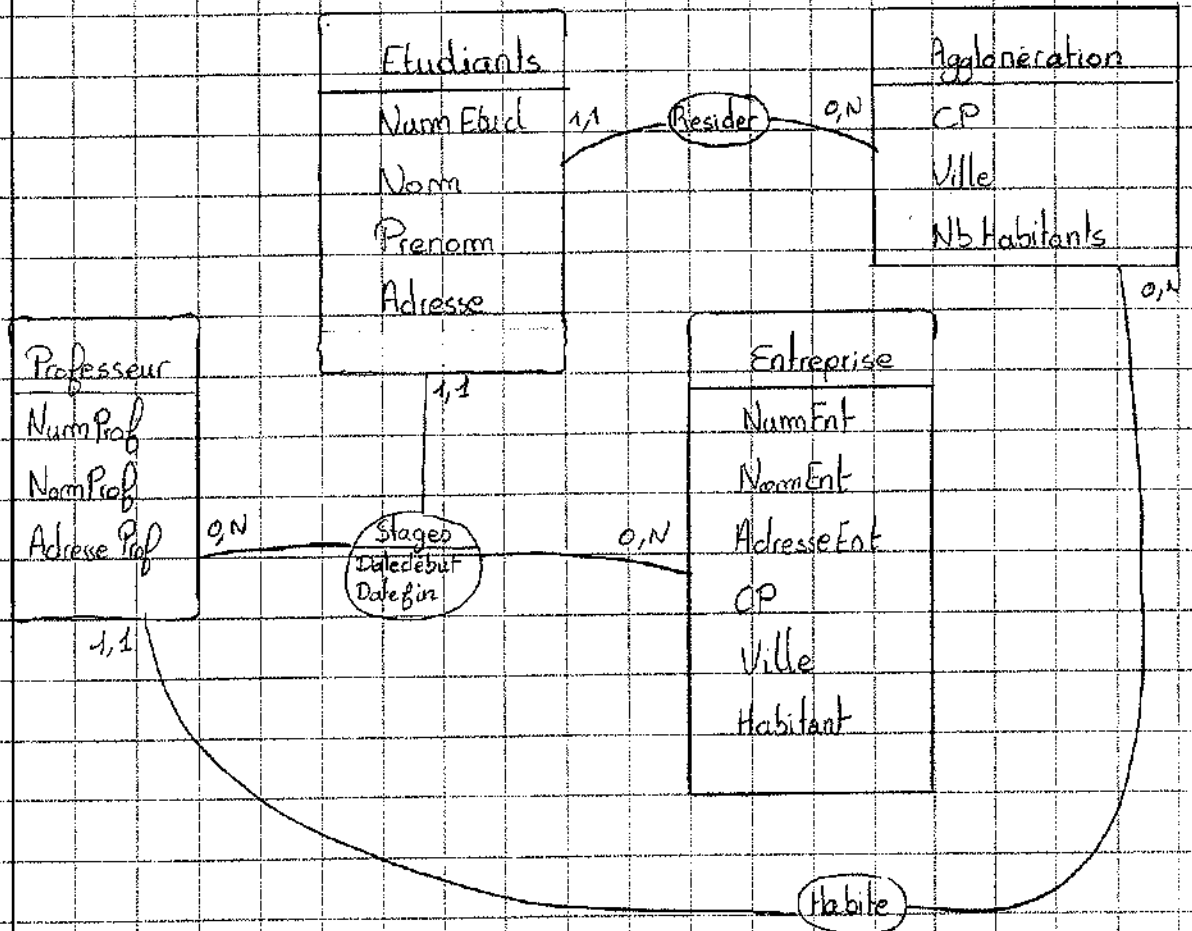
MLD de l'exercice 4

voir les règles



TD2

Application n°1 - Le geste



Application n°2 - la critique

a)

Num Produit	Nom Produit	Prix / Produit	Q stock Produit	Num Depot	Q Depot	CP Depot	Ville Depot
P ₁	Chaise	19,000	500	D ₁	1, av. republique	94800	Villejuiif
P ₁	Chaise	19,000	1800	D ₂	25, rue Paris	78000	Versaille
P ₁	Chaise	12,000	100	D ₃		89000	Lille
P ₂	Table	22,000	1500	D ₃		59000	Lille
P ₃	Tableau	35,00	0	Null	Null	Null	Null
Null	NULL	NULL	Null	D ₄	1, place..	31000	Toulouse

Tel Depot Fax Depot

b) Anomalies

- Redondance

→ les infos sont presentes dans les lignes contenant les produits de ce depots

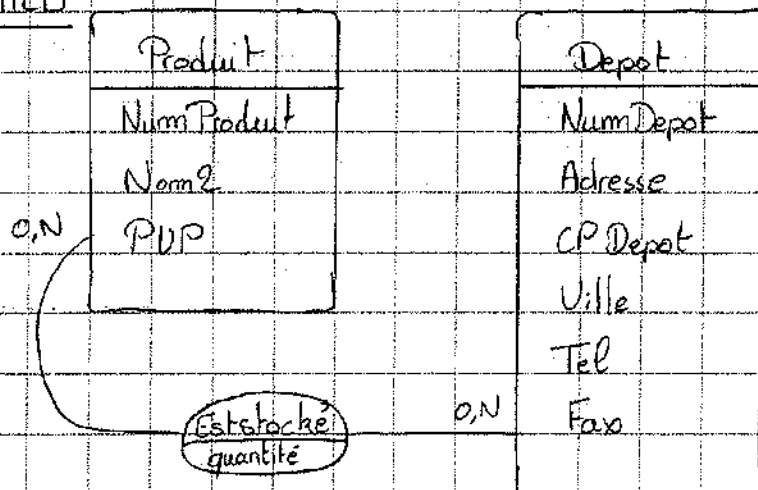
→ les infos sur un produit sont presentes dans tous les depots contenant ce produit

→ Anomalies lors de la MAS

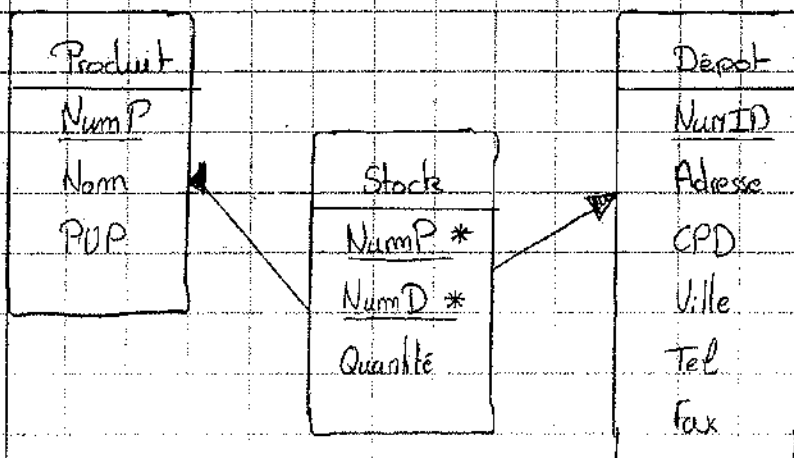
- Valeurs NULL → Anomalies MAS

c) NumProduit : identifiant des Produits
NumDepot : identifiant des Depots

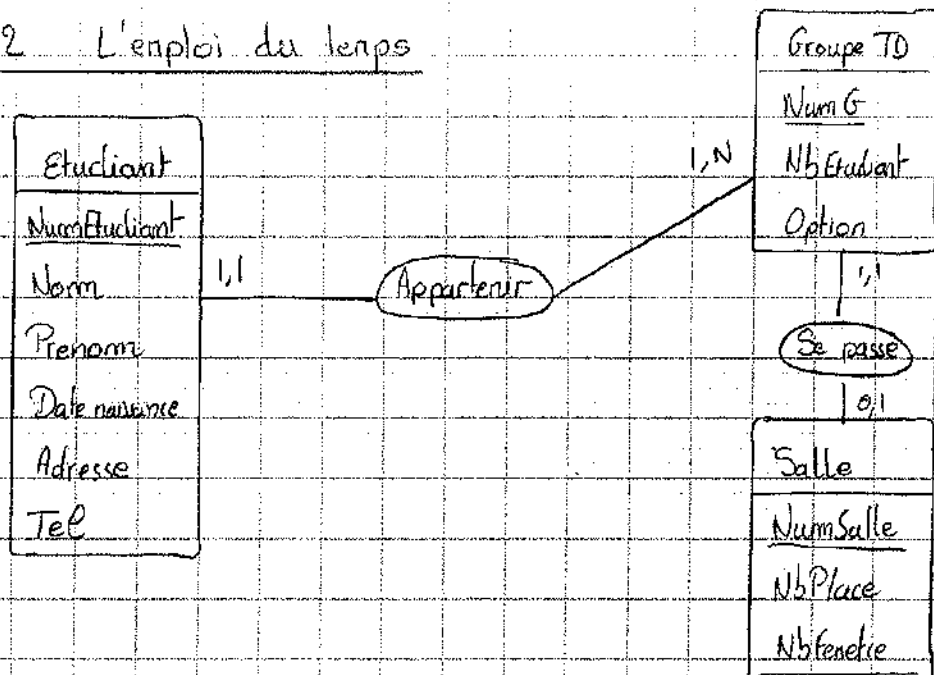
d) ICD



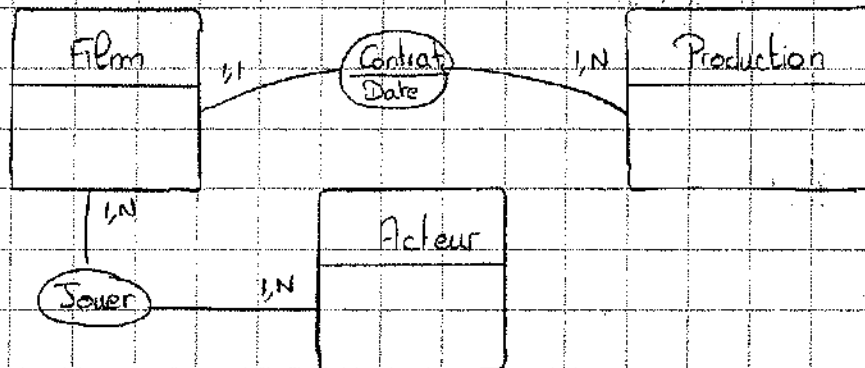
MLD



2.2 L'emploi du temps

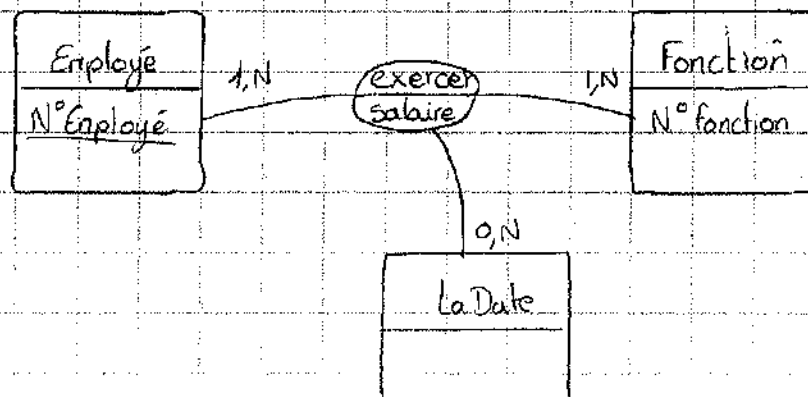


2.3 le cinema

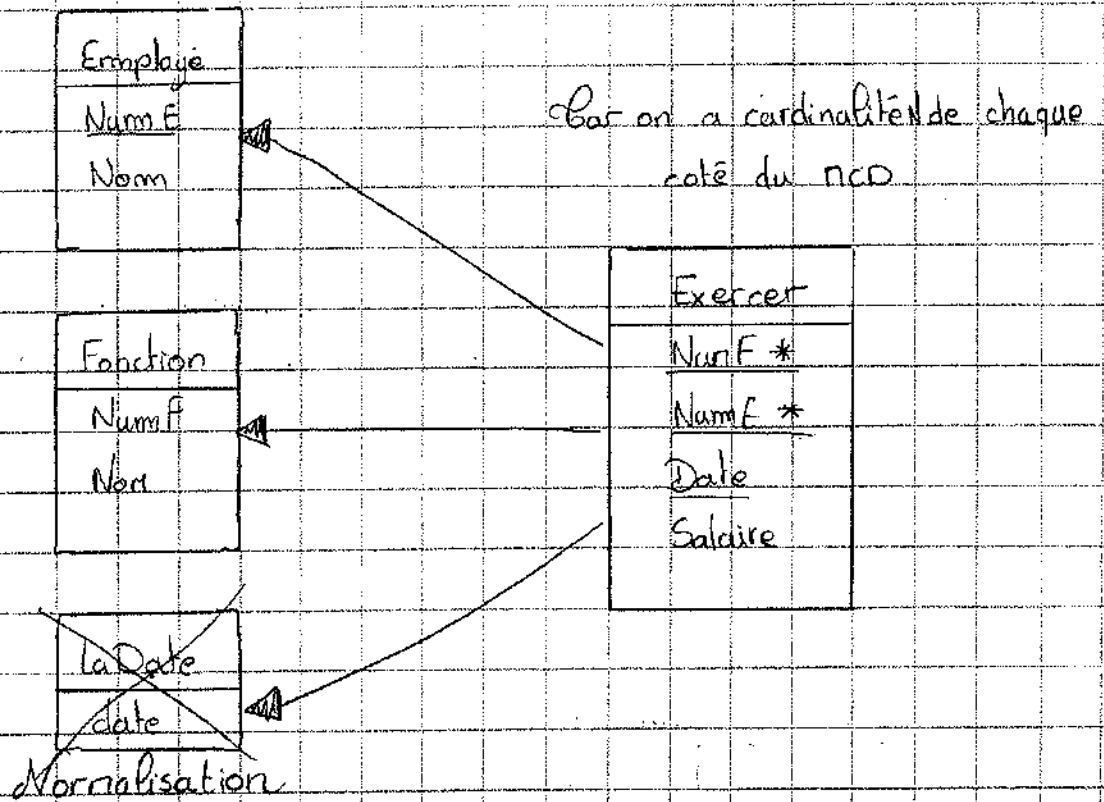


2.4 la carrière

MCD

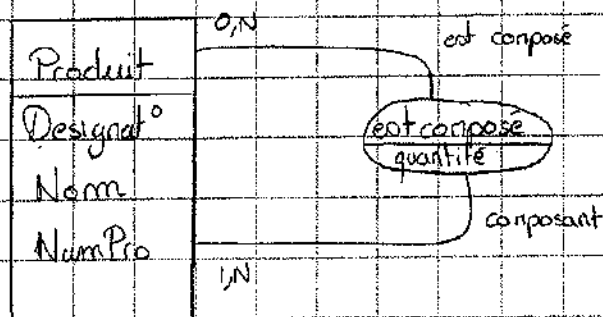


MLD

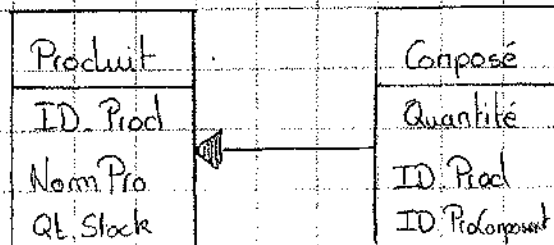


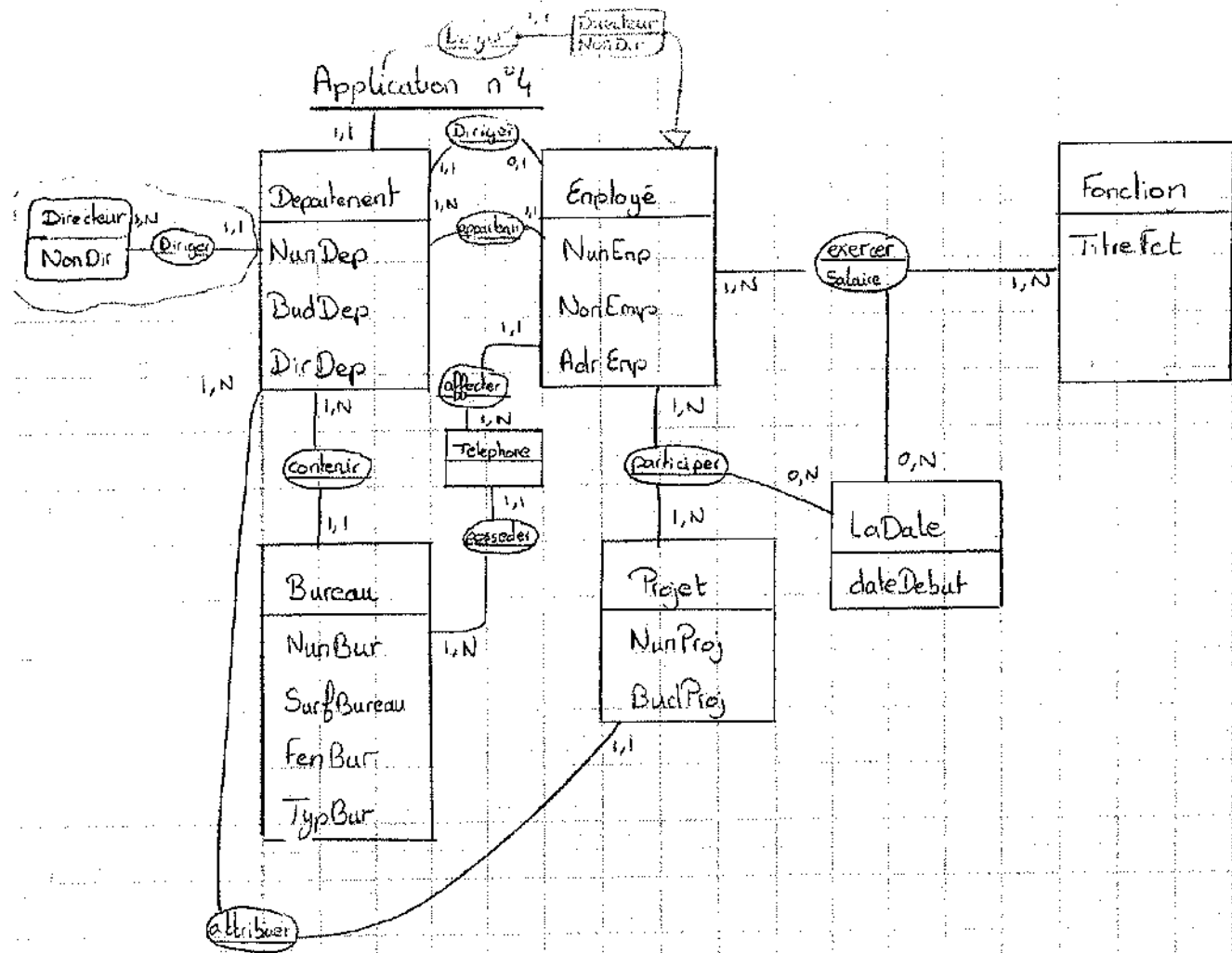
Application n°3

MCD



MLD





Application n°5

Rel1 (Cours, Etudiant, Note)

Rel2 (Cours, Etudiant, Heure, Salle, Professeur) - Heure: Heure + Date -

Rel3 (Etudiant, Examen, Horaire)

Rel4 (NunVol, DateDepart, Porte d'Acces, HeureDepart, Destination)

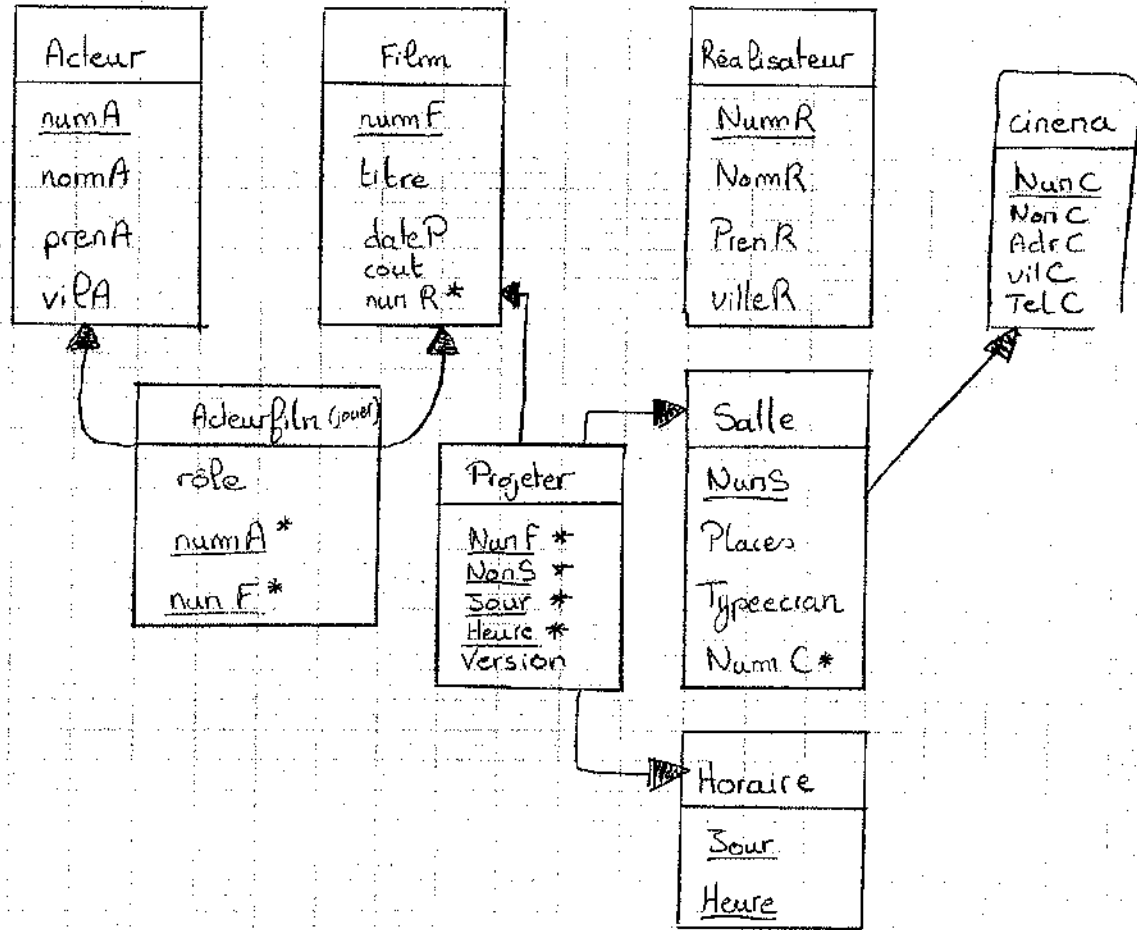
Rel5 (Employe, Directeur, Salaire, Service) - Employe sert comme clé primaire s'il n'a qu'un seul salaire -

Application n°6

1. cf poly (version est un attribut de l'association Projeter)
2. entité → relation

NLD

num * ← clé secondaire
si fonction multiple → relation
1,1 une simple flèche suffit

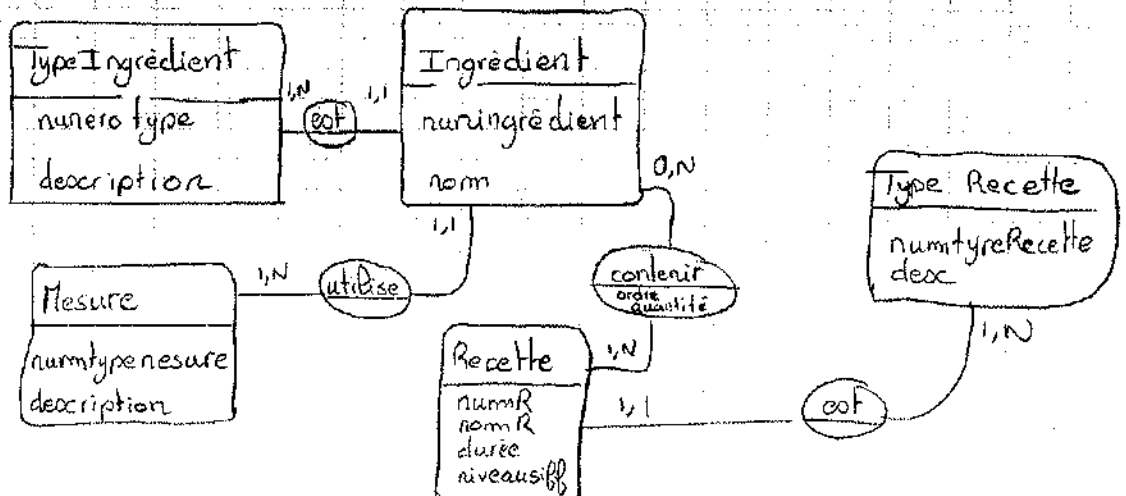


Application n°7

cf. poly

Partie A

Exercice 2



P_F : $\left. \begin{array}{l} \text{Departement} \rightarrow \text{Directeur} \\ \text{Employé} \rightarrow \text{Departement} \end{array} \right\} \Rightarrow \text{Employé} \rightarrow \text{Directeur}$

① Clé primaire ?

Employe ou Employé → Département
" " → Directeur

② Emploi en LFN ?

Emploi est en 1FN car tous ses attributs sont atomique

- mono valeur
- non composé

Emploi en 2FN ?

Emploi est en 2FN $\left(\begin{array}{l} 1 - \text{Emploi est 1FN} \\ 2 - \text{Pas de DF entre une partie de la clé} \\ \text{et un attribut non clé} \end{array} \right)$

Emploi en 3FN?

Emploi n'est pas 3FN car Département $\rightarrow$ Directeur
et Département n'est pas la clé

③ Décomposition Emploi

Solution 1 Emploi 1.1 (Employe, Departement *)

Emploi 12 (Département, Directeur)

Solution 2.

Emploi 21 (Employé, Département)

Emploi 22 (Employé, Directeur)

ID3

① cf. poly

② CREATE TABLE Eleve (
 NumEleve NUMBER(6) NOT NULL ✓ PRIMARY KEY
 Nom VARCHAR2(26) NOT NULL
 Prenom VARCHAR2(26)
 Date Naissance DATE
 Poids NUMBER(5,2) DEFAULT 60
 Année NUMBER(1) CHECK (Année BETWEEN 1 AND 5)
 PRIMARY KEY (NumEleve)
); CONSTRAINT PReleve PRIMARY KEY (NumEleve)

CREATE TABLE Activite (
 Niveau NUMBER(1) NOT NULL
 Nom VARCHAR2(20) NOT NULL
 Equipe VARCHAR2(30)
 CONSTRAINT PKActivite PRIMARY KEY (Niveau, Nom)
);

2-1

```
ALTER TABLE Eleves ADD CodePostal NUMBER(5)
" " " ADD Ville VARCHAR2(20)
```

2-2

```
UPDATE Eleves
SET CodePostal = '75013', Ville = 'paris'
WHERE NumEleve = 1;
```

2-3

```
CREATE TABLE Agglomeration (
  CP NUMBER(5),
  Ville VARCHAR2(20) CHECK (Ville = UPPER(Ville)),
  CONSTRAINT PKAgglo PRIMARY KEY (CP)
  ↳ CONSTRAINT CKville CHECK (Ville = UPPER(Ville))
);
```

2-4

```
INSERT INTO Agglomeration (CP, Ville) VALUES (75001, 'Paris');
```

2-5

```
UPDATE Eleves
SET ville = (SELECT Ville FROM Agglomeration WHERE
             CP = CodePostal);
```

4- Requêtes

B bis) SELECT nom, Nvl (Spécialité, '****') FROM professeurs

B bis) SELECT Eleve.nom Eleve.Prenom FROM Eleves E JOIN

Achivé Pratique AP ON E.numEleve = AP.numEleve WHERE UPPER(AP.Nom)

= 'SURF' And Ap Niveau = 1;

10) SELECT X.nom, Y.nom FROM professeur X, professeur Y
WHERE X.specialite = Y.specialite AND X.num_prof <
Y.num_prof

11) SELECT point * 5 FROM resultat NATURAL JOIN Eleves WHERE
nom = 'Tsuno'

12) SELECT AVG(poids) FROM Eleves WHERE annee = 1

13) SELECT SUM(points) FROM resultats WHERE NumEleve = 3

14) SELECT MIN(points), MAX(points) FROM resultats NATURAL
JOIN Eleves WHERE nom = 'Brisefer'

15) SELECT count(*) FROM Eleves WHERE Année = 2;

18) SELECT numeleve, nom, avg(points) FROM Eleves NATURAL JOIN
resultats WHERE annee = 1 group by numeleve, nom;

19) SELECT NumEleve, AVG(Points) as Moyenne FROM Resultats
GROUP BY NumEleve HAVING SUM(Points) > 40;

20) SELECT * FROM Eleves WHERE Année = 1 AND NumEleve
IN (SELECT NumEleve FROM Eleves E, Resultats R WHERE
Année = 1 AND E.NumEleves = R.NumEleve GROUP BY NumEleve
HAVING AVG(Point) > (SELECT AVG(Points) FROM Eleve, Resultats
WHERE Eleve.NumEleve, Resultat.NumEleve
AND Année = 1));

21) SELECT Nom, Poids FROM Eleves WHERE Année = 1 AND
Poids > (ALL ((SELECT DISTINCT Poids FROM Eleves WHERE Année = 2)
(SELECT max(Poids) FROM Eleves WHERE Année = 2))

22) SELECT Nom, Poids FROM Eleves WHERE Année = 1 AND Poids >
 $\frac{3}{\text{ANY (SELECT DISTINCT Poids FROM Eleves WHERE Année = 2)}}$;
SELECT MIN(Poids) FROM Eleves WHERE Année = 2

23) SELECT Nom, Poids, Année FROM Eleves E1 WHERE Poids >
(SELECT AVG(Poids) FROM Eleves E2 WHERE E2 Année = (P.Année))

24) SELECT Nom FROM Professeurs WHERE NumProf NOT IN
(SELECT DISTINCT NumProf FROM chargés WHERE NumAnn = 1);

25) SELECT Nom FROM Eleves E, Resultat R, ActiviteProf AP
 WHERE E.NumEleve = R.NumEleve AND E.NumEleve = AP.NumEleve
 AND Annee = 1 AND AP.Nom = 'Tennis'
 GROUP BY R.NumEleve, E.Nom
 HAVING AVG(Coef) > 12 ;

26)

Charge	S	Cours Q	Professeurs R
NumCours *	NumProf *	NumCours, Nom, Annee	NumProf
	1		1
	1		
	2		
	2		2
0	3	division	

SELECT * FROM Professeurs WHERE NOT EXISTS (SELECT * FROM Cours WHERE Annee = 2 AND NOT EXISTS (SELECT * FROM charge WHERE Charge.NumCours = Cours.NumCours AND Charge.NumProf = Professeur.NumProf));

Autre solution:

SELECT P.NumProf, P.Nom FROM Professeurs P, Cours Co, Charge Ch
 WHERE P.NumProf = Ch.NumProf AND Co.NumCours = Ch.NumCours
 AND Co.Anee = 2
 GROUP BY P.NumProf, P.Nom
 HAVING COUNT(*) = (SELECT COUNT(*) FROM Cours WHERE Annee = 2),

5 - Les vues

1- CREATE OR REPLACE VIEW ElevesAnnee1 AS (SELECT NumEleve, Nom, Prenom, DateNaissance, Poids FROM Eleves WHERE Annee = 1);

2- SELECT NumEleve, AVG (Points) AS moyenne FROM EleveAnnee1 NATURAL JOIN resultats GROUP BY NumEleves ORDER BY moyenne desc;

3- (SELECT Nom FROM ActivitesPratiquées) MINUS
(SELECT Ap.Nom FROM ActivitesPratiquées AP WHERE AP.NumEleve NOT IN (SELECT Num Eleve FROM Eleve Annee 1)),

TD5 BDD

I -

II - Creation de BD

```
CREATE TABLE Compagnie (  
  NumGrip CHAR(2)  
  Nom VARCHAR(25) NOT NULL  
  Pays VARCHAR(30),  
  CONSTRAINT PKCompagnies PRIMARY KEY (NumGrip)
```

```
CREATE TABLE Pilots (  
  NumPil  
  );
```

```
ALTER TABLE Pilote ADD CONSTRAINT FKPilotes
Compagnie FOREIGN KEY (NumComp)
REFERENCES Compagnie (NumComp);
```

III - Interrogation

① a.

```
SELECT NumPil, NomPil
FROM Pilotes
WHERE NumPil IN (SELECT DISTINCT NumPil
FROM Vols
WHERE (Upper(VilleD) = 'PARIS')
AND (Upper(VilleA) = 'TOULOUSE'))
```

b.

```
SELECT NumPil, NomPil
FROM Pilotes NATURAL JOIN Vols WHERE (Upper(VilleD)
LIKE 'PARIS') AND (Upper(VilleA) LIKE 'TOULOUSE');
```

c.

```
SELECT P.NumPil, NomPil
FROM Pilotes P, Vols V
WHERE P.NumPil = V.NumPil AND (Lower(VilleD)
= 'paris') AND (Lower(VilleA) = 'toulouse');
```

②

```
SELECT DISTINCT NumPil
FROM Vols
WHERE (Upper(VilleD) != 'PARIS') or (Lower(VilleA) !=
'toulouse'),
-- run
```

③

```
SELECT NumVols
FROM Vols
WHERE (DateA - DateD) BETWEEN 2*60*60
AND 4*60*60;
```

④ SELECT NumPil, NomPil
FROM Pilotes
WHERE Upper (VillePil) NOT IN ('MARSEILLE', 'NICE')

⑤ SELECT NumVol
FROM Vols JOIN Avions ON Vol.NumAv = Avion.Nu
WHERE Upper (NomAv) = 'A380' AND Upper (VilleD) IN
('LYON', 'PARIS');

⑥ SELECT ^{Apparition qu'une fois} DISTINCT NumPil
FROM Vols
WHERE Upper (VilleD) NOT IN ('MARSEILLE', 'PARIS')

⑦ SELECT *
FROM Pilotes
WHERE Upper (NomPil) Like 'A%' vide ou plein
↑ plein

⑧ SELECT NumPil
FROM Vols
WHERE (Upper (VilleD) = 'LILLE') AND (DateA - DATED > 60

opérateur Like
pour termes
génériques

⑨ SELECT NomAv, NumAv
FROM Avions
WHERE Upper (NomAv) NOT LIKE '%A%';

⑩ SELECT NumPil
FROM Vols
WHERE (DateA > Current_Time AND DateD < Current_Time

⑪ SELECT *
FROM Avions
ORDER BY NomAv, CapAv DESC;

⑫ SELECT COUNT(DISTINCT VilleD)
FROM Vols;

⑬ SELECT NumVol
FROM Vols
WHERE (DateA - DateD) = (SELECT Max(DateA - DateD)
FROM Vols);

⑭ SELECT NomPil, NaisPil
FROM Pilotes
WHERE VillePil = 'Villejuif' AND NaisPil > 35;

⑮ SELECT NumVol
FROM Vols
WHERE NumAv NOT IN (101, 401)

⑯ SELECT *
FROM clients C
WHERE NOT EXISTS (SELECT *
FROM Vols V
WHERE Upper(VilleD) = 'PARIS'

AND NOT EXISTS (SELECT *
FROM Reservations R
WHERE C.NumCP = R.NumCP
AND V.NumVol = R.NumVol))

Reservation

NumCP NumVol

Vols

NumVol

clients

NumCP

IV - Manipulations sur les jointures

Pilotes + Compagnies

						Supp		
NumPil	NomPil	NaisPil	VillePil	NumComp	NumComp		Nom	Pays
10	Paul		Paris	AF1	AF1		Air France	France
10	Paul		Paris	AF1	IBL		IBERIA	Espagne
10	Paul		Paris	AF1	LVS		Lufthansa	Allemagne
20	Elizabeth			AF1	AF1		Air France	France
20				AF1	IB2		IBERIA	Espagne
20				AF1	LVS		Lufthansa	Allemagne
30	Miguel			IB2	AF1		Air France	France
30				IB2	IB2		IBERIA	Espagne
30				IB2	LVS		Lufthansa	Allemagne

Produit Cartesien

```
SELECT *  
FROM Pilotes, Compagnies
```

② Jointure interne

```
SELECT *  
FROM Pilotes INNER JOIN Compagnies  
ON Pilotes.NumComp = Compagnies.NumComp
```

③ Jointure naturelle

```
SELECT *  
FROM Pilotes NATURAL JOIN Compagnies
```

④ JOIN externe gauche (ident que join car aucun pilote sans compagnies)

SELECT *
FROM Pilotes LEFT OUTER JOIN Compagnie ON
Pilotes.NumComp = Compagnie.NumComp

ou

SELECT *
FROM Pilotes LEFT JOIN Compagnies ON NumComp

⑤ Jointure externe droite (ident que jointure interne avec en plus la compagnie sans pilote (LVS Lufthansa Allemagne) avec la partie pilote à Null)

SELECT *
FROM Pilotes RIGHT OUTER JOIN Compagnies ON NumComp

ou

SELECT *
FROM Pilotes RIGHT JOIN Compagnie ON NumComp

Commande : MySQL

```
SELECT Nom  
FROM Compagnier LEFT JOIN Pilotes ON NumComp  
WHERE NumPil IS NULL
```

TD6 BDD

I - Transactions et contrôle de concurrence

I.1)

3. Configuration

↳ Implicite

→ CREATE / ALTER / DROP

→ DISCONNECT / EXIT

→ GRANT

→ mode de configuration automatique

} Connit
implicite

↳ Explicites : CONNIT

4. Annulation

↳ Explicite : ROLLBACK

↳ Implicite : déconnexion anormale (Panne)

I.3)

1. Exemple

2. Contraintes :

• l'attribut montant ≥ 0

- l'attribut Client dans Voyage doit faire Référence à Client dans Banque

3. Programme incorrect car Jules peut voyager gratuite

- soit suite a une panne apres le INSERT
- Soit UPDATE echoue car le montant nouveau est nég

4. Solution mettre INSERT et UPDATE dans une table transaction CSET AUTOCONMIT OFF et COMMIT apres UPDATE)