II - Programmation avancé T-SQL :

A - procedures stockées suivantes :

1 - create procedure afficheFamilles1

as

select * From AdHerent A, Inscription_majeur m

where A.N_adh = m.N_adh

AND montant > 0

Go // Fin Let exec afficheFamilles1

2 - create procedure afficheFamilles2

as Begin

Select * From AdHerent A, Inscription_majeur m

, Activite t where A.N_adh = m.N_adh

AND m.Nom = t.Nom

m.Nom

AND (select count(Nom) From Activite) > 2

Go

decimal

exec afficheFamille2 ;

3) create procédure afficheFamille3

as

declare @ montant global decimal

set @ montant global = (select Sum (montant)

From Adherent A, Emcription_majeur m

Inscription_Enfant E

Where A . N_adh = m . N_adh

AND m . Nom = E . Nom)

print ' Le montant globale est : ' + @ montant global

↓

convert (varchar(30), @montant glob)

end

go // Fin Lot

exec afficheFamille2 ;

3)

Create procédure afficheFamille3

as

declare @ a int, @ b int, @ c varchar(30)

, @ d decimal, @ E date

declare C cursor

for

select N_adh, count (nom epoux), Nom, sum (montant)

, 'Lolo' From Adherent A, Enfant E

Inscription_majeur m, Inscription_enfant t

Activité V, where A. N_adh = E. N_adh

.A. N_adh = m. N_adh

m. Nom = t. Nom

t. Nom = V. Nom

m. Nom = V. Nom

open C ;

Fetch next From C into @a, @b, @c, @d, @e

while @@ Fetch_status = 0

Begin count(varchar(5), @a)

print 'La famille ' + @a + ' ayant ' + @count(varchar(10), @b)

' montant global ' + count(varchar(10), @d)

' de activité ' + @c + ' annuel : ' + count(varchar(10), @E)

Fetch next from C into @a, @b, @c, @d, @E

end

close C;

deallocate C;

Go

B- Les fonctions :

Create function afficheEnfant(@a varchar(20))

returns int

as

@a = (select Count(nom) From Inscription-enfant E

, Activités t where

E.Nom = t.Nom

AND @t.type = @a

return @a

Go

print 'Le Nombre d'enfants inscrit au foot "

+ dbo.afficheEnfant("Football"));

7 - processus des traitement de vérifier le délai (dir)

8 - avis favorable (

9 - retour de dossier (dir → dem) (flux out)

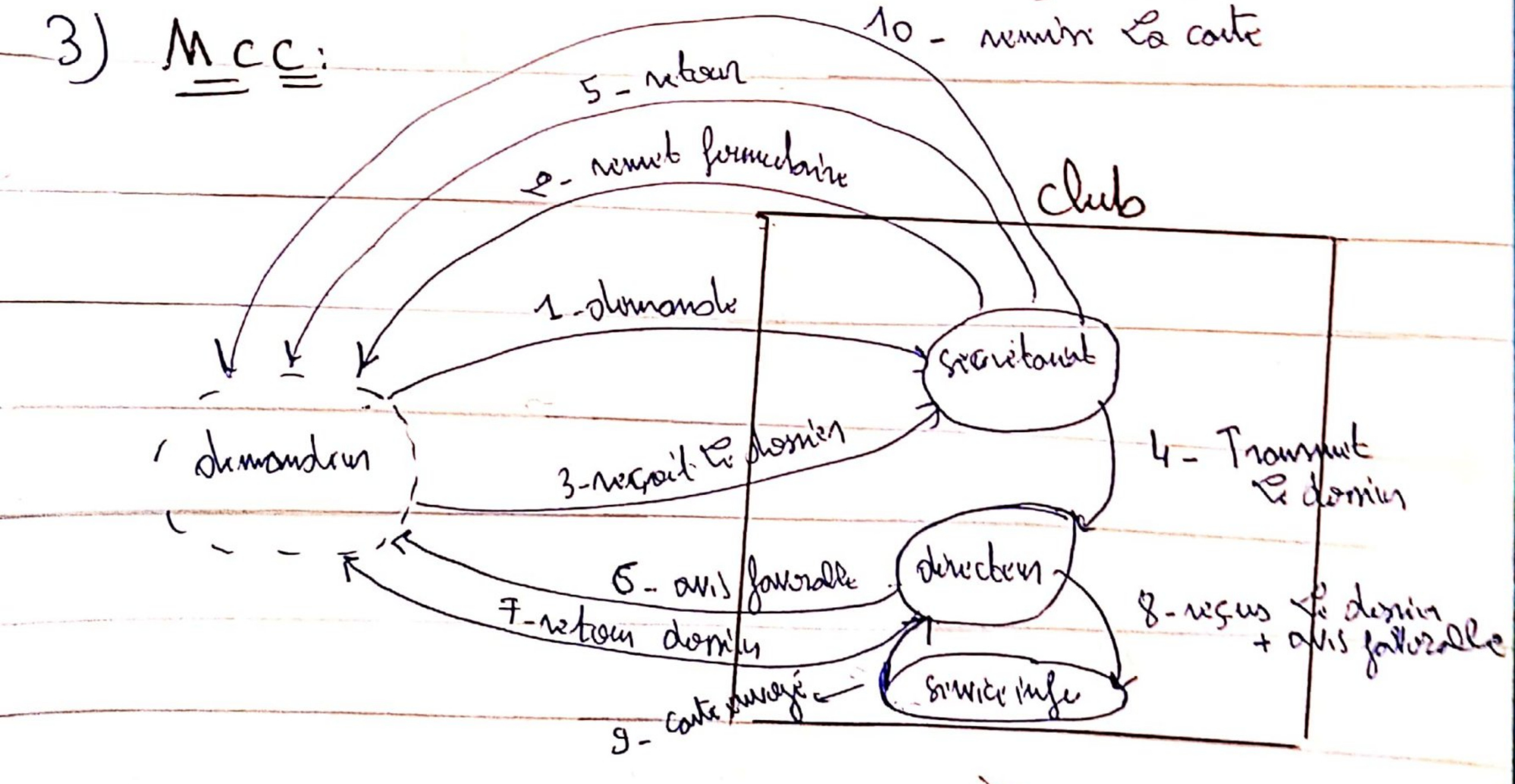
10 - reçus de dossier avec avis favorable (info ← dir)

11 - enregistré dans la base données

12 - carte envoyé a la direction (info → dir)

13 - remise de carte (sec → dem) (flux out)

3) MCC:



Acteur	Type	Rôle
- demandeur	externe	- demandeur :
- secrétariat	interne	+ passe la demande
- directeur	interne	- secrétariat :
- service informatique	interne	+ remet un formulaire à remplir
		- directeur :
		- Validation des dossiers
		- service informatique
		enregistre à la base de données

- Activité : Création demande d'adhésion

- flux messages échangés :

(flux in) 1 - passe la demande (demandeur → secrétariat)

(flux out) 2 - remet un formulaire (sec → dem)

(flux in) 3 - reçoit le dossier (dem → sec)

4 - vérification préliminaire du dossier (sec)

5 - Transmet à la direction (sec → directeur)

(flux out) 6 - remet au demandeur (retour) (sec → dem)

e- create function afficheIDnimi () returns Table
as
returns
select Nom from Activités A, Inscription_majeurs m,
Adhérent R,
Where A. Nom = m. Nom
AND m. N_adR = R. N_adR
AND R. mon mot = 'IDnimi';
Go

Select * From .dbo.afficheIDnimi()

I- Modélisation dynamique

1- Domaine | organisme : Clubs Ma fondation

2- Remplire de Tableau :

4 -

Evenement déclencheur : demande - Envoi du dossier
- Transmet

Evenement Resultat : remet formulaire - retour dossier par sec

- retour dossier par dir - remise à cote

opérations : - Vérification préalable de dossier

- procède au traitement de vérifier

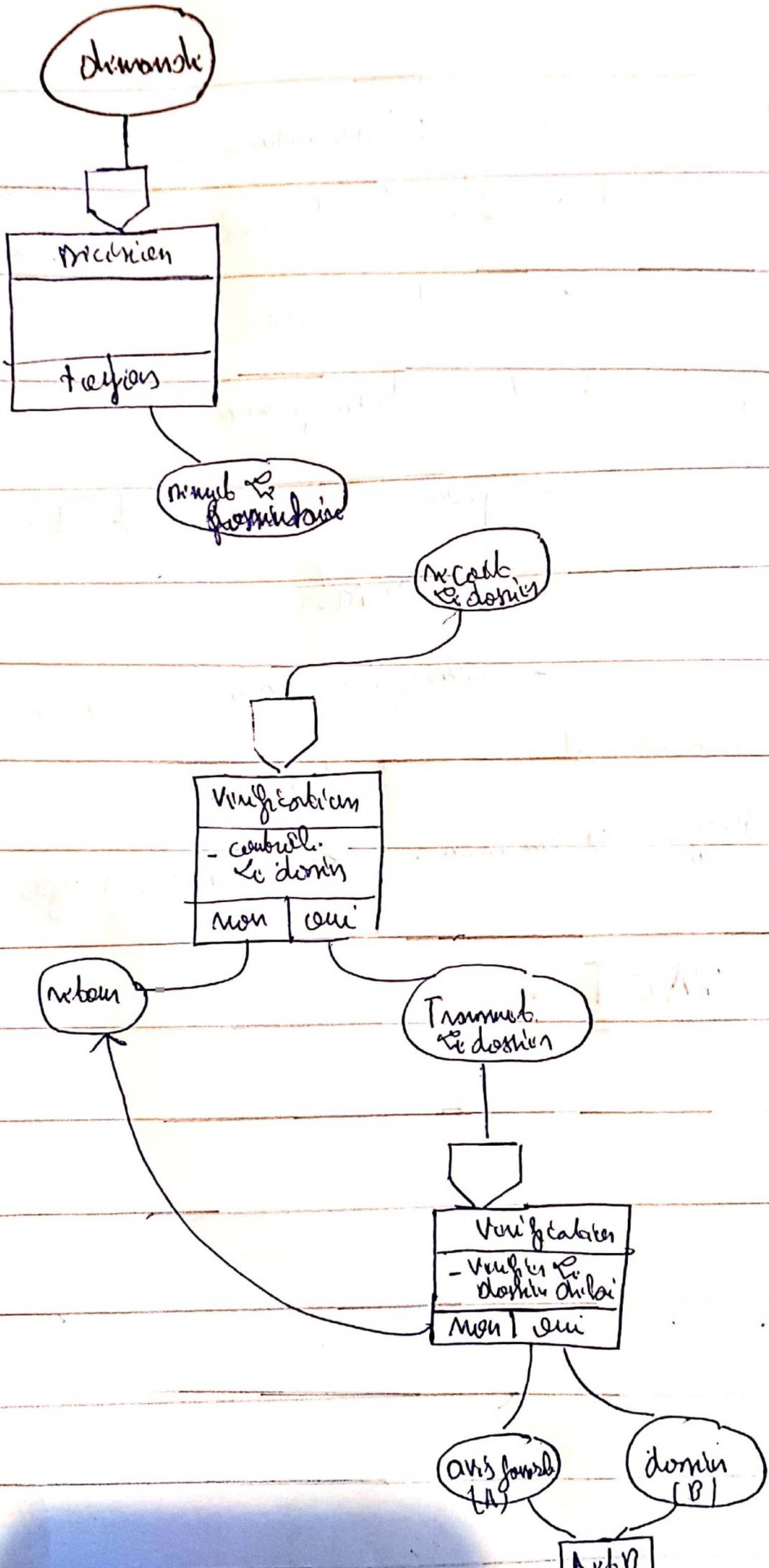
le dossier

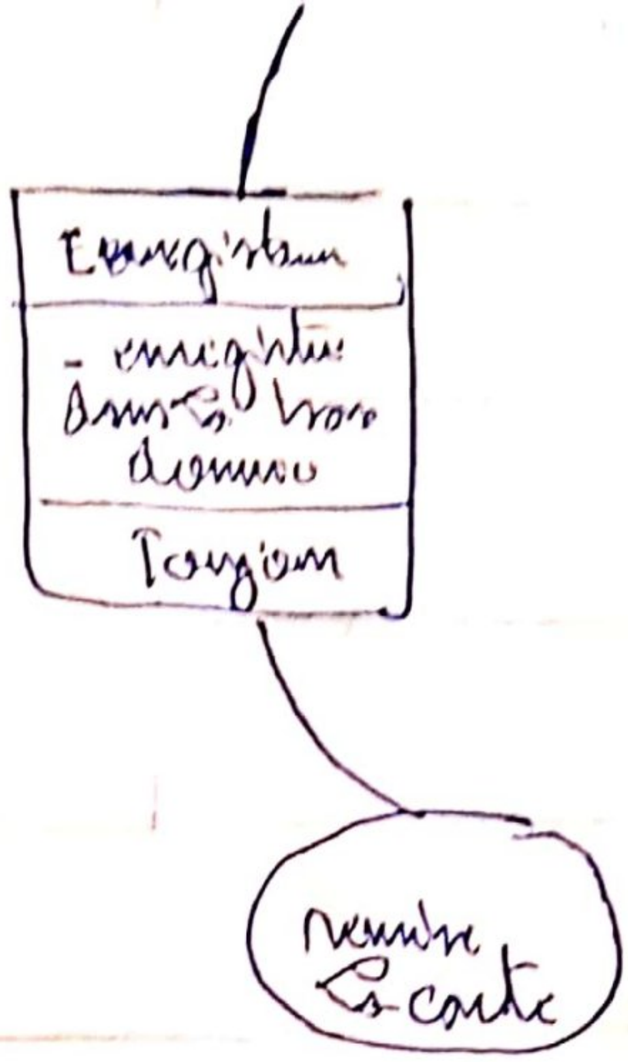
- Enregistre dans la base données

Synchronisation : A ou B

Regles d'emission : oui ou non, Toujours

5) MCT :





6 - MoT :

phase 1 :

phase 2 :

phase 3 :