

The slide has a white background with a yellow pencil on the right side. The title 'Evaluation' is in large blue text. Below it is a bulleted list of evaluation components:

- **Contrôle des connaissances :**
- 1 écrit (CC) de modélisation de 1h15 (25%) le 03/10
- 1 écrit (CC) de SQL de 1h15 (25%), soit le 21/11
- 1 un examen final de 1^{ère} session de 2h (50%)
- Deuxième session : 1 examen de rattrapage (100%)

At the bottom left are three colored pencils (yellow, green, and red). A small number '2' is in the bottom right corner.

Bases de données

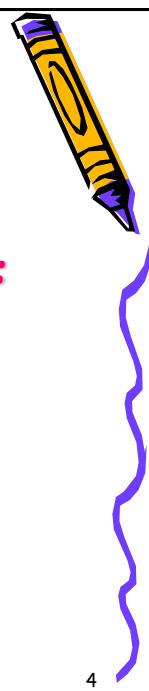


- Définitions
- Glossaire des SGBD relationnels
- Démarche
- Structures physiques



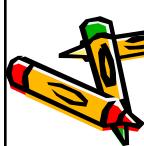
3

BDD : définitions



BDD = Bases Des Données :

- Fichier
- Attribut
- Identifiant
- Bases de données
- SGBD

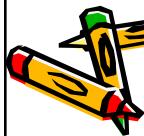


4

BDD : glossaire SGBDR

SGBDR = Système de Gestion des Bases de Données Relationnel

- Table ou relation
- Clé primaire
- Clé externe(étrangère)
- Champ ou attribut
- Domaine
- Tuple



5



BDD : démarche

- Conception : modélisation, normalisation
- Développement : création, optimisation
- Utilisation
- Maintenance



6

BDD : Conception



- Méthode Merise : MCD, MLD, MPD

MCD = Modèle Conceptuel des Données

MLD = Modèle Logique des Données

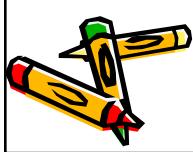
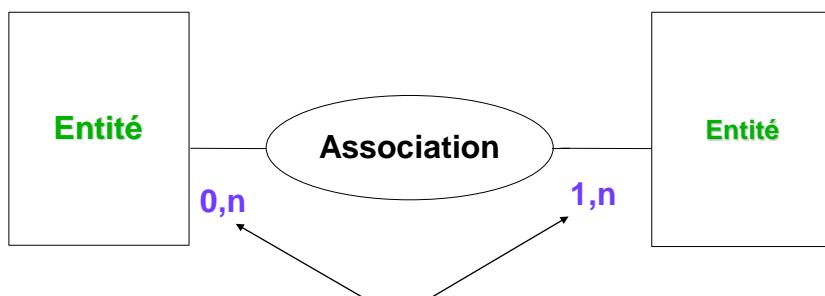
MPD = Modèle Physique des Données

- Normalisation : formes normales

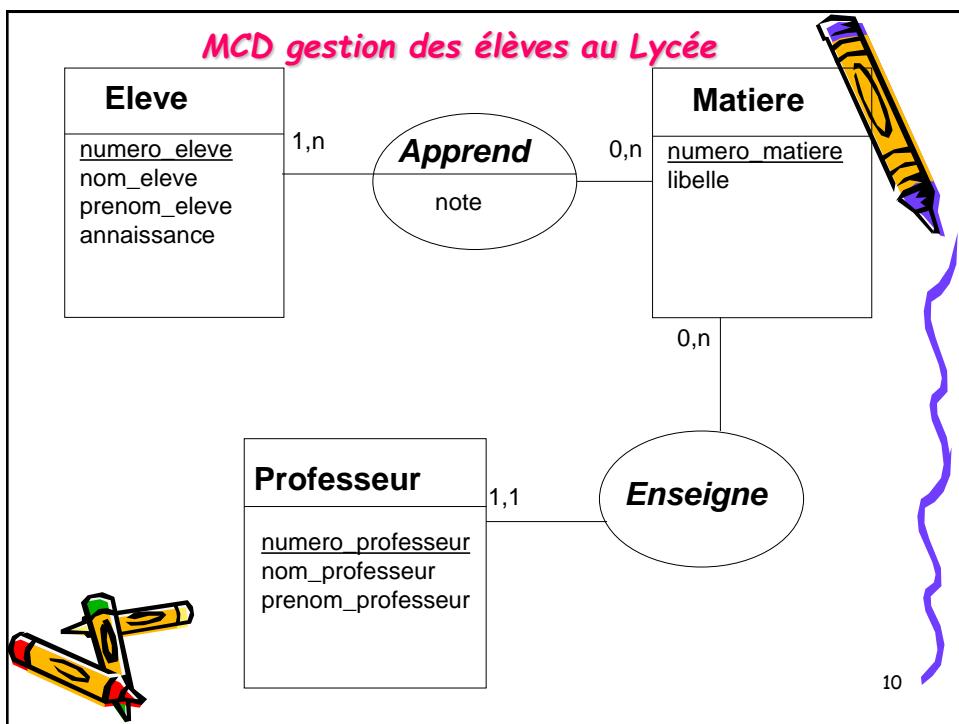
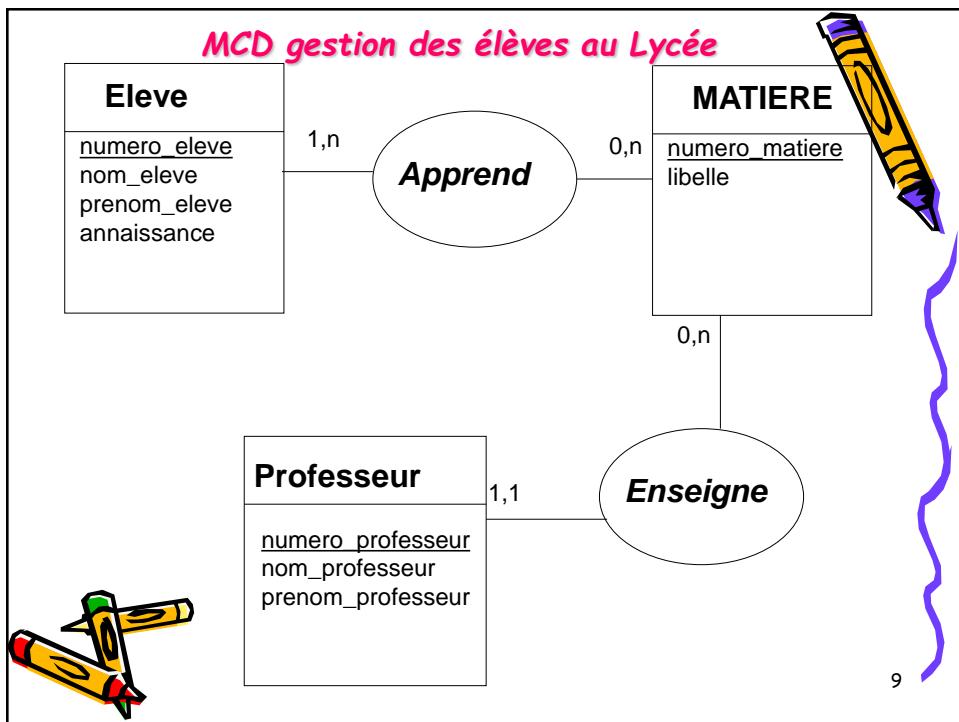


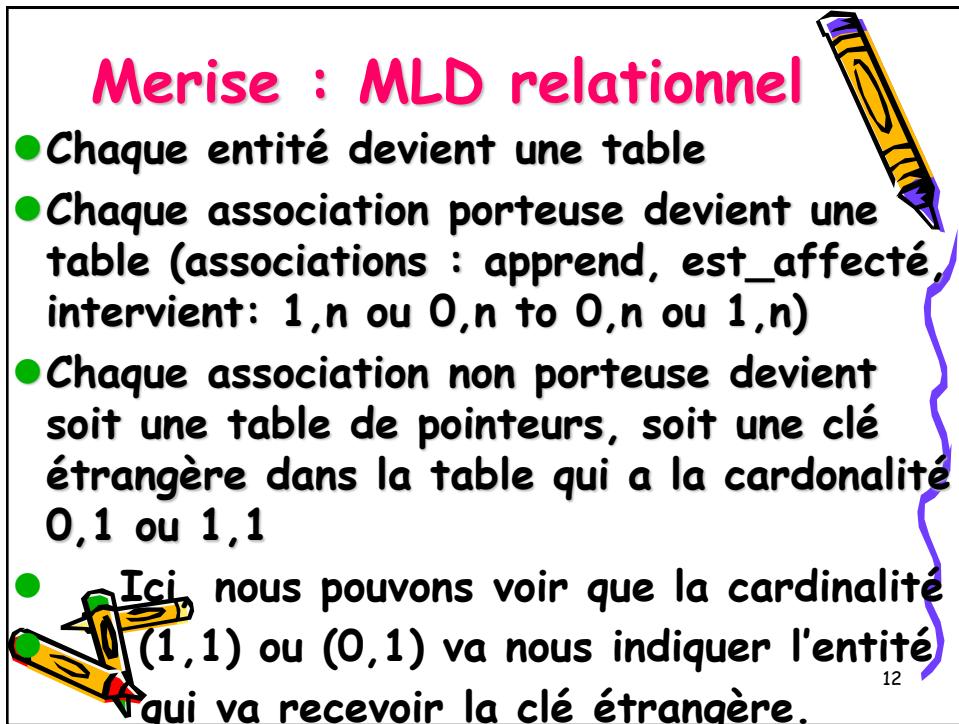
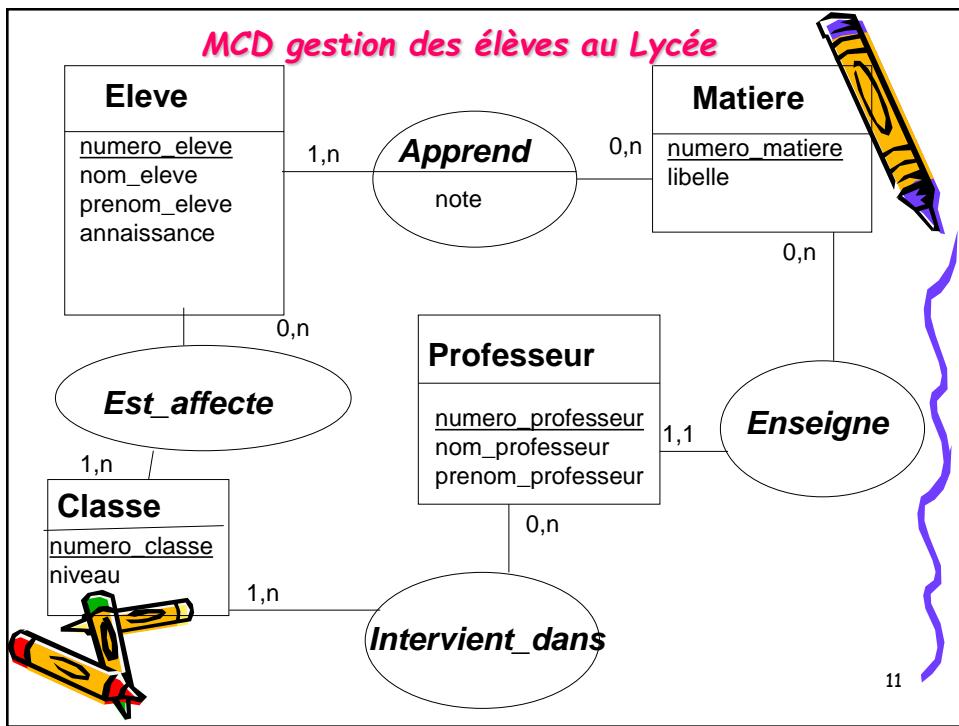
7

Merise données : MCD



8



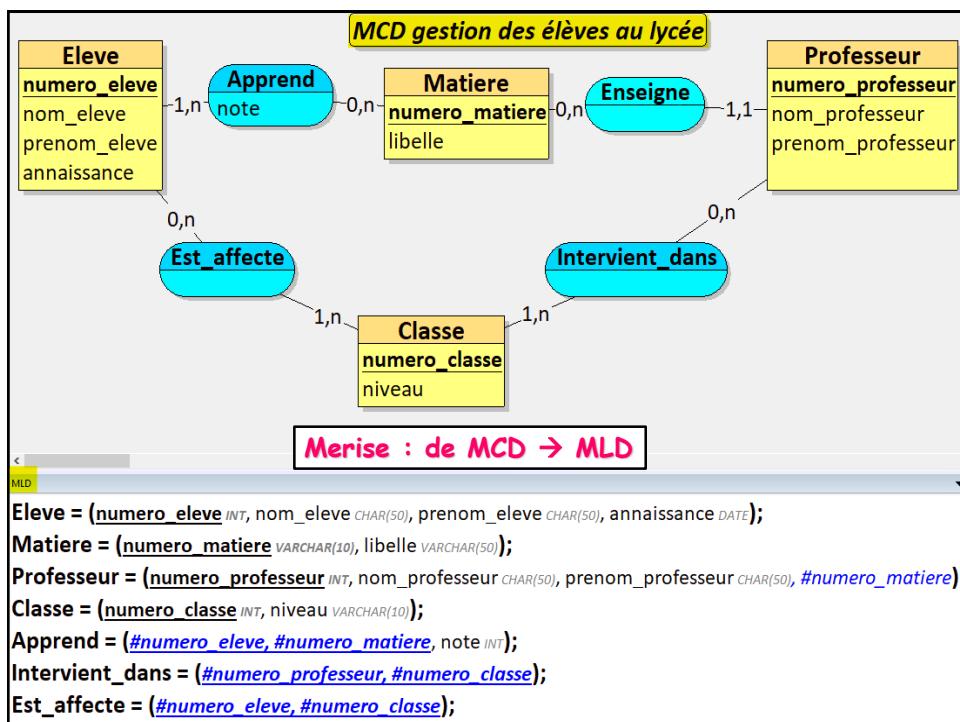


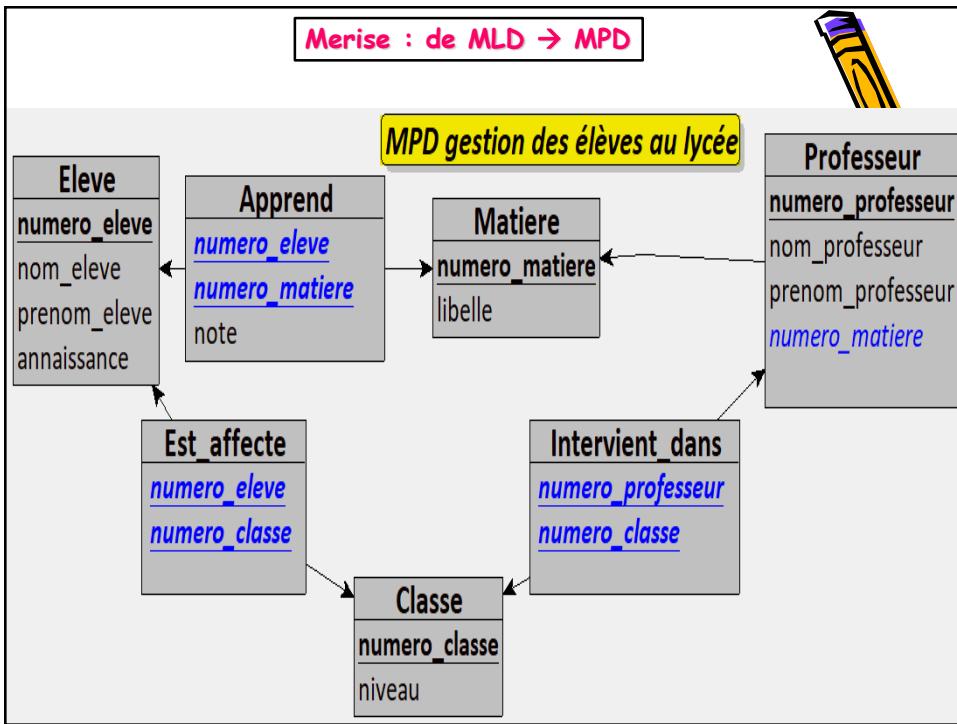
Merise : MLD relationnel

1. **Eleve** (numero_eleve, nom_eleve, prenom_eleve, annaissance)
2. **Apprend** (#numero_eleve, #numero_matiere, note)
3. **Matiere** (numero_matiere, libelle)
4. **Professeur** (numero_professeur, nom_professeur, prenom_professeur, #numero_matiere)
5. **Classe** (numero_classe, niveau)
6. **Intervient_dans** (#numero_classe, #numero_professeur)
7. **Est_affecte** (#numero_eleve, #numero_classe)

Le nouvel identifiant (clé primaire) des nouvelles entités **Intervient_dans**, **Apprend**, **Est_affecte**, seront une concaténation des deux clés étrangères.

13





Normalisation de Codd : point de départ



- Liste des attributs
- Dépendances fonctionnelles :

une contrainte entre deux ensembles d'attributs dans une relation (table) d'une base de données, vise à caractériser des relations qui peuvent être décomposées sans perte d'information.

Les DF nous ont permis jusque-là de mettre en évidence une forme de redondance, que les formes normales cherchent à faire disparaître.

Mais des cas de redondance ne sont pas capturés par les DF.

- Dépendances multivaluées



17

Dépendances multivaluées



• Sur la relation *Livre(isbn, auteur, keyword)*

- Si un livre peut avoir plusieurs auteurs et plusieurs mots-clés(keyword), la relation livre possède la dépendance multivaluée
 $isbn \rightarrow auteur$

Autrement dit, à isbn fixé, on a toutes les combinaisons possibles d'auteurs et de mots-clés pour lisbn en question :

- Si (i, m_1, a_1) et (i, m_2, a_2) sont dans R
- Alors (i, m_1, a_2) et (i, m_2, a_1) sont aussi dans R

isbn	keyword	auteur
t_1	i	m_1
t_2	i	a_1
t_3	i	m_2
t_4	i	a_2



18

Normalisation : processus

Normaliser c'est regrouper la liste des attributs en relations

- ayant du sens
- cohérentes
- sans redondance



19

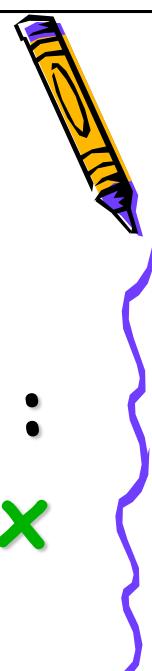


Normalisation : Point d'arrivée

On obtient une série de tables : fichiers et index



20



Normalisation :
1FN = Forme Normale

- Tout attribut de R contient:

une valeur monovaluée
et non composée



21



Normalisation : 1FN

- Exemples :

➤ prenom 1, prenom 2
➤ Adresse: numero_voie,
type voie, nom voie,
nom ville, code postal



22



Normalisation : 1FN



● R (numero_eleve,
prenoms, Adresse)
n'est pas en 1FN



23

R normalisée en 1FN

R = Relation

● R (numero_eleve,
prenom1, prenom2,
numero_voie, Type_voie,
Nom_voie, Nom_ville,
Code_postal)



24

Normalisation : 2FN

1. R est en 1FN
2. Tout attribut non clé de R dépend de la totalité de la clé

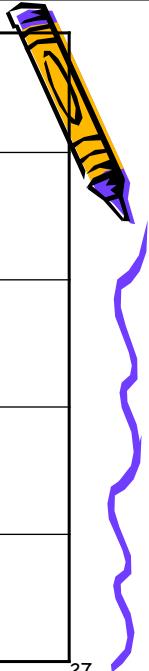
25

Normalisation : 2FN

- R (numero_eleve,
numero_matiere, note,
nom_eleve)

Le nom_eleve ne depend fonctionnellement que du
numero_eleve

26



27

100	005	12	Dupont
100	001	16	Dupont
100	002	08	Dupont
100	003	10	Dupont
200	004	12	Martin




28

Normalisation : 2FN

- R n'est pas en 2FN
- On normalise :

R1 (numero_eleve, nom_eleve)

R2 (numero_eleve, numero_matiere, note)



Normalisation : 3FN



1. R est en 2FN

2. Tout attribut de R ne dépend pas de la clé par transitivité



$X \rightarrow Y - Y \rightarrow Z$ transitivité $X \rightarrow Z$

Normalisation : 3FN



**• R (numero_professeur,
nom_professeur,
numero_matiere,
libelle_matiere)**

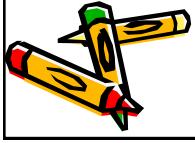
libelle_matiere dépend fonctionnellement de numero_professeur par transitivité, à travers le numero_matiere






100	Duval	001	Anglais
101	Farenc	001	Anglais
102	Gervais	001	Anglais
103	Justin	001	Anglais
104	Loliée	002	Espagnol

31

On normalise en 3FN

R1 (numero_professeur,
Nom_professeur,
numero_matiere)

R2 (numero_matiere,
libelle_matiere)

32

Normalisation 4FN et 5FN



- On traite les DM
Dépendances multivaluées



33

4FN : la relation ne contient qu'une seule DM

R (Etudiant, Cours, Sport)

avec

Etudiant DM Cours

Etudiant DM Sport



R n'est pas en 4FN

34




A table showing student activities:

Pierre	Maths	Karaté
Pierre	Anglais	Karaté
Pierre	Physique	Karaté
Jacques	Philo	Judo
Jacques	Philo	Natation
Jacques	Philo	Marathon

35




Normalisation de R en 4FN

R1 (Etudiant, Cours)

R2 (Etudiant, Sport)

36

R1

Pierre	Maths
Pierre	Anglais
Pierre	Physique
Jacques	Philo



37

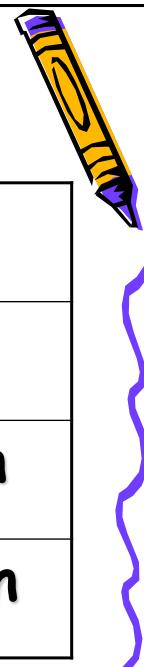


R2

Pierre	Karaté
Jacques	Judo
Jacques	Natation
Jacques	Marathon



38



5FN : DM de jointure

**R (Revendeur, Marque,
Type_produit)**

avec

Revendeur DM Marque

Revendeur DM Type_produit

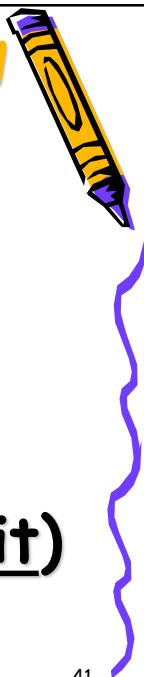
Marque DM Type_produit

39

Pierre	Renault	Voiture
Pierre	Renault	Camion
Pierre	Volvo	Voiture
Paul	Renault	Camion
Jacques	Renault	Voiture
Jacques	Volvo	Voiture

40

Normalisation de R en 5FN



R1 (Revendeur, Marque)

**R2 (Revendeur,
Type_produit)**

R3 (Marque, Type_produit)



41

Pierre	Renault
Pierre	Volvo
Jacques	Renault
Paul	Renault
Paul	Volvo



R1

42

Pierre	Voiture
Pierre	Camion
Paul	Camion
Jacques	Voiture

R2



43



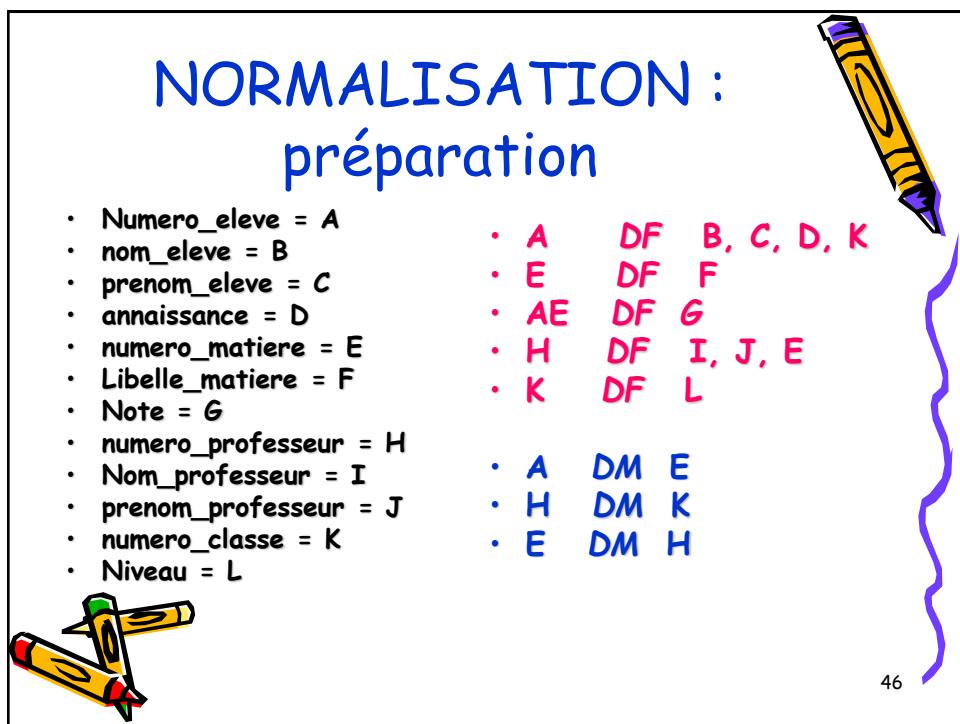
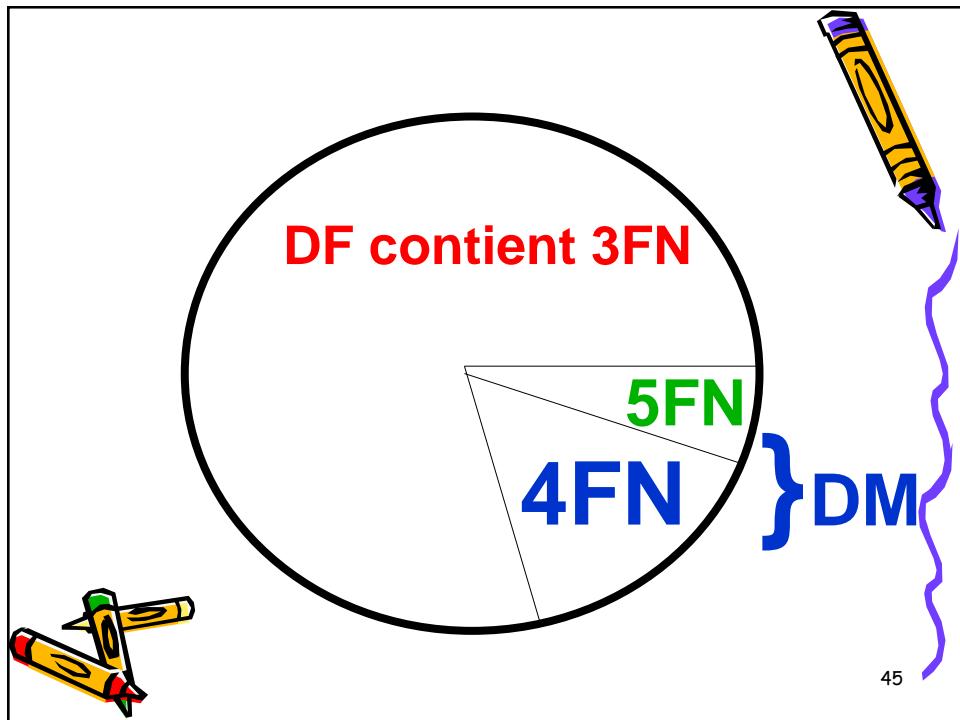
Renault	Voiture
Renault	Camion
Volvo	Voiture

R3



44





NORMALISATION : exécution

- A DF B, C, D, K
- E DF F
- AE DF G
- H DF I, J, E
- K DF L
- A DM E
- H DM K
- E DM H



1) DF

- Elève (A, B, C, D, K)
- matière (E, F)
- Apprend (A, E, G)
- Professeur (H, I, J, E)
- Classe (K, L)

2) DM

- Intervient dans (H, K)

47

Création & manipulation des BDD : requêtes SQL

- « Structured Qwery Language » 1986

- Opérateurs de l'algèbre relationnelle



48

« LDD » : langage de définition des données



```
CREATE TABLE eleve  
(numero_eleve NUM(6) [ UNIQUE] [NOT  
NULL] [PRIMARY KEY],  
nom_eleve CHAR(30)[NOT NULL],  
prenom_eleve CHAR(20) [NOT NULL],  
annaissance NUM(4)...)
```



49

« LDD » langage de définition des données



```
CREATE TABLE apprend  
(numero_eleve [ UNIQUE] [NOT NULL]  
[FOREIGN KEY],  
numero_matiere [ UNIQUE] [NOT NULL]  
[FOREIGN KEY],  
note NUM(2,2) [VALEUR: 0,00 à 20,00])
```



50

INDEX : structures d'accès

**CREATE [UNIQUE] INDEX i1
ON eleve (numeroélève)**

**CREATE INDEX i2 ON eleve
(annaisance)**

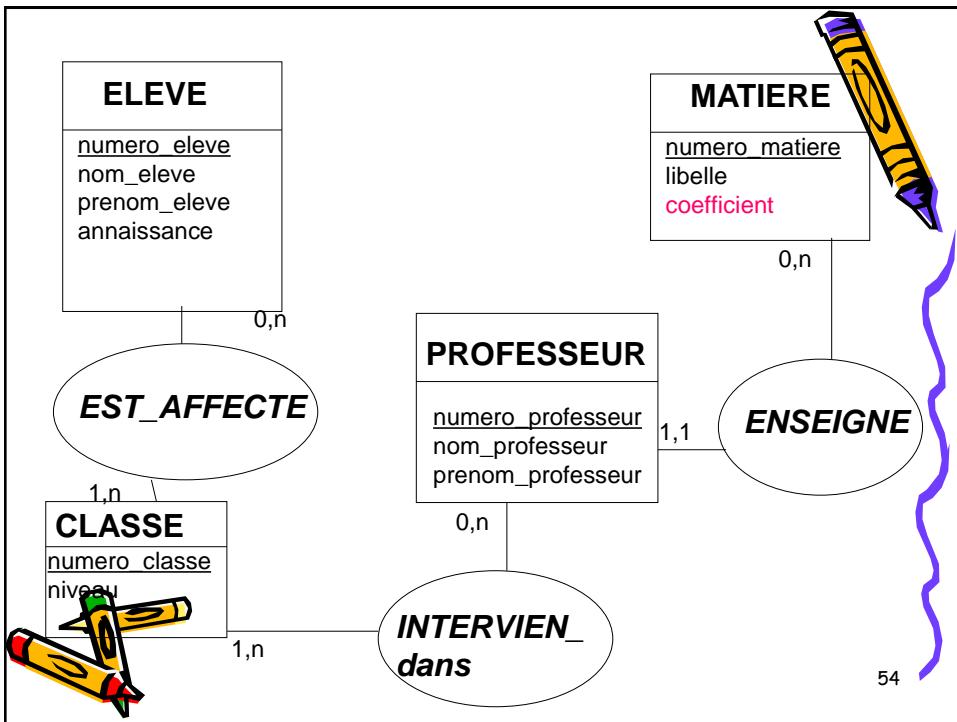
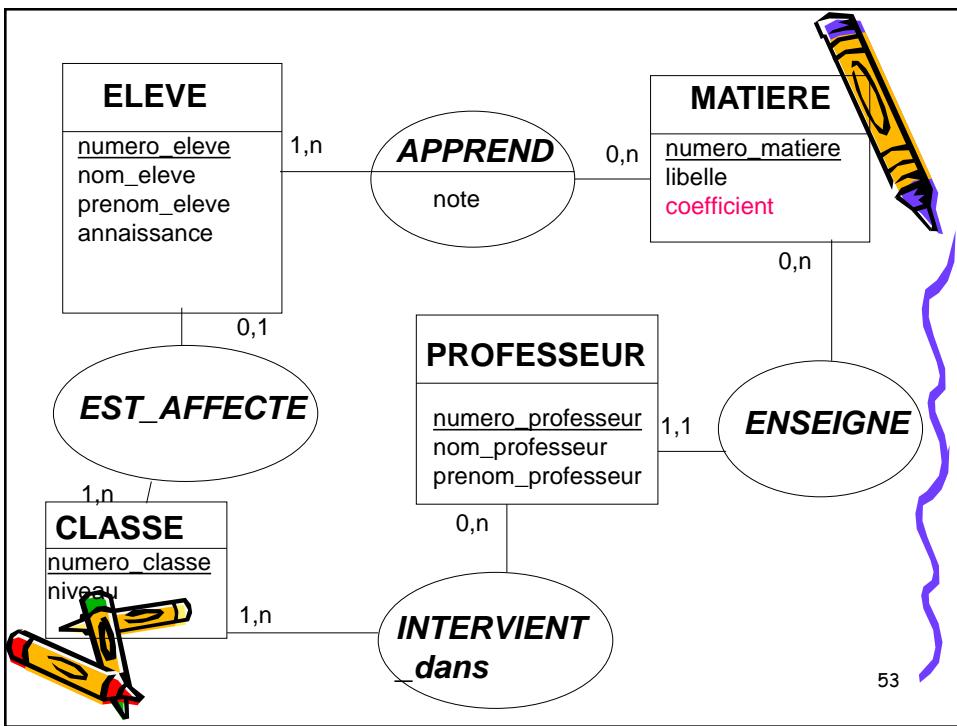
51

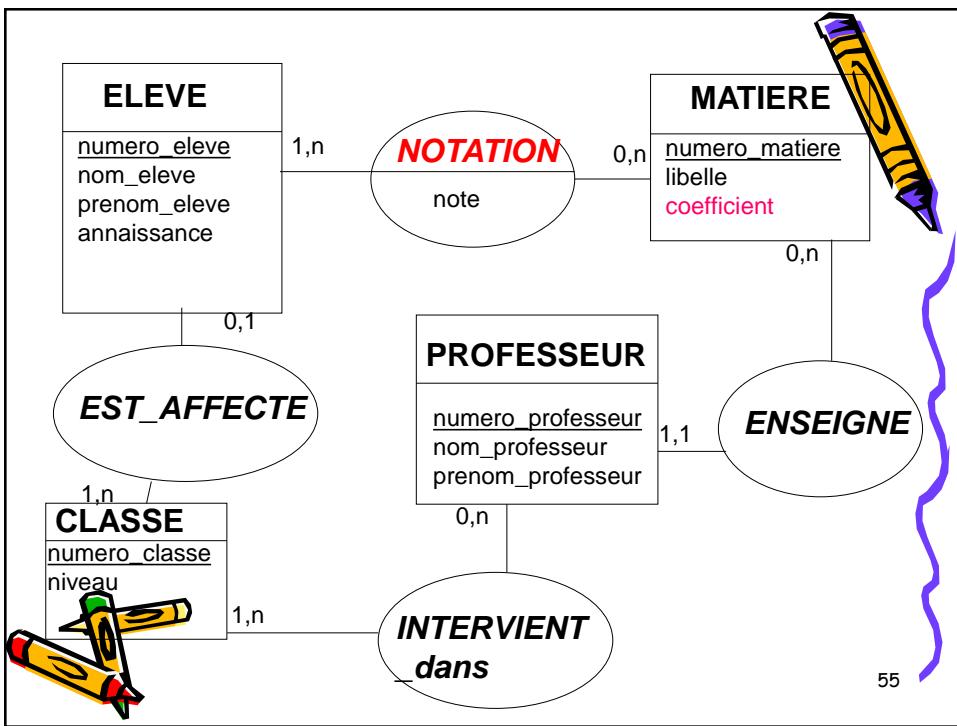
ALTER TABLE matière
ADD COLUMN coefficient
NUM(1)

DROP TABLE apprend

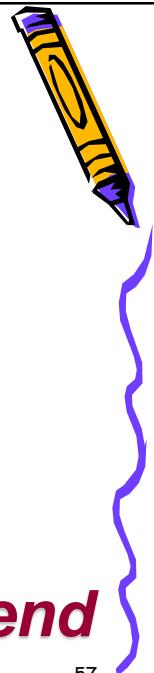
RENAME apprend to notation

52





**GRANT INSERT
ON apprend
TO professeur**



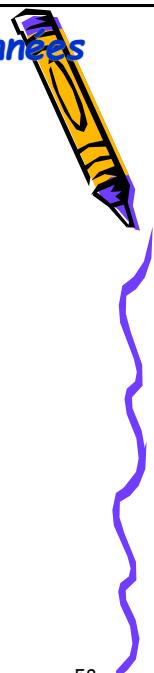
**GRANT SELECT
ON apprend
TO PUBLIC**

**REVOKE ALL ON apprend
FROM PUBLIC**

57

« LMD » : langage de manipulation des données

SELECT



INSERT

UPDATE



DELETE

58

SELECT

SELECT

nom_eleve, prenom_eleve

FROM eleve

WHERE annaissance = 1994

ORDER BY nom_eleve



59

SELECT

**SELECT nom_eleve, prenom_eleve,
note**

FROM eleve, apprend

WHERE annaissance = 1994

AND eleve.numero_eleve=apprend.numero_eleve

AND note NOT NUL

ORDER BY nom_eleve ASC



60

INSERT

**INSERT INTO eleve
(numero_eleve, numero_classe,
nom_eleve, prenom_eleve,
annaissance)**

**VALUES (105, C217, Durant,
Alexis, 2000)**



61

UPDATE

**UPDATE apprend
SET note = 14
WHERE numero_eleve = 106
AND numero_matiere = 5**



62

DELETE

DELETE FROM
eleve
WHERE
numero_eleve = 78



63



Les agrégats

SUM

GROUP BY pour regrouper les données
appartenant à la même
entité(obligatoire dans **SELECT**).

SELECT numero_client, SUM(montant)

FROM commande

GROUP BY numero_client



64



AVG

2 requêtes, Est-ce possible de les regroupées !

```
SELECT numero_eleve,nom_eleve  
FROM eleve
```

```
SELECT numero_eleve, AVG(note)  
FROM apprend  
GROUP BY numero_eleve  
OU :  
SELECT numero_eleve,nom_eleve, AVG(note)  
FROM eleve, apprend  
WHERE eleve.numero_eleve=apprend.numero_eleve  
GROUP BY AVG(note) DESC
```



65



MIN ou MAX

```
SELECT MAX(note)  
FROM apprend  
WHERE numero_eleve = 105
```



66



COUNT

SELECT COUNT(*)

FROM eleve

**SELECT COUNT(*) FROM
apprend**

WHERE numero_eleve = 102



67



Opérateurs ensemblistes

EXISTS

UNION

INTERSECT

MINUS



68



EXISTS

la commande EXISTS s'utilise dans une clause conditionnelle pour savoir s'il y a une présence ou non de lignes lors de l'utilisation d'une sous-requête.

SELECT nom_eleve FROM eleve

WHERE EXISTS

(SELECT * FROM apprend

WHERE note = 20

**AND eleve.numero_eleve =
apprend.numero_eleve)**

69

NOT EXISTS

SELECT nom_eleve FROM eleve

WHERE NOT EXISTS

(SELECT * FROM matiere,apprend

**AND eleve.numero_eleve =
apprend.numero_eleve**

**AND matiere.code_matiere =
apprend.code_matiere)**

70

UNION

Permet de concaténer les résultats de 2 requêtes ou plus. Pour l'utiliser il est nécessaire que chacune des requêtes à concaténer retournes le même nombre de colonnes, avec les mêmes types de données et dans le même ordre.

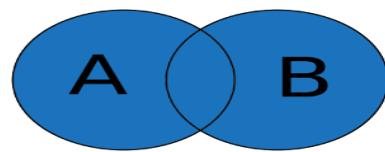
apprend(numero_eleve, code_matiere)

pratique(numero_eleve, code_sport)

SELECT numero_eleve FROM apprend

UNION

SELECT numero_eleve FROM pratique



71

INTERSECT

Permet d'obtenir l'intersection des résultats de 2 requêtes. Cette commande permet donc de récupérer les enregistrements communs à 2 requêtes. Cela peut s'avérer utile lorsqu'il faut trouver s'il y a des données similaires sur 2 tables distinctes.

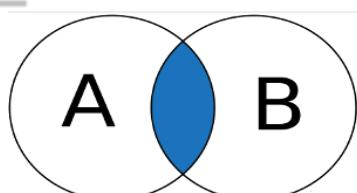
SELECT numero_eleve

FROM apprend

INTERSECT

SELECT numero_eleve

FROM pratique



72

EXCEPT/MINUS

EXCEPT s'utilise entre 2 instructions pour récupérer les enregistrements de la première instruction sans inclure les résultats de la seconde requête. Si un même enregistrement devait être présent dans les résultats des 2 syntaxes, ils ne seront pas présent dans le résultat final.

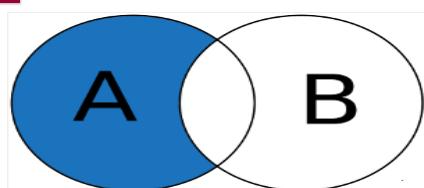
SELECT numero_eleve

FROM apprend

MINUS

SELECT numero_eleve

FROM pratique



73

VUE : table virtuelle

Non stockée (temporaire)
créée par une requête
sur une ou plusieurs tables
(permanentes)



74

- **CREATE VIEW** France_Custom

AS

```
SELECT CustomerName, ContactName
FROM Customers
WHERE Country = 'France';
```

- **SELECT * FROM** France_Custom;



75



DF : Dépendances Fonctionnelles

Exercice 1 :

Considérons le schéma de la relation suivante : R(A, B, C, D, E)

Cette relation est définie en extension par les tuples suivants :

A	B	C	D	E
a1	b2	c2	d3	e2
a1	b2	c2	d1	e1
a2	b3	c2	d1	e5
a2	b4	c5	d1	e5

I) Parmi les dépendances fonctionnelles suivantes, lesquelles s'appliquent à l'extension de R ?

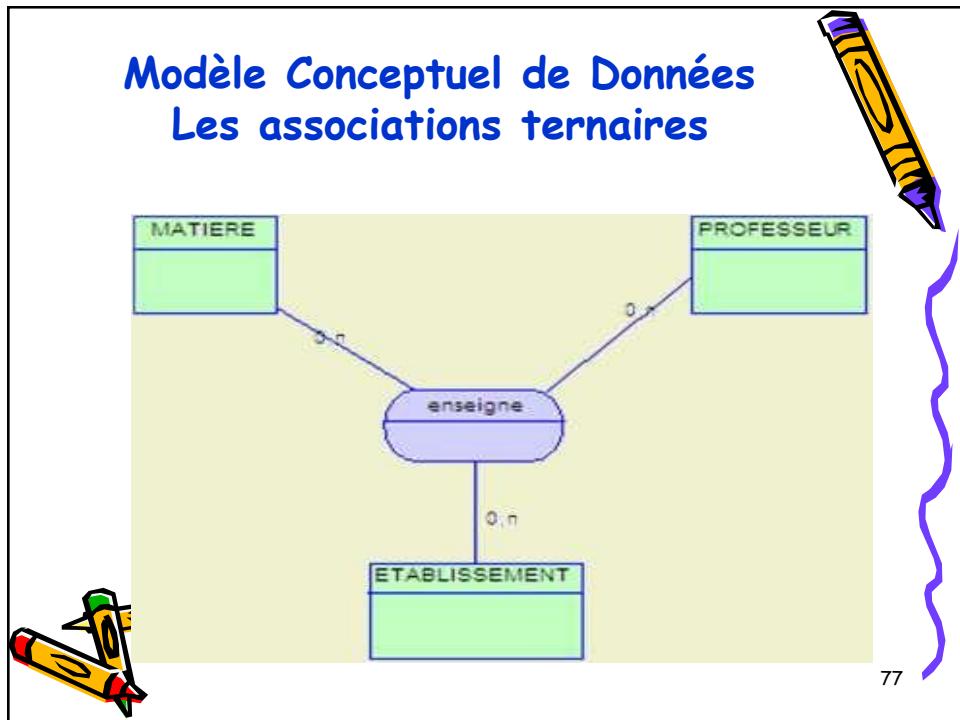
- | | | |
|---------------------------|---------------------------|--------------------------------|
| <input type="radio"/> E→D | <input type="radio"/> E→A | <input type="radio"/> {A, E}→C |
| <input type="radio"/> D→E | <input type="radio"/> B→C | <input type="radio"/> {A, E}→D |
| <input type="radio"/> C→A | <input type="radio"/> B→D | <input type="radio"/> {A, D}→E |
| <input type="radio"/> E→B | <input type="radio"/> B→A | <input type="radio"/> {A, B}→A |



76

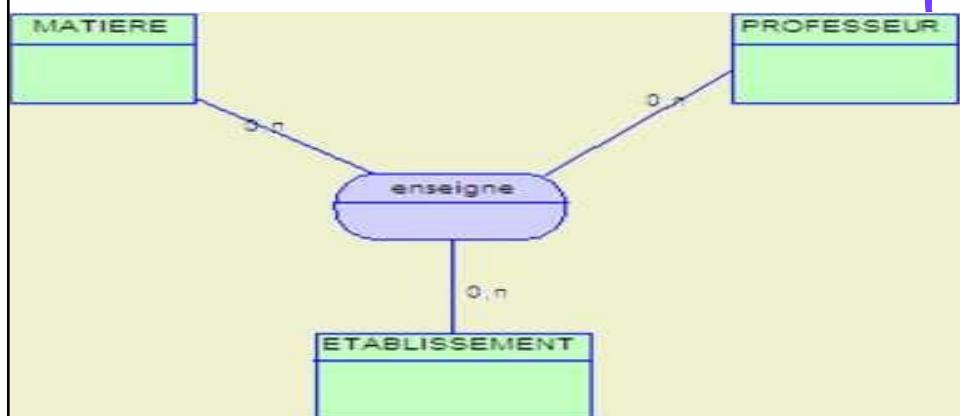


Modèle Conceptuel de Données Les associations ternaires



Intérêt d'une association ternaire

Pour savoir quelle matière est enseignée par tel professeur dans tel établissement, il faut relier les 3 entités par une association ternaire



Associations binaires ou ternaires ?

Quel serait le MCD dans cet exemple ?

Les professeurs enseignent dans des établissements dans certaines matières. Il est possible qu'un professeur enseigne des matières différentes dans les établissements de l'enseignement supérieurs.

Décomposons cet exemple en phrases simples :

Un professeur peut enseigner dans plusieurs établissements supérieurs

Un établissement peut avoir plusieurs professeurs

Une matière est enseignée par plusieurs professeurs

Un professeur peut enseigner plusieurs matières

Un établissement propose plusieurs matières

Une matière est proposée dans plusieurs établissements



79

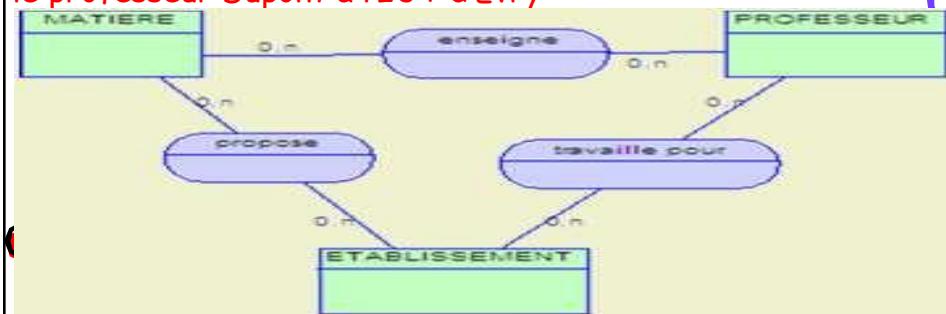
Associations binaires ? Problème...

Avec cette modélisation, nous savons qu'un professeur enseigne certaines matières et travaillent dans certains établissements.

Le professeur Dupont enseigne les mathématiques et l'informatique.

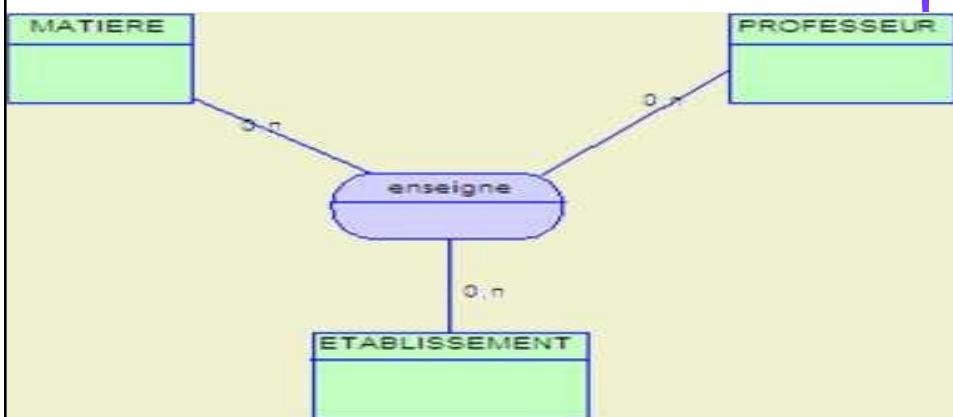
Le professeur Dupont travaille à l'université D'Evry et à l'IUT

Mais nous ne savons pas quelles matières sont enseignées par le professeur Dupont à l'IUT d'Evry



Intérêt d'une association ternaire

Pour savoir quelle matière est enseignée par tel professeur dans tel établissement, il faut relier les 3 entités par une association ternaire



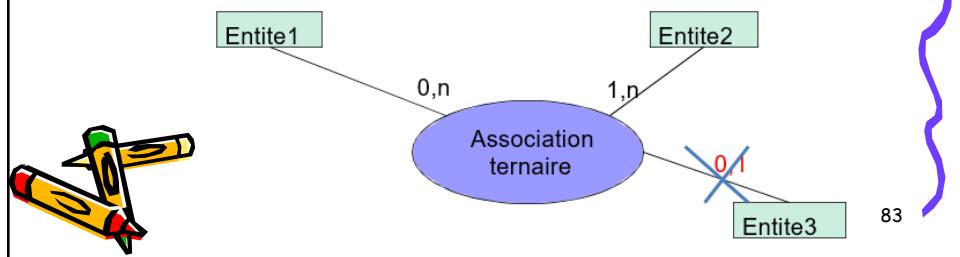
Associations n-aires

- Il peut exister des quaternaires, voire des n-aires mais c'est extrêmement rare.
- Il y a peut être une erreur de conception si le nombre d'entités reliées (arité) est supérieur ou égale à 4.



Association ternaire et cardinalité maximale

- Dans une association ternaire,
 - Les cardinalités minimales peuvent être à 0 ou 1 selon le contexte (participation facultative ou obligatoire de l'entité à l'association).
 - Toutes les cardinalités maximales sont obligatoirement à n.
- La cardinalité maximale à n représente le fait qu'une occurrence de l'entité peut participer n fois à l'association.
Si l'une des cardinalités maximales n'est pas à n, il y a une erreur de conception



Associations ternaires : limites

- Une association ternaire met en jeu les 3 entités.
Dans l'association ternaire « enseigne », on ne peut pas mémoriser des professeurs qui enseignent des matières en dehors d'un établissement car dans ce cas, l'entité « établissement » ne serait pas concernée par la relation.
Par contre, on peut mémoriser des professeurs qui n'enseignent pas, car la cardinalité minimale est à zéro.



Association ternaire et agrégat

Si un professeur peut enseigner plusieurs matières mais si un professeur ne peut enseigner qu'une seule matière par établissement, nous sommes en présence d'une "fausse ternaire".

Ce cas particulier s'appelle un agrégat et sera vu plus en détail ultérieurement.

Il est possible de le traiter comme une ternaire dans le MCD mais cela aura une répercussion sur la clé de la nouvelle table dans le MLD.



85

Association ternaire et réflexive

L'étude concerne un seul établissement.

Un professeur peut enseigner plusieurs matières à d'autres professeurs.

Il y a une association ternaire avec :

- 1 lien vers l'entité Matière
- 1 lien vers l'entité Professeur (rôle de professeur-formateur)
- 1 autre lien vers l'entité Professeur (rôle de professeur-formé)



86

Interprétation d'un MCD : cas Sport

Nous allons proposer plusieurs cas qui modélisent un même contexte avec 3 entités : Adhérent, Centre, Sport.

Chaque modélisation a une signification différente.
Indiquez en français ce que modélise chaque schéma.

Essayez de voir les limites de chaque schéma.

Il est fortement conseillé d'essayer de rédiger la signification de chaque schéma avant de regarder la correction



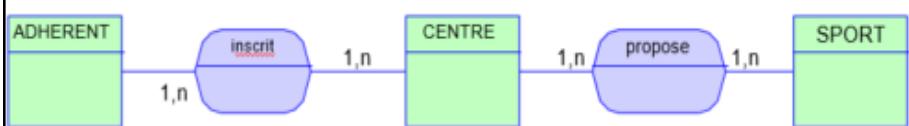
87

Interprétation d'un MCD - cas 1

Des adhérents sont inscrits dans des centres.

Les centres proposent des sports.

On ne sait pas quel(s) sport(s) pratique un adhérent.



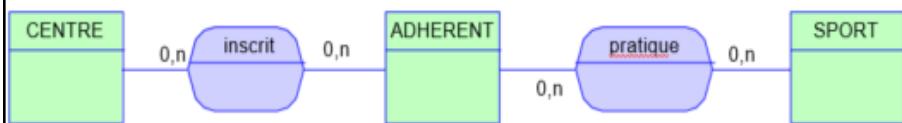
88

Interprétation d'un MCD - cas 2

Des adhérents sont inscrits dans des centres.

Les adhérents pratiquent certains sports.

On ne sait pas quel(s) sport(s) propose un centre.



89

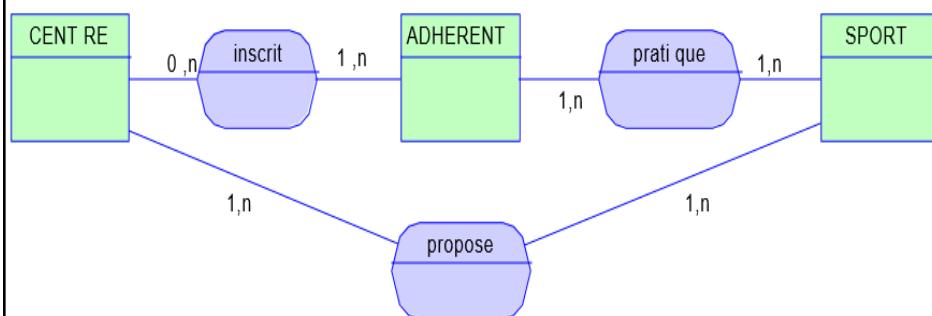
Interprétation d'un MCD - cas 3

Des adhérents sont inscrits dans des centres.

Les adhérents pratiquent des sports.

Les centres proposent des sports.

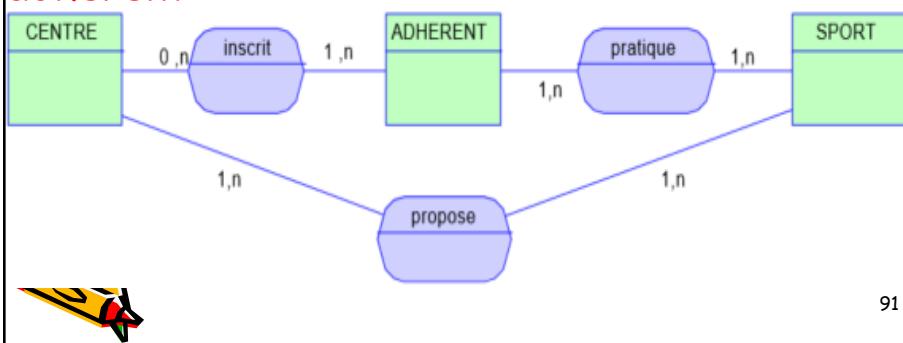
On ne sait pas dans quel(s) centre(s) un adhérent pratique un sport donné.



Interprétation d'un MCD - cas 4

Des adhérents pratiquent des sports dans des centres.

On ne sait pas quel sport est proposé par un centre si ce sport n'est pratiqué par aucun adhérent

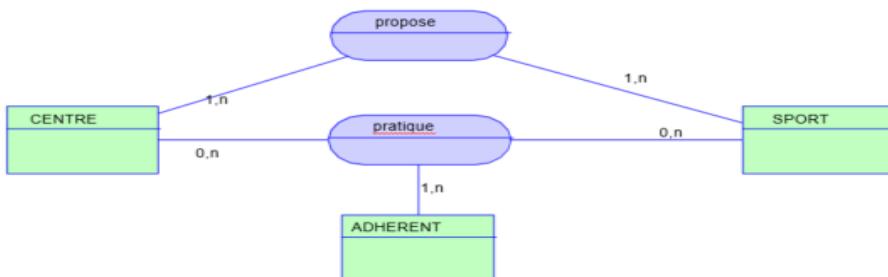


Interprétation d'un MCD - cas 5

Des adhérents sont inscrits dans des centres pour pratiquer des sports.

Les centres proposent des sports à pratiquer.

On ne connaît pas les sports pratiqués par les adhérents hors de ces centres (ex : sports individuels).

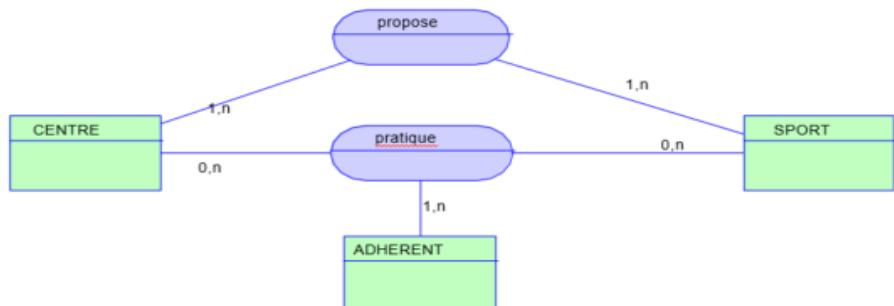


Interprétation d'un MCD - cas 6

Des adhérents sont inscrits dans des centres pour pratiquer des sports.

Les centres proposent des sports à pratiquer.

Proposez une modification pour connaître les sports pratiqués par les adhérents hors des centres.



MCD

```

classDiagram
    class Centre {
        numCentre
        nomCentre
    }
    class Sport {
        id_Sport
        nomSport
        nbLicence
    }
    class Adherent {
        numAdherent
        nomAdherent
        prenomAdherent
        telAdherent
    }
    class propose {
        id_Sport
        numCentre
    }
    class pratiquer {
        id_Sport
        numAdherent
        numCentre
        dateDebut
    }

    Centre "1..n" --> "0..n" propose : 
    Centre "0..n" --> "1..n" Adherent : 
    Adherent "0..n" --> "1..n" pratiquer : 
    propose "1..n" --> "0..n" Sport : 
    propose "1..n" --> "0..n" pratiquer : 
  
```

MPD

Sport = ([id_Sport INT](#), nomSport VARCHAR(50), nbLicence INT);
Centre = ([numCentre INT](#), nomCentre VARCHAR(50));
Adherent = ([numAdherent INT](#), nomAdherent VARCHAR(50), prenomAdherent VARCHAR(50), telAdherent INT);
pratiquer = ([#id_Sport](#), [#numAdherent](#), [#numCentre](#), dateDebut DATE);
propose = ([#id_Sport](#), [#numCentre](#));

```

CREATE TABLE Sport(
    id_Sport INT,
    nomSport VARCHAR(50),
    nbLicence INT,
    PRIMARY KEY(id_Sport)
);
CREATE TABLE Centre(
    numCentre INT,
    nomCentre VARCHAR(50),
    PRIMARY KEY(numCentre)
);
CREATE TABLE Adherent(
    numAdherent INT,
    nomAdherent VARCHAR(50),
    prenomAdherent VARCHAR(50),
    telAdherent INT,
    PRIMARY KEY(numAdherent)
);
CREATE TABLE pratiquer(
    id_Sport INT,
    numAdherent INT,
    numCentre INT,
    dateDebut DATE,
    PRIMARY KEY(id_Sport, numAdherent, numCentre),
    FOREIGN KEY(id_Sport) REFERENCES Sport(id_Sport),
    FOREIGN KEY(numAdherent) REFERENCES Adherent(numAdherent),
    FOREIGN KEY(numCentre) REFERENCES Centre(numCentre)
);
CREATE TABLE propose(
    id_Sport INT,
    numCentre INT,
    PRIMARY KEY(id_Sport, numCentre),
    FOREIGN KEY(id_Sport) REFERENCES Sport(id_Sport),
    FOREIGN KEY(numCentre) REFERENCES Centre(numCentre)
);
  
```

Ternaires et modèle logique de données

- Les règles définies pour les binaires de type plusieurs à plusieurs s'appliquent aux ternaires...
- Les cardinalités maximales sont à n sur chaque lien de l'association dans le MCD, donc l'association est transformée en une nouvelle table dans le MLD.
- La clé primaire de cette nouvelle table est **une clé primaire composée des clés primaires des tables liées** (donc clé composée de 3 éléments pour une ternaire).
- Chaque élément de la clé primaire est une clé étrangère qui référence la clé primaire de chaque table associée.
- Si l'association porte des propriétés, ces propriétés deviennent des attributs de la table résultante

Passage au MLD d'une ternaire

Table Centre(numCentre, nomCentre)

numCentre: clé primaire de la table Centre

Table Sport(idSport, nomSport, nbLicences)

idSport: clé primaire de la table sport

Table Adherent(numAdherent, nomAdherent, prenomAdherent, telAdherent)

numAdherent: clé primaire de la table Adherent

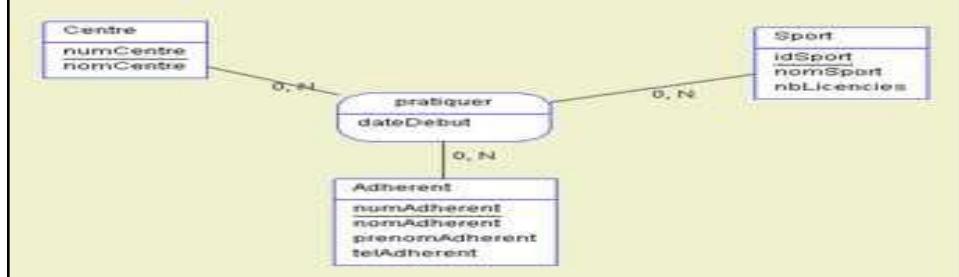
Table Pratique(numCentre, idSport, numAdherent, dateDebut)

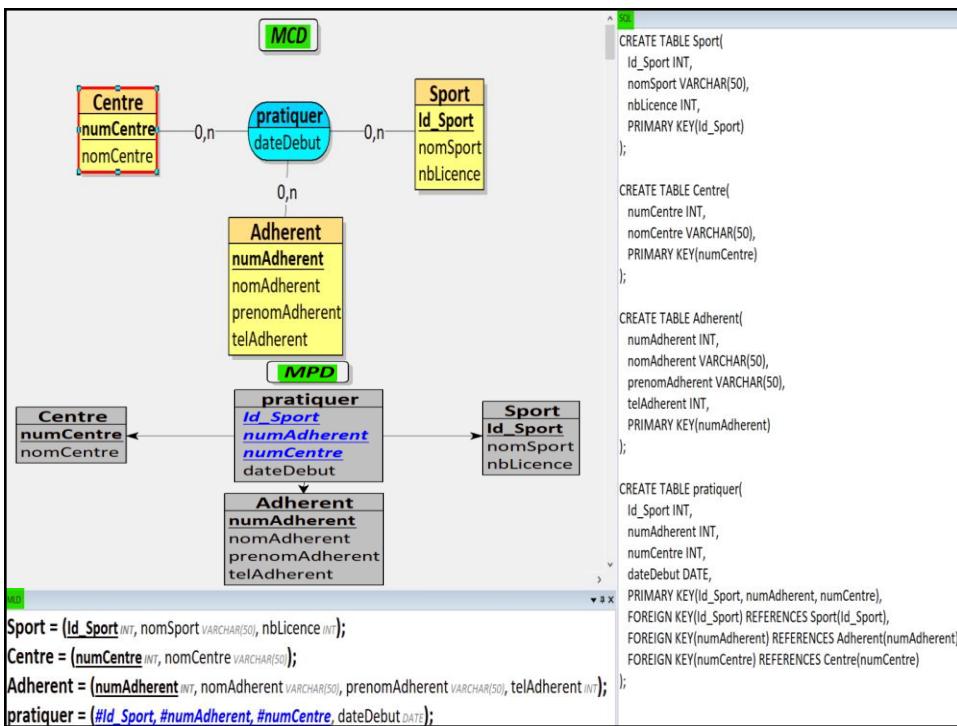
numCentre, idSport, numAdherent: clé primaire composée de la table Pratique

numCentre clé étrangère qui référence numCentre de la table Centre

idSport clé étrangère qui référence idSport de la table Sport

numAdherent clé étrangère qui référence numAdherent de la table Adherent



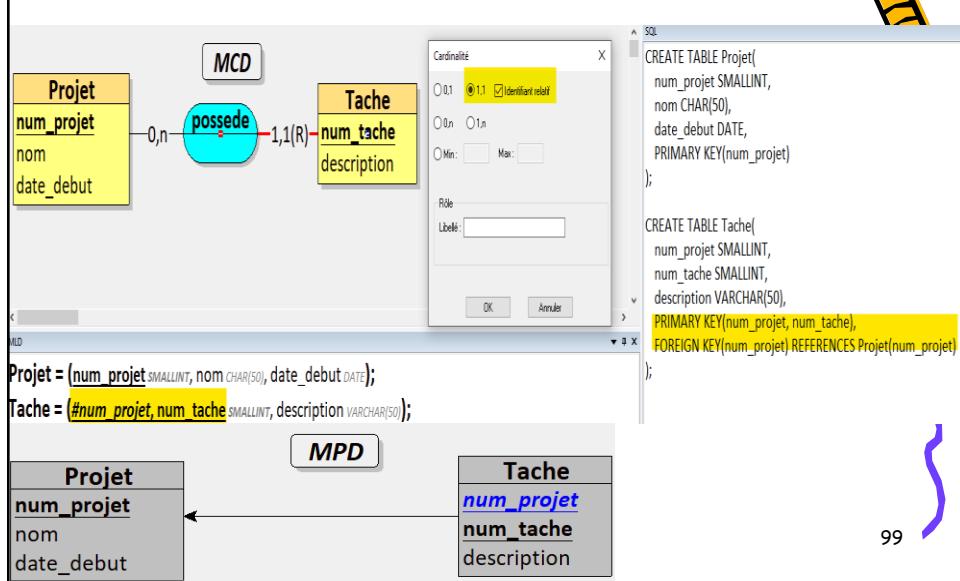


Entité de type faible

- Certaines entités dites "faibles" n'existent qu'en référence à d'autres entités dites "identifiantes". L'entité identifiante est appelé "identifiant étranger" et l'association qui les unit "association identifiante".
- Une entité de type faible, également appelée entité non identifiée, possède une clé locale (appelée identifiant relatif) qui permet d'identifier une de ses occurrences parmi l'ensemble des occurrences associées à une occurrence de l'entité identifiante. La clé complète d'une entité faible est la concaténation de la clé de l'entité identifiante et de sa clé locale.
- Le repérage des entités de type faible est très important dans le processus de modélisation, il permet de réfléchir à la meilleure façon d'identifier de façon unique les entités et donc de trouver les meilleures clés. Notons de plus que le repérage d'entités faibles aura une influence importante sur le modèle relationnel résultant.

Exemple Entité faible

L'entité Tache est complètement dépendante de l'entité Projet et sa clé locale (num_tache) n'est pas suffisante à l'identifier de façon absolue



99

Transformation des associations 1:1

- Il existe deux solutions pour transformer une association 1:1 :
- Avec deux relations : on traite une association 1:1 comme un cas particulier d'association 1:n, puis l'on ajoute une contrainte UNIQUE sur la clé étrangère migrée pour limiter la cardinalité maximale à 1 ;
- Avec une seule relation :

On fusionne les deux entités en une seule relation :



Transformation des associations Avec deux relations

