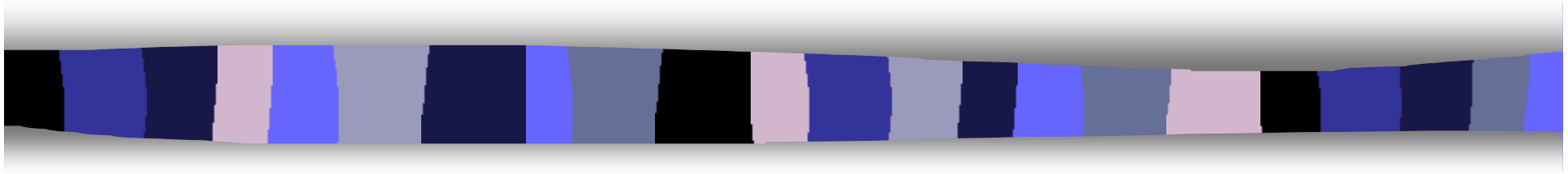
Base de données



Chapitre 1
Introduction

Plan du cours

- **Introduction**
- Modèle relationnel
- SQL
- Conception

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990

- Petite base de données avec 1 seule table (appelons la *professeur*)
  - Conserver les données
  - Supporter des opérations sur les données

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990

- Sélection

SELECT NOM, FONCTION
FROM PROFESSEUR
WHERE AN_ENT > 1992

NOM	FONCTION
Gilles	Prof_adj
Myriam	Prof_agr
Claudine	Prof_adj

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990

- Insertion

```
INSERT INTO
PROFESSEUR   (MAT, NOM, FONCTION, COURS, AN_ENT)
VALUES       (66231, 'Jian', 'Prof_adj', 'MRT2323', 1996)
```

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990
66231	Jian	Prof_adj	MRT2323	1996

- Insertion

```
INSERT INTO
PROFESSEUR   (MAT, NOM, FONCTION, COURS, AN_ENT)
VALUES       (66231, 'Jian', 'Prof_adj', 'MRT2323', 1996)
```

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990
66231	Jian	Prof_adj	MRT2323	1996

- Mise à jour

```
UPDATE PROFESSEUR
SET COURS = 'MRT2325'
WHERE MAT = 66231
```

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990
66231	Jian	Prof_adj	MRT2325	1996

- Mise à jour

```
UPDATE PROFESSEUR
SET COURS = 'MRT2325'
WHERE MAT = 66231
```

Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990
66231	Jian	Prof_adj	MRT2325	1996

- Destruction

```
DELETE FROM PROFESSEUR
WHERE MAT = 66231
```

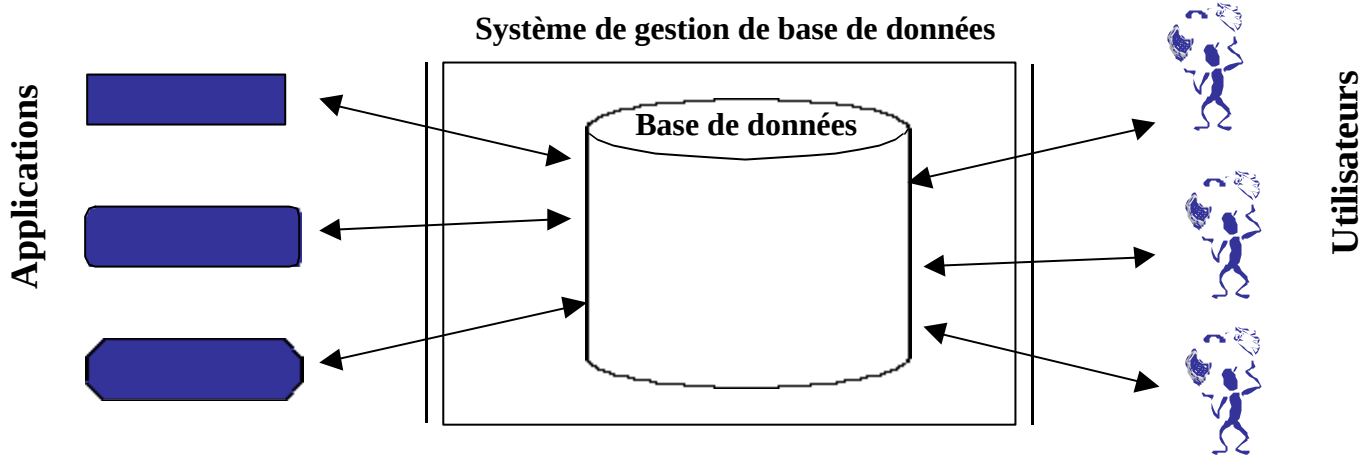
Exemple

MAT	NOM	FONCTION	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990

- Destruction

```
DELETE FROM PROFESSEUR
WHERE MAT = 66231
```

Système de gestion de bases de données

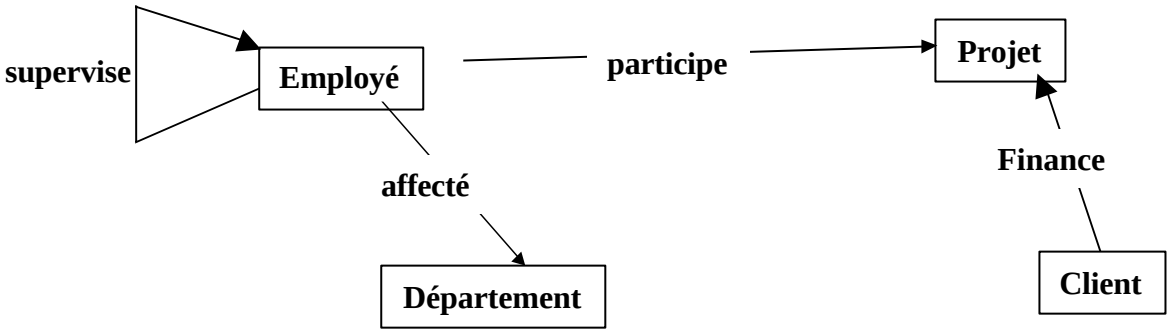


- Un système informatique de conservation de l'information nécessite :
  - Données
  - Matériel
  - Logiciel
  - Utilisateurs

Base de données

- Définitions
  - Données persistantes
    - Durée de vie dépasse celle de l'exécution d'un programme
  - Base de données
    - Collection de données persistantes utilisées par des systèmes informatiques
- Exemples
  - Organisations
    - banques, hôpitaux, université, compagnies, etc.
  - Systèmes
    - comptabilité, gestion du personnel, gestion de stock, etc.

Base de données



- Information
  - Entités
  - Relations
  - Propriétés
    - Employé
      - Mat
      - Nom
      - ...

Base de données

- Acteurs
  - Concepteur
  - Administrateur de données
  - **Administrateur de la base de données**
  - Utilisateur final
- Une même personne peut occuper plusieurs rôles
- Un même rôle peut être occupé par plusieurs personnes

Base de données

- Avantages
  - Réduit les redondances
    - Un fichier du personnel pour la paye/comptabilité
    - Un fichier du personnel pour les ressources humaines
  - Évite l'incohérence
    - Ajout d'un nouvel employé
  - Permet le partage des données
    - Entre applications existantes et futures
  - Permet l'application des normes
    - ODMG, CORBA, etc.

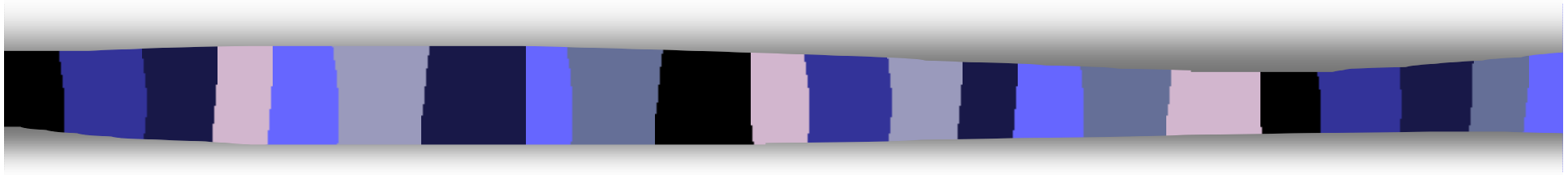
Base de données

- Avantages
  - Garantie la sécurité
    - Utilisateurs (politique)
    - Opérations (consultation, destruction, insertion)
  - Assure l'intégrité
    - Données exactes
  - Permet d'établir des priorités
    - Optimisation des accès pour certaines applications au détriment des autres.
  - Assure l'indépendance des données
    - Applications, représentation interne, etc.

Historique

- Génération 0 (années 60)
  - fichiers reliés par des pointeurs
    - IDS.I (Honeywell) et IMS.I (IBM)
    - pour les programmes de la conquête spatiale (APOLLO)
- 1ère génération (fin des années 60)
  - modèles hiérarchique et réseaux (travaux de CODASYL)
    - TOTAL, IDMS, IDS.2 et IMS.2, etc.
- 2éme génération (depuis 1970)
  - modèle relationnel
    - SGBD commercialisés à partir de 1980
    - ORACLE, SYSBASE, DB2, etc.
- 3ème génération (début des années 80)
  - extension du relationnel (Oracle 10i, DB2 Universal Database, etc.)
  - à objets (ObjectStore, GemStone, O2, etc.)
- 4ème génération … Internet, les informations non structurées, le multimédia, la découverte des données, XML

Base de données



Chapitre 2
Le modèle relationnel

Types de modèles

- Modèles conceptuels
  - Entités-Relations
  - UML
  - …
- Modèles de bases de données
  - Réseau
  - Hiérarchique
  - Relationnel
  - À objets
  - …

Plan du cours

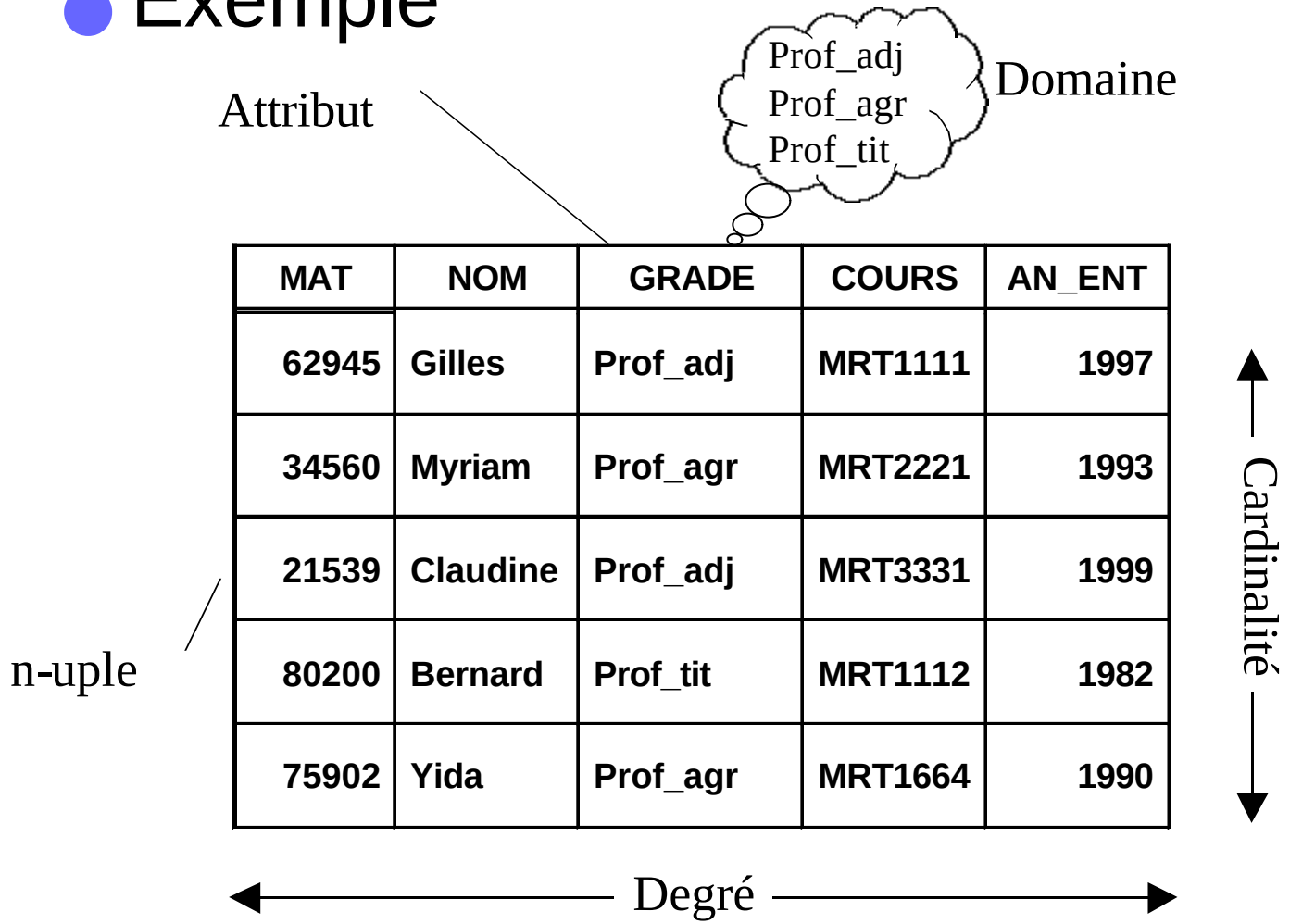
- Introduction
- **Modèle relationnel**
- SQL
- Conception

Concepts de base

- Attribut
  - Colonne d ’une relation caractérisée par un nom
- Relation
  - Degré
    - nombre d ’attributs
  - Cardinalité
    - nombre de n-uplets

Concepts de base

- Exemple



MAT	NOM	GRADE	COURS	AN_ENT
62945	Gilles	Prof_adj	MRT1111	1997
34560	Myriam	Prof_agr	MRT2221	1993
21539	Claudine	Prof_adj	MRT3331	1999
80200	Bernard	Prof_tit	MRT1112	1982
75902	Yida	Prof_agr	MRT1664	1990

Concepts de base

- Relation vs. Table
  - Relation : définition d ’un objet abstrait (voir plus haut)
  - Table : représentation concrète de cet objet abstrait
  - Souvent confondues

Intégrité des données

- Introduction
  - A tout instant la base représente une partie du monde réel à travers une configuration particulière des valeurs des données
  - Certaines configurations des valeurs n ’ont pas de sens
    - exemple age d ’une personne = -35
  - Intégration de règles d ’intégrité pour informer le SGBD de certaines contraintes du monde réel

Intégrité des données

- Exemple de contraintes d ’intégrité
  - Matricule doit être sous la forme nnnnn
  - La valeur de GRADE doit appartenir à l ’ensemble {Prof_adj, Prof_agr, Prof_tit}
  - L ’année de recrutement doit être supérieure à 1920 (ou pourquoi pas 1878)
  - L ’année de recrutement doit être inférieure ou égale à l ’année courante
  - Tous les professeurs recrutés avant 1994 doivent être au moins agrégés
- Contraintes spécifiques à une base de donnés particulière

Intégrité des données

- Propriétés d ’intégrité générale
  - clés candidates et clés primaires
  - clés étrangères
- Clés candidates
  - Ensemble minimal d ’attributs dont la connaissance des valeurs permet d ’identifier un n-uplet unique de la relation considérée
  - Chaque relation contient au moins une clé candidate

Intégrité des données

- Clés candidates
  - Utilité
    - mécanisme d ’adressage au niveau des n-uplets d ’une table
    - Exemple

    Professeur WHERE MAT = 34560, produit un seul n-uplet en résultat

    Professeur WHERE AN_ENT = 1994, va produire un nombre imprévisible de n-uplets en résultat

Intégrité des données

- Clé primaire
  - Une clé choisie parmi l ’ensemble des clés candidates
  - Le choix est arbitraire
  - Peux se faire selon des heuristiques
    - Plus simple par exemple
  - Exemple
    - Clés candidates MAT et NOM - COURS
    - Clé primaires MAT

Intégrité des données

- Clé primaire
  - Exemple

  Etudiant (Code_per, Nom, Prenom, Adresse, Date_nais, Lieu_nais, NASS)

  Quelles sont les clés candidates ?

  Quelle clé primaire choisir ?

Intégrité des données

- Clés étrangères
  - Soir R2 une relation, un sous-ensemble d ’attributs FK dans R2 est dit clé étrangère dans le cas où

  * il existe une relation R1 avec une clé candidate CK (R1 peut être R2)

  * à tout instant, chaque valeur de FK dans R2 est identique à une valeur de CK dans R1

Intégrité des données

- Clés étrangères
  - Propriétés
    - Il n ’est pas nécessaire d ’avoir pour toute valeur de CK dans R1 une valeur identique de FK dans R2
    - Une clé étrangère est composée ou simple
    - Chaque attribut d ’une clé étrangère doit être défini sur le même domaine que son équivalent dans la clé candidate correspondante

Intégrité des données

- Clés étrangères
  - Propriétés
    - Un attribut d ’une clé étrangère peut faire partie d ’une clé candidate de sa relation

Employe (Mat, Nom, …)

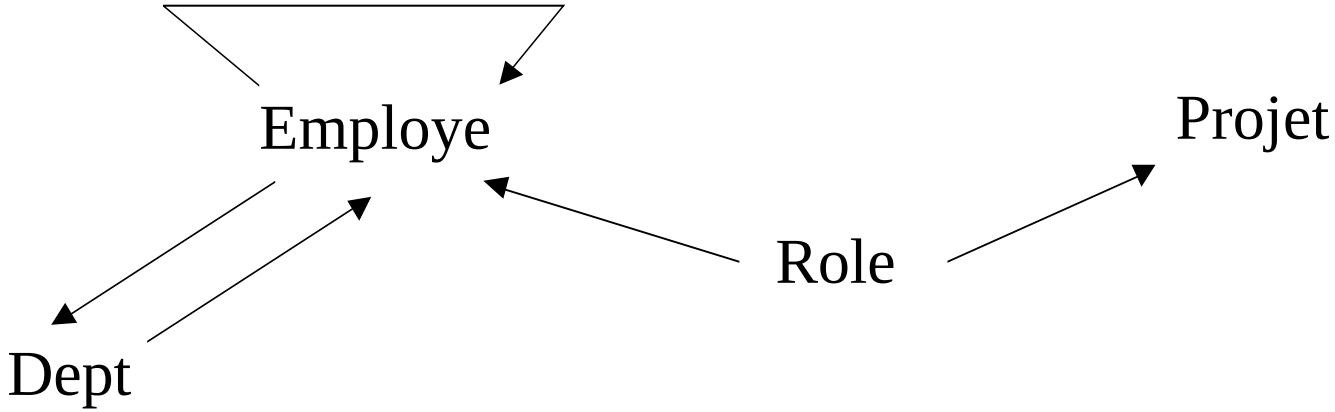
Projet (Num, Designation, …)

Role (Mat, Num, role)

- On dit que R2 référence

Intégrité des données

- Clés étrangères
  - Diagramme référentiel
    - représentation graphique des références



Intégrité des données

- Exemple

  Employe (mat# : entier, nom : caracteres, adresse : caracteres, dept : entier, sup : entier)

  Projet (num# : entier, designation : caracteres)

  Departement (dept# : entier, dir : entier, nom : caracteres)

  Role (mat# : entier, num# : entier, role_emp : caracteres)

Intégrité des données

- Clés étrangères
  - Intégrité référentielle
    - La base de données ne doit pas contenir une valeurs de clé étrangère non « unifiable »
    - Cette contrainte est dite référentielle
    - Conséquence : Il faut créer et détruire les n-uplets dans un ordre particulier
    - Dans certain SGBD (version 92 de SQL), on peux différer la vérification d ’une contrainte d ’intégrité référentiel (restreinte vs. cascade)

Aperçu de déclaration de table en SQL

- Domaine de base
  - Entier : INTEGER
  - Décimal : DECIMAL(m,n)
  - Réel flottant : FLOAT
  - Chaîne de caractères : CHAR(n)
  - Date : DATE
  - ….
- Création d ’une table (forme simple)

  CREATE TABLE <nom-table> (<attribut> <type>, <attribut> <type>, <attribut> <type>, …)

Aperçu de définition de données en SQL

- Création d ’une table (forme simple)
  - Exemple

```
CREATE TABLE VIN (
    NV INTEGER,
    CRU CHAR(10),
    MIL INTEGER)
```

- Contraintes d ’intégrité
  - NOT NULL
  - UNIQUE ou PRIMARY KEY
  - FOREIGN KEY
  - REFERENCES
  - CHECK

Aperçu de définition de données en SQL

- Avec contraintes d ’intégrité

Exemple (SQL 86)

```
CREATE TABLE COMMANDE (
      NC INTEGER UNIQUE NOT NULL,
      NV INTEGER NOT NULL
      QUANTITE DECIMAL(6))
```

Exemple (SQL 89)

```
CREATE TABLE COMMANDE (
      NC INTEGER PRIMARY KEY,
      NV INTEGER NOT NULL REFERENCES VIN,
      QUANTITE DECIMAL(6) CHECK(QUANTITE > 0))
```

```
      NC INTEGER,
      PRIMARY KEY (NC),
      NV INTEGER NOT NULL,
       FOREIGN KEY (NV) REFERENCES VIN,
```

SQL 92

Aperçu de définition de données en SQL

- Modification du schéma d ’une table

  ALTER TABLE <table>

  ADD COLUMN<attribut> <type>, <attribut> <type>, …

  – Exemple

  ALTER TABLE COMMANDE

  ADD COLUMN DATE_COM DATE

- Suppression d ’une table

  DROP TABLE <table>

  – Exemple

  DROP TABLE COMMANDE

Aperçu de définition de données en SQL

- Exemple (SQL)

  Employe (mat# : entier, nom : caracteres, adresse : caracteres, dept : entier, sup : entier)

  Projet (num# : entier, designation : caracteres)

  Departement (dept# : entier, dir : entier, nom : caracteres)

  Role (mat# : entier, num# : entier, role_emp : caracteres)

Aperçu de définition de données en SQL, exemple

Employe (mat# : entier, nom : caracteres, adresse : caracteres, dept : entier, sup : entier)

```
CREATE TABLE EMPLOYE (
    MAT INTEGER,
    NOM CHAR(12),
    ADRESSE CHAR(25),
    DEPT INTEGER NOT NULL,
    SUP INTEGER NOT NULL,
    PRIMARY KEY (MAT),
    FOREIGN KEY (DEPT) REFERENCES DEPARTEMENT,
    FOREIGN KEY (SUP ) REFERENCES EMPLOYE (MAT));
```

Aperçu de définition de données en SQL, exemple

Projet (num# : entier, designation : caracteres)

```
CREATE TABLE PROJET (
    NUM INTEGER,
    DESIGNATION CHAR(20),
    PRIMARY KEY (NUM));
```

Aperçu de définition de données en SQL, exemple

Departement (dept# : entier, dir : entier, nom : caracteres)

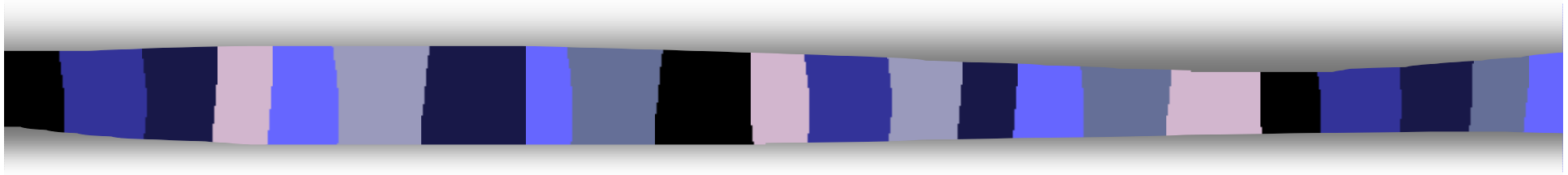
```
CREATE TABLE DEPARTEMENT (
    DEPT INTEGER,
    DIR INTEGER NOT NULL,
    DESIGNATION CHAR(20),
    PRIMARY KEY (DEPT),
    FOREIGN KEY (DIR) REFERENCES EMPLOYE (MAT));
```

Aperçu de définition de données en SQL, exemple

Role (mat# : entier, num# : entier, role_emp : caracteres)

```
CREATE TABLE ROLE (
    MAT INTEGER,
    NUM INTEGER,
    ROLE_EMP CHAR(15),
    PRIMARY KEY (MAT, NUM),
    FOREIGN KEY (MAT) REFERENCES EMPLOYE ,
    FOREIGN KEY (NUM) REFERENCES PROJET));
```

Base de données



Chapitre 3
SQL

Plan du cours

- Introduction
- Modèle relationnel
- **SQL**
- Conception

Introduction

- Permet de retrouver et de manipuler les données sans préciser les détails
- Dérivé de SEQUEL 2 (76) lui-même dérivé de SQUARE (75)
- Proposé par IBM (82 puis 87)
- Première version normalisée SQL1 (ISO89)
- Deuxième version normalisée SQL2 (ISO92)
- SQL3 en cours de normalisation
- Sert de couche basse aux L4G (par exemple Access)

Définition des données

- Domaine
  - Définition de domaines peu utilisée
  - Définition à partir des types de bases

ANSI SQL	Oracle
CHARACTER(n), CHAR(n)	CHAR(n)
CHARACTER VARYING(n), CHAR VARYING(n)	VARCHAR(n)
NUMERIC(p,s), DECIMAL(p,s), DEC(p,s)[1]	NUMBER(p,s)
INTEGER, INT, SMALLINT	NUMBER(38)
FLOAT(b)[2], DOUBLE PRECISION[3], REAL[4]	NUMBER

Définition des données

- Domaine
  - Exemple

```
CREATE DOMAIN COULEUR CHAR(6) DEFAULT ‘vert’
   CONSTRAINT VALIDE_COULEUR
   CHECK (VALUE IN ‘rouge’, ‘blanc’, ‘vert’, ‘bleu’, ‘noir’)
```

  - Modification de la définition (ALTER DOMAIN)
  - Destruction d’un domaine

```
DROP DOMAIN domaine [RESTRICT | CASCADE]
```

Définition des données

- Table
  - Création d ’une table (forme simple)
    - Exemple

```
CREATE TABLE VIN (
   NV NUMBER,
   CRU CHAR(10),
   MIL NUMBER)
```

  - Contraintes d ’intégrité
    - NOT NULL
    - UNIQUE ou PRIMARY KEY
    - FOREIGN KEY
    - REFERENCES
    - CHECK

Définition des données

- Table
  - Avec contraintes d ’intégrité

Exemple (SQL 86)

```
CREATE TABLE COMMANDE (
        NC NUMBER UNIQUE NOT NULL,
        NV NUMBER NOT NULL
        QUANTITE NUMBER(6))
```

Exemple (SQL 89)

```
CREATE TABLE COMMANDE (
        NC NUMBER PRIMARY KEY,
        NV NUMBER NOT NULL REFERENCES VIN,
        QUANTITE NUMBER(6) CHECK(QUANTITE > 0))
```

SQL 92

```
NC NUMBER,
PRIMARY KEY (NC),
NV NUMBER NOT NULL,
FOREIGN KEY (NV) REFERENCES VIN,
```

Définition des données

- Table
  - Modification du schéma d ’une table

    ALTER TABLE <table>

    ADD COLUMN<attribut> <type>, <attribut> <type>, …
    - Exemple

    ALTER TABLE COMMANDE

    ADD COLUMN DATE_COM DATE
  - Suppression d ’une table

    DROP TABLE <table>
    - Exemple

    DROP TABLE COMMANDE

Accès aux données

- Clause SELECT
  - Forme

SELECT [ALL | DISTINCT] {<liste_attributs> | *}

- Clause FROM
  - Forme

FROM <liste_tables>

- Exemple

Employe (mat#, nom, adresse, dept, sup)

Projet (num#, designation, budget)

Departement (dept#, dir, appellation)

Role (nume#, nump#, role_emp)

Accès aux données

- Exemple
  - Donner les noms et adresses des employés

```
SELECT nom, adresse
FROM Employe
```

  - Donner la liste des projets

```
SELECT *
FROM Projet
```

  - Donner les noms et les départements des employés

```
SELECT nom, appellation
FROM Employe, Departement
```

Accès aux données

- Clause WHERE
  - Forme

WHERE <condition>

- Exemple
  - Donner les noms et adresses des employés qui travaillent au DIRO

```
SELECT nom, adresse
FROM Employe, Departement
WHERE appellation = 'DIRO'
```

  - Donner les noms des employés responsable de projets

```
SELECT nom
FROM Employe E, Role R
WHERE E.mat = R.nume
AND role_emp = 'responsable'
```

Accès aux données

- Fonctions de calcul
  - COUNT, SUM, AVG, MAX, MIN
- Exemple
  - Donner le nombre d ’employés qui travaillent au DIRO

```
SELECT COUNT (*)
FROM Employe, Departement
WHERE appellation = ‘DIRO’
```

  - Donner le budget total des projets sous la responsabilité de l’employé numéro 35677

```
SELECT SUM (budget)
FROM Employe E, Role R, Projet
WHERE E.mat = R.nume
AND role_emp = ‘responsable’
AND mat = 35677
```

Accès aux données

- Forme de conditions
  - IN (NOT IN), Donner la liste des directeurs de départements

```
SELECT *
FROM Employe
WHERE mat IN
        (SELECT dir
        FROM Departement)
```

  - ANY ou ALL, Donner la liste des employés responsables de projets

```
SELECT *
FROM Employe
WHERE mat = ANY
        (SELECT nume
        FROM Role
        WHERE role_emp = ‘responsable’)
```

Accès aux données

- Forme de conditions
  - IS (NOT) NULL, Donner la liste des employés sans superviseurs

```
SELECT *
FROM Employe
WHERE sup IS NULL
```

  - EXISTS, Donner les numéro de projets dans lesquels intervient au moins un employé autre que le responsable

```
SELECT DISTINCT nump
FROM Role
WHERE EXISTS
        (SELECT nump
        FROM Role
        WHERE role_emp != ‘responsable’)
```

Accès aux données

- Clause GROUP BY
  - Forme

  GROUP BY <liste_attributs>
- Exemple
  - Donner les noms des employés regroupés par projet

```
SELECT nom
FROM Employe E, Role R
WHERE E.mat = R.nume
GROUP BY R.nump
```

Accès aux données

- Clause HAVING
  - Forme

HAVING <condition>

  - Exemple

Donner les numéros de projets dans lesquels interviennent au moins dix employés autres que le responsable

```
SELECT nump
FROM Role
WHERE role_emp != ‘responsable’
GROUP BY nump
HAVING COUNT(nume) > 9
```

Accès aux données

- Unions
  - Forme

    <requete> UNION <requete>
  - Exemple, liste des cadres

```
SELECT *
FROM Employe
WHERE mat IN ( SELECT dir
               FROM Departement)
UNION
SELECT *
FROM Employe
WHERE mat = ANY (      SELECT nume
                       FROM Role
                       WHERE role_emp = ‘responsable’)
```

Manipulation des données

- Insertion
  - Forme

INSERT INTO <table> [(<liste_colonnes>)]

{VALUES (<liste_valeurs>) | <requete>}

  - Exemple 1

```
INSERT INTO Employe (mat, nom, adresse, dept, sup)
VALUES (34098, ‘Gilles’, ‘3456, Gaspe, Mtl’, 456, 68735)
```

  - Exemple 2

```
INSERT INTO Directeur
SELECT *
FROM Employe
WHERE mat IN ( SELECT dir
               FROM Departement)
```

Manipulation des données

- Mise à jour
  - Forme

UPDATE <table>

SET {<nom_colonne> = <expression>}$^{+}$

WHERE {<condition> | CURRENT OF <nom_curseur>}

  - Exemple 1

```
UPDATE Employe
SET adresse = '3456, Gaspe, Mtl'
WHERE mat = 34098
```

  - Exemple 2

```
UPDATE Projet
SET budget = budget + 1000
WHERE num IN ( SELECT nump
                FROM Role
                WHERE nume = 34098
                AND role_emp = ' responsable ')
```

Manipulation des données

- Suppression
  - Forme

DELETE FROM <table>
[WHERE {<condition> | CURRENT OF <nom_curseur>}]

  - Exemple 1

```
DELETE FROM Employe
WHERE mat = 34098
```

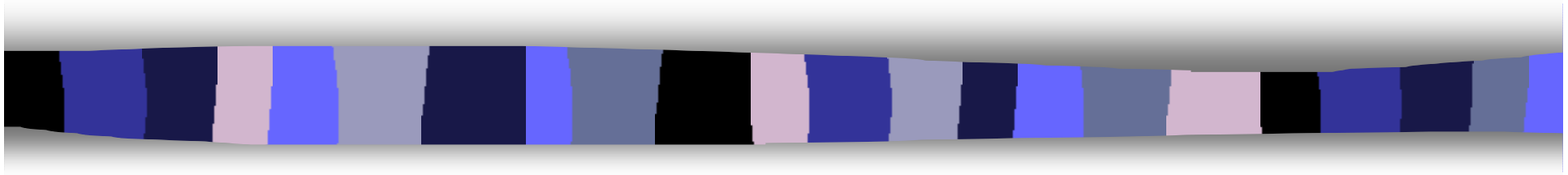
  - Exemple 2

```
DELETE FROM Projet
WHERE num IN ( SELECT nump
                FROM Role
                WHERE nume = 68754)
```

SQL intégré (JDBC)

- Introduction
  - SQL peut être intégré dans un langage hôte (C, COBOL, PL/1, PASCAL, JAVA, etc.)
  - Conflit (opération ensembliste vs. opération séquentielle)
    - traitement de plusieurs n-uplets
  - Étude du cas du langage Java (JDBC)

Base de données



Chapitre 4
Conception de bases de données

Plan du cours

- Introduction
- Modèle relationnel
- SQL
- **Conception**

Introduction

- Problème de la conception
  - Associer une structure logique aux données. En d'autres termes, trouver les relations et leurs attributs
- Techniques
  - Dépendances fonctionnelles
  - Les formes normales
  - Le modèle Entité-Association

Modélisation sémantique

- Permet de modéliser les donnés
- En utilisant différents formalismes
  - exemples : E/A, UML, …
- Une conception de bases de données qui commence par la modélisation sémantique est dite conception descendante (conception allant du plus haut niveau vers le plus bas niveau)

Modèle Entité/Association

- Généralités
  - Appelé également Entité/Relation
  - Proposé par Chen en 1976
  - Englobe à l'origine un nombre restreint de concept (entité, association, propriété)
  - Diverses extensions ont été proposé par la suite (contraintes, généralisation agrégation, …)

Modèle Entité/Association, Concepts

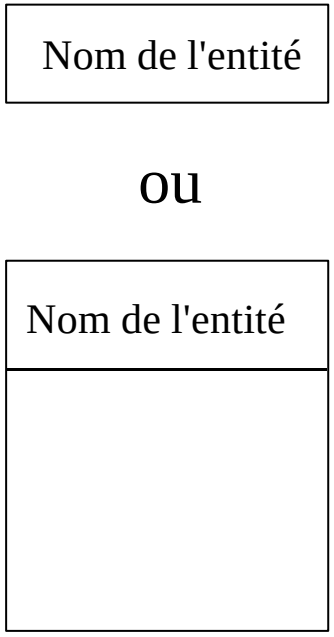
- Entité
  - "Une chose qui peut être identifiée distinctement"
  - Par abus de langage, on confond *entité* et *type d'entité*
  - Exemples :
    - Concrètes : voiture, employé,
    - Abstraites : projet, cours
  - Mais, mariage ???

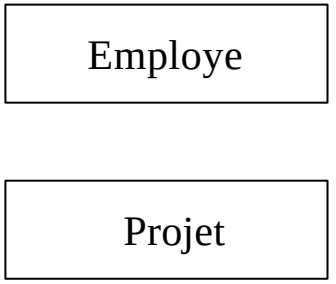
Modèle Entité/Association, Concepts

- Entité (suite)
  - On distingue souvent
    - Entité régulière
    - Entité faible dont l'existence dépend d'une autre entité
    - Exemple
      - Employé …. Entité régulière
      - Personne_a_charge …. Entité faible dont l'existence dépend de celle de l'entité Employé

Modèle Entité/Association, Concepts

- Entité (suite)
  - Représentation graphique





Modèle Entité/Association, Concepts

- Propriété
  - Les entités possèdent des propriétés qui les décrivent
  - Appelées également *Attributs*
  - Exemples :
    - Pour Employé, matricule, nom, adresse,
    - mais pas le numéro de département

Modèle Entité/Association, Concepts

- Propriété
  - Elle peut être
    - Simple ou composée
      - âge (simple)
      - adresse (composée)
    - Un identificateur (ou clé)
      - code_permanent
    - mono ou multi-valuée
      - nom (mono-valuée)
      - prénom (multi-valuée)

Modèle Entité/Association, Concepts

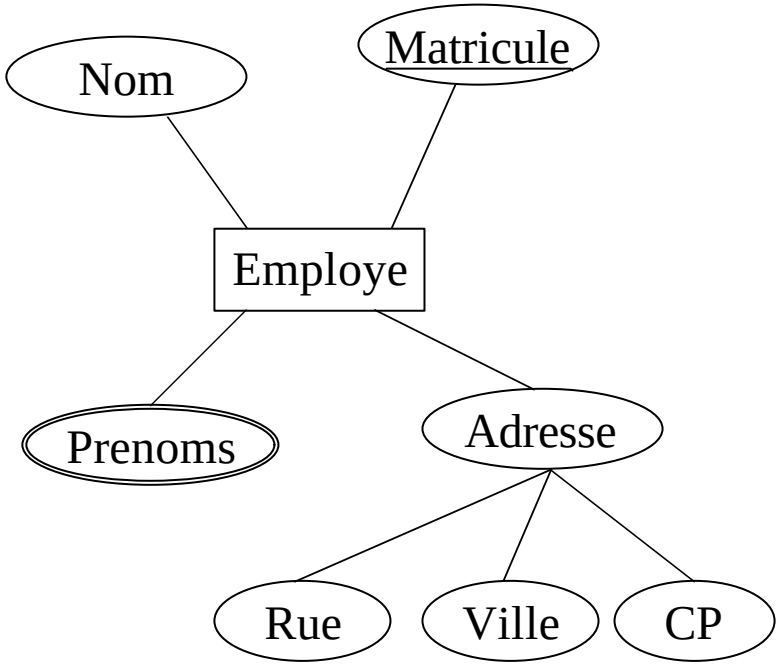
- Propriété
  - Elle peut être
    - Manquante
      - valeur inconnue
      - non applicable (date_mariage pour un célibataire)
    - De base ou dérivée
      - nom (base)
      - ancienneté (dérivée de la date d'embauche et de la date d'aujourd'hui)
      - Quantité totale (dérivé de la somme des quantités)

Modèle Entité/Association, Concepts

- Propriété
  - Représentation graphique

Parfois, on utilise cette notation pour les propriétés multi-valuées

Parfois, on utilise cette notation pour les identificateurs



Modèle Entité/Association, Concepts

- Association
  - Est une relation entre entités
  - Même abus de langage entre association et type d'associations
  - Exemple
    - *Participe_dans* entre Employé et projet
    - *Récolte* entre Producteur et Vin
  - Comme une entité, elle peut posséder des propriété

Modèle Entité/Association, Concepts

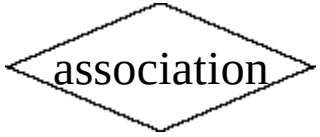
- Association
  - Elle est dite binaire si elle relie uniquement deux entité, ternaire si elle en relie trois, etc.
  - Chaque participant (entité) possède un rôle dans l'association
  - Elle peut être réflexive
    - Exemple : *est_mariée_à* entre personne et personne

Modèle Entité/Association, Concepts

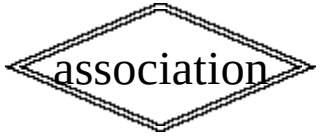
- Association
  - Cardinalités
    - Indiquent le nombre maximal et minimal d'occurrences d'une association pour une occurrence d'un participant.
    - Par exemple pour l'association *contrôle* entre département et projet
      - un département contrôle 0 ou plusieurs projet
      - un projet est contrôlé par un et un seul département
    - Les différentes possibilités sont
      - 0,1 ; 1,1 ; 0,N ; 1,N

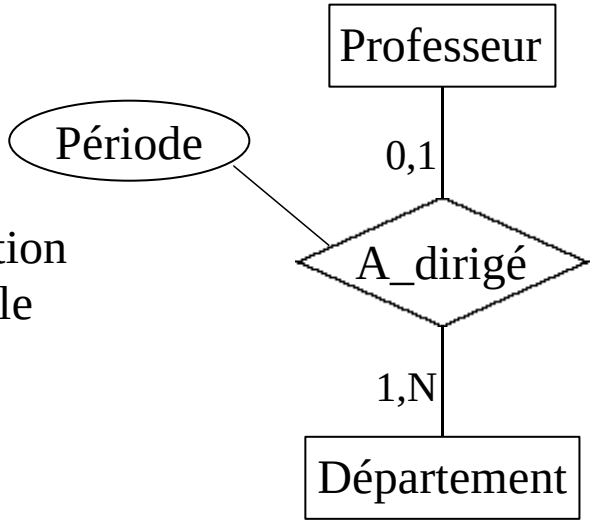
Modèle Entité/Association, Concepts

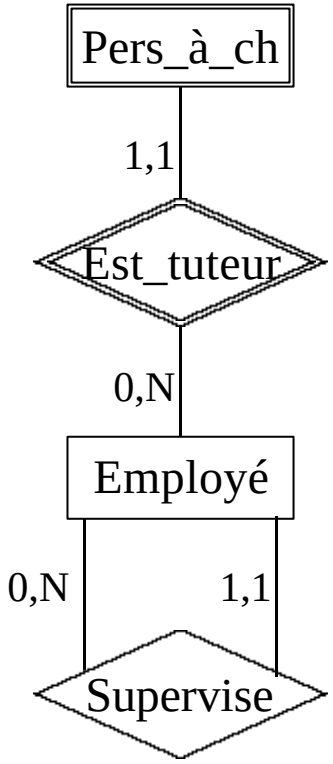
- Association
  - Représentation graphique



Parfois, on utilise cette notation pour une association impliquant une entité faible

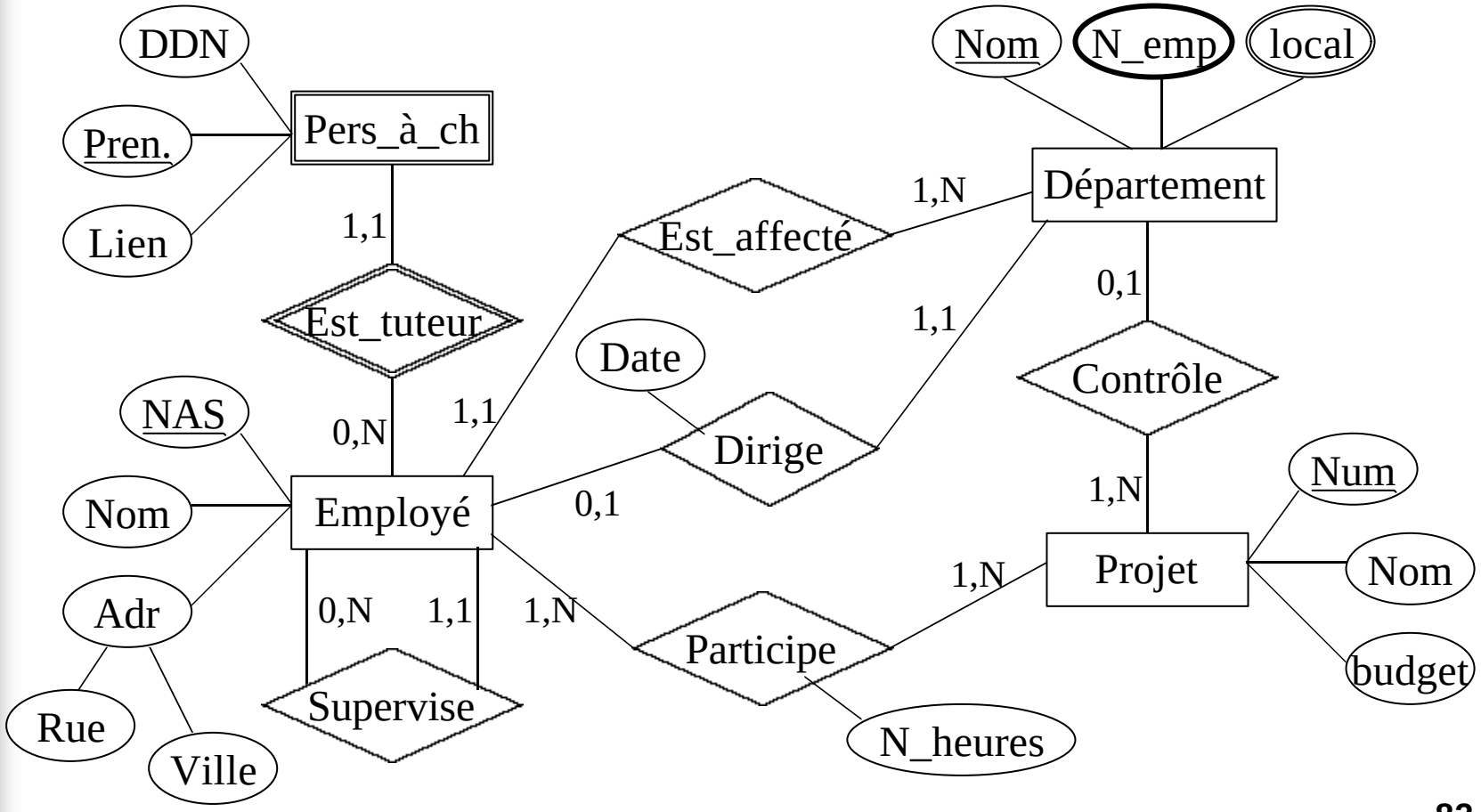






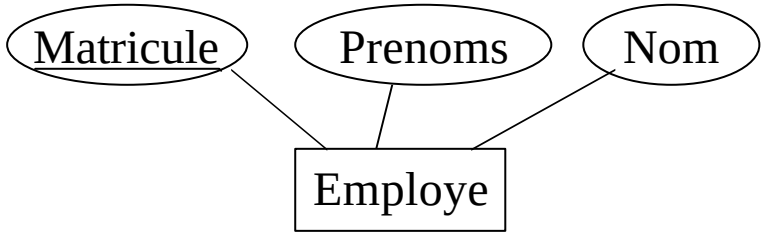
Modèle Entité/Association, Concepts

- Un exemple complet



Conception de la base de données E/A vers relationnel

- Entité
  - Chaque type d'entité devient une relation. Son identificateur devient la clé et les propriétés deviennent des attributs. Les propriétés multi-valuées sont transformées conformément à la 1NF



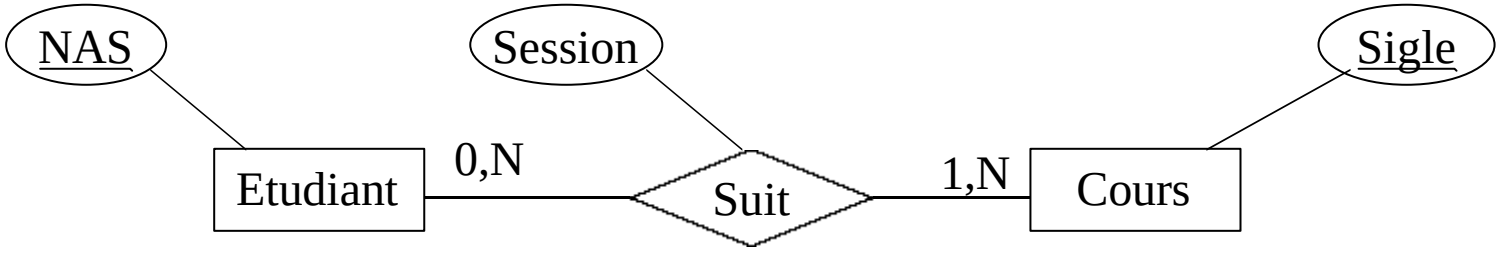
Employe(Matricule, Nom, Prenom)

Conception de la base de données E/A vers relationnel

- Association plusieurs à plusieurs

{0,N | 1,N} vers {0,N | 1,N}

  - Chaque type d'associations devient une relation
  - Les identificateurs des participants deviennent la clé de la relation

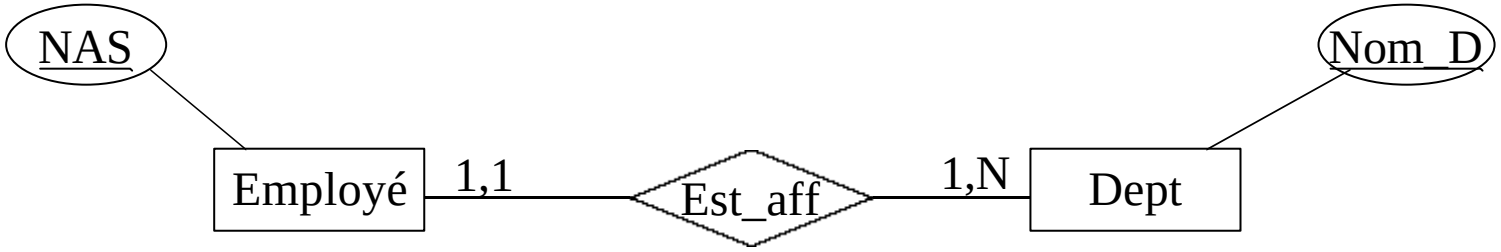


Inscription(NAS, Sigle, Session)

NAS et Sigle sont des clés étrangères également

Conception de la base de données E/A vers relationnel

- Association un à plusieurs {0,1 | 1,1} vers {0,N | 1,N}
  - Chaque type d'associations se traduit par un ajout d'une clé étrangère

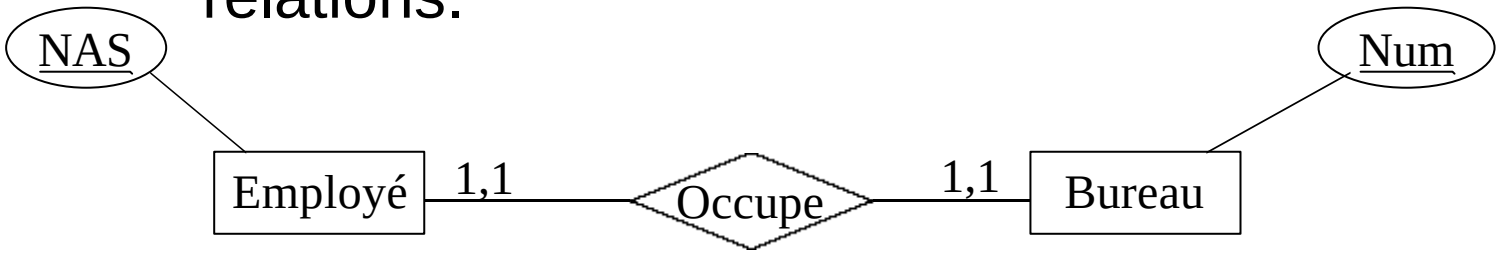


Employé(NAS, …, Nom_D)

Cas particulier d'une association impliquant une entité faible (contrainte CASCADE)

Conception de la base de données E/A vers relationnel

- Association un à un {0,1 | 1,1} vers {0,1 | 1,1}
  - Chaque type d'associations se traduit par
    - Une fusion des deux relations correspondantes
    - Un ajout de clé étrangère dans l'une ou les deux relations.



Option 1 :
Employé(NAS, …, Num)

Option 2 :
Employé(NAS, …, Num)
Bureau(Num, …, NAS)

Conception de la base de données E/A vers relationnel

- Convertir le modèle suivant en Relationnel