# Chapitre 01 : Généralités sur les bases de données.

## Introduction

L'approche classique de mise en place d'une application informatique dans une entreprise, consistait le plus souvent à l'écriture d'un certain nombre de programmes destinés à l'exploitation d'un ensemble de fichiers qu'il fallait aussi créer. Les principaux problèmes posés par cette démarche sont : la redondance des informations et la dépendance entre les données et les programmes qui les manipulent.

Les bases de données jouent un rôle central dans le développement des systèmes informatiques. Elles permettent de stocker l'information relative à un domaine d'application, d'en préserver l'intégrité, de l'extraire en utilisant un langage de haut niveau, de traiter plusieurs transactions simultanément, de répartir les données de façon transparente sur plusieurs supports, et d'assurer la sécurité et le recouvrement des données.

Dans les reste de ce cours, le mot BDD sera utilisé pour faire référence à une base de données.

## Définition

Une **base de données** est un ensemble d'informations ***structurées*** permettant la mise en place d'une série d'applications informatiques destinées à une grande variété d'utilisateurs.

D'après cette définition, on peut retirer les caractéristiques suivantes d'une base de données.

Données structurées : Les informations stockées dans une base de données ont une structure bien définie (enregistrements, tables, champs, …).

## Systèmes de gestion de bases de données (SGBD)

Un SGBD est un ensemble de programme qui permettent de créer et maintenir une base de données. Un SGBD doit fournir aux utilisateur les opérations suivantes :

## Modèle Entité-Association (E-A)

Dans le modèle E-A les données sont définies en tant qu'*Entités*, *Relations* et *Attributs*.

**Entités et leurs Attributs :**

Une **Entité** est un objet ayant une existence physique (ex : Personne, Voiture, Livre, …) ou ayant une existence conceptuelle (ex : Cours, Travail, …). Chaque entité possède un ensemble d'***attributs***. Une entité a une existence autonome.

Un **Attribut** est une caractéristique qui permet de décrire une entité. Par exemple, une entité *Etudiant* est décrite par les attributs : *Numéro d'inscription, Nom, Prénom, Date naissance, ….* Chaque attribut possède une valeur. Les valeurs d'attributs de chaque entité sont stockées dans une base de données.

On distingue plusieurs types d'attributs dans le modèle E-A :

**Attributs élémentaires (Simple, Atomique)** : sont des attributs qui ne peuvent pas être divisé en autres sous-attributs. (ex : code-étudiant, âge, note, nom-module, ….).

**Attributs composites** : peuvent être divisés en sous parties, qui représentent des attributs élémentaires ayant des significations différentes. (Ex : l'attribut *adresse* peut-être diviser en attributs élémentaires : *numéro, rue, ville, code-postal*).

Les attributs composites permettent de modéliser une situation dans laquelle l'utilisateur fait référence soit à l'attribut composite dans sa *globalité*, soit à ses *composants*. *Si* l'on fait référence à l'attribut composite dans *sa globalité*, c'est *inutile* de le *subdiviser*. Par exemple, si on n'a pas besoin de faire référence aux sous-attributs d'une adresse (rue, ville code postal), l'adresse entière peut être utilisée comme un seul *attribut simple*.

**Attributs monovalués** : sont des attributs qui ont une *valeur unique*. Par exemple le *nom d'un étudiant, l'âge, le numéro d'inscription, …*

**Attributs multivalués** : sont des attributs qui peuvent posséder un *ensemble de valeurs en même temps*. Par exemple l'attribut *couleur* pour les voitures ayants *plusieurs* couleurs. Ou l'attribut *Numéro-téléphone* pour les étudiants qui ont *plusieurs* numéros de téléphones.

**Attributs stockés et attributs dérivés (calculés) :**

La valeur d'un ***Attribut stocké*** est indépendante des autres attributs. On ne peut pas connaitre ou calculer cette valeur à partir des valeurs d'autres attributs.

Par contre, La valeur d'un ***Attribut dérivé*** peut être calculée à partir des valeurs d'autres attributs. Par exemple l'*âge* d'un étudiant peut être *calculé* à partir de sa *date de naissance*.

**Valeurs NULL :**

La valeur NULL signifie qu'on ne peut pas affecter une valeur à un attribut d'une entité. Elle est utilisée dans 02 cas :

***Valeur inexistante*** : par exemple la valeur de l'attribut *Numéro de téléphone* pour les étudiants qui *ne possèdent pas* un téléphone.

***Valeur inconnue*** : dans ce cas, la valeur existe, mais on le la connait pas. Par exemple, l'étudiant possède un téléphone, mais on ne connait pas son numéro.

**Types d'entités :**

Un type d'entité est un ensemble d'entités ayant les mêmes attributs (ex : *Etudiant, Livre, Cours, ...*). Chaque type d'entité est décrit par son nom et ses attributs. L'ensemble des entités d'un type est appelé ***ensemble d'entités***[1] (exemple : *étudiant 1, étudiant 2, étudiant 3, ...*).

Un type d'entité est représenté par une boite rectangulaire contenant son nom. Les noms d'attributs sont indiqués dans des ovales et sont rattachés à leurs type l'entités par des lignes droites. Les attributs composites sont rattachés à leurs sous attributs par des lignes droites. Les attributs multivalués sont affichés dans des ovales doubles.

Un type d'entité représente le « ***schéma*** ou ***l'intention*** » d'un ensemble d'entités (qui partagent la même structure), tandis que l'ensemble des entités (occurrences) représentent « ***l'extension*** » des entités. Le schéma d'une entité ne va pas changer fréquemment car il décrit la structure de l'entité. L'extension peut changer souvent : à chaque insertion ou suppression d'une occurrence d'entité.

**Clé ou Identifiant d'une entité (Contraint d'unicité) :**

Un attribut clé (Identifiant) permet d'identifier chaque entité de façon unique ; deux entités différentes ne peuvent pas avoir la même valeur de l'attribut clé. (*Ex : deux étudiants différents ne peuvent pas avoir le même numéro d'inscription*).

- Dans certains cas, la clé d'une entité peut être obtenu par la concaténation d'un ensemble d'attributs (exemple : *Nom + Date Naissance*).
- La clé permet de s'assurer que c'est une entité (et non une association).
- Plusieurs identifiants peuvent co-exister. (Ex : *Numéro d'inscription* et *Numéro de sécurité sociale d'un étudiant*).

**Domaines d'attributs :**

Un domaine d'attribut est *l'ensemble des valeurs* que *peut prendre* cet attribut (*ex : le domaine de l'attribut âge de l'étudiant est définit sur les nombres entiers positif entre [15 et 60]).* Les ensembles de

[1] On appelle aussi ensemble d'entités « **Occurrences d'entité** ».

valeurs ne sont pas affichés dans les diagrammes E-A, ils sont définis à l'aide des types de données disponible dans la plupart des langages de programmation (entiers, chaines de caractères, réels, booléens, …).

**Types de relations :**

Un type de relation R parmi un ensemble d'entités $E_1,E_2,\ldots,E_n$ définit un ensemble d'associations parmi des entités appartenant à ces types. Chaque instance de relation Ri dans R associe exactement une entité de chaque type à une autre. Chaque instance de relation Ri est la représentation symbolique du lien existant entre les entités (dans le système à modéliser). Par exemple le type de relation *Enseigner* associe un type d'entité *Enseignant* à un type d'entité *Etudiant.*

Chaque type d'entité (ou occurrence d'entité) joue un rôle particulier dans l'association. Une relation possède un nom et peut contenir des attributs. Les valeurs de ces attributs dépendent des valeurs des entités qui participent à cette relation.

**Degré des relations :**

C'est le nombre de types d'entités qui participent à la relation. Une relation de degré deux est une relation binaire, Une relation de degré trois est une relation ternaire, …

**Noms de rôles**

Le nom de rôle indique le rôle joué par chaque entité dans une relation et permet de décrire la relation.

**Relations récursives**

Dans ce type de relation, un *même type* d'entité peut participer plusieurs fois à *un type* de relation en jouant un *rôle diffèrent*. Le nom de rôle permet alors de faire cette distinction.

**NB :** Les noms de rôles ne sont pas nécessaires dans les types de relations dans lesquels tous les types d'entités participantes sont distincts : chaque nom de type d'entité peut être utilisé comme nom de rôle.

**Contraintes sur les types de relations**

***Ratio de cardinalité (Cardinalité maximale) :*** définit le nombre *maximal* d'instances de relations auxquelles une entité peut participer.

***Contrainte de participation (Cardinalité minimale) :*** définit le nombre *minimal* d'instances de relations auxquelles une entité peut participer.

Si la cardinalité minimale est égale à 0, la participation du type d'entité dans le type de relation est *partielle*.

Si la cardinalité minimale est égale à la cardinalité maximale (généralement 1), la participation du type d'entité dans le type de relation est *Totale*. (Appelé aussi ***dépendance existentielle***).

**Types d'entités faibles**

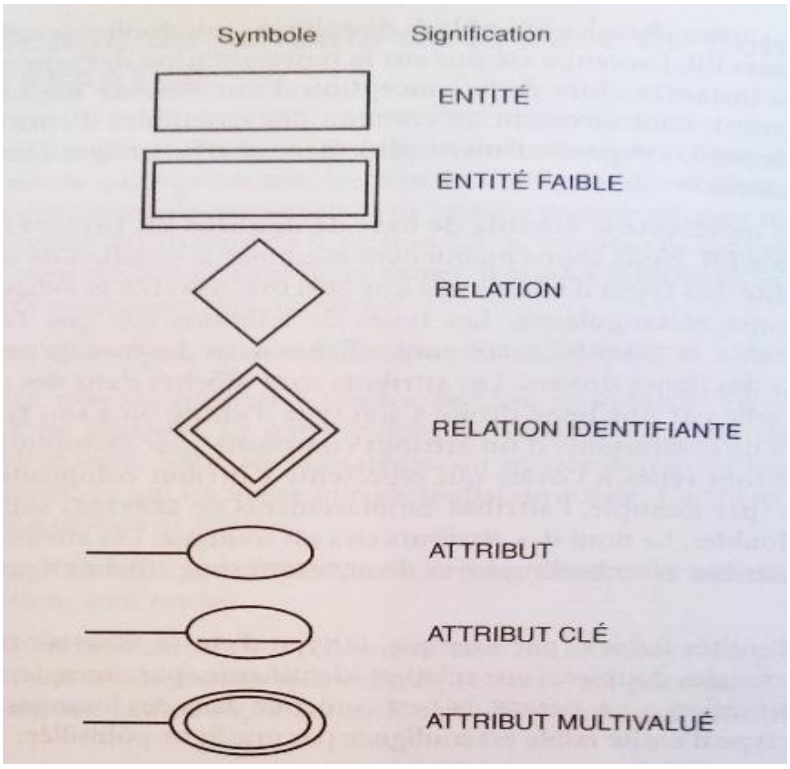
Sont des types d'entités qui *n'ont* pas d'attributs ***clés***. Les entités appartenant à ce type sont dites *Identifiables* : elles sont liées à un autre type d'entité (entité *identifiante* ou *propriétaire*) et à l'une de leur valeur d'attribut. Le type de relation qui lie un type d'entité faible à son propriétaire est appelé relation *identifiante*. Les types d'entités faibles sont toujours soumis à une contrainte de participation totale dans leurs relation identifiante (cardinalités 1-1).

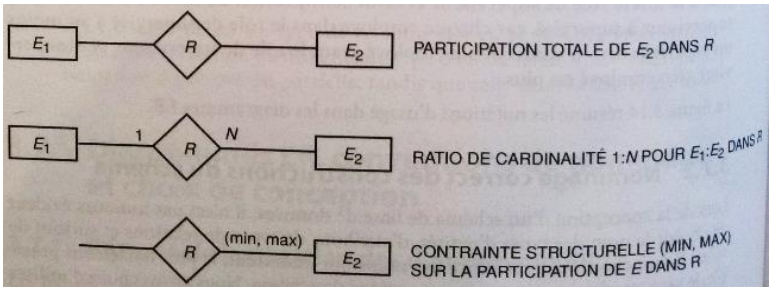
Un type d'entité faible possède généralement une *clé partielle* (un ou un ensemble d'attributs) permettant d'identifier de façon unique les entités faibles liés à la même entité propriétaire.

Dans les diagrammes E-A, les types d'entités faibles et leurs relations identifiantes sont représentés par des boites et des losanges entourés de lignes doubles. L'attribut le la clé partielle est soulignée en pointillés de tirets.

**NB** : les types d'entités faibles sont parfois représentées à l'aides d'attributs composites ou multivalués (petit nombre d'attributs). S'il y a beaucoup d'attributs, la représentation utilisant les types d'entités faibles est préférable.

**Notation des diagrammes E-A**



**Passage du modèle E-A au modèle relationnel**

**Objectif :** Conception d'un schéma d'une BDD relationnelle à partir à partir d'un modèle de schéma conceptuel (modèle E-A).

**Etape 01 : Conversion des types d'entités normaux :**

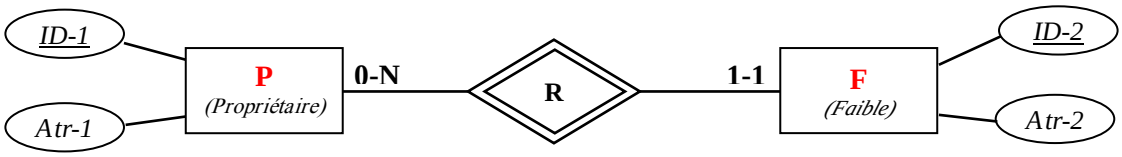
Pour chaque type d'entité normale *E* dans le modèle E-A :

- Créer une relation (Table) ***R*** qui inclut tous les attributs simples de ***E***.
- S'il y a des attributs composites, inclure seulement les attributs de composants simples d'un attribut composite.
- Choisir l'un des attributs *clé* de *E* (s'il y a plusieurs) comme clé primaire de ***R***.
- Si la clé choisie est composite, la clé primaire de ***R*** sera constituée de l'ensemble des attributs simples qui la constituent.

**Etape 02 : Conversion des types d'entités faibles :**

Pour chaque type d'entité faible ***F*** dans le modèle E-A :

- Créer une relation (Table) ***R*** qui inclut tous les attributs simples de ***F***.
- Inclure les attributs simples (ou composants simples des attributs composites) d'un attribut composite comme attributs de ***R***.
- Inclure l'attribut clé primaire de la *relation propriétaire* comme clé *étrangère* de ***R***.
- La *clé primaire* de ***R*** est la combinaison de la *clé primaire* de la relation *propriétaire* et la *clé partielle* de l'entité ***F***.




**P1** (ID-1, Atr-1).

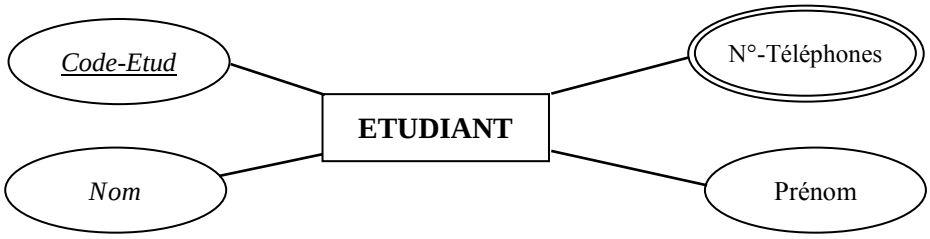
**F2** (ID-2, #ID-1, Atr-2).

**NB** : S'il existe un type d'entité faible *F2* dont le propriétaire est un type d'entité faible *F1*, *F1* doit être traduit avant *F2* afin de déterminer sa clé primaire en premier.

**Etape 03 : Conversion des attributs multivalués**

Pour chaque attribut multivalué *A* :

- Créer une relation ***R*** qui contient un attribut correspondant à *A*. (Si l'attribut *A* est composite, ses composants simples doivent être inclus).
- Inclure l'attribut *K* (*clé primaire*) de l'entité qui contient l'attribut *A* comme *clé étrangère* de ***R***.
- La *clé primaire* de ***R*** est la ***combinaison*** des attributs ***A*** et ***K***.




Etudiant (**Code-Etud**, Nom, Prénom).

Téléphones **(#Code-Etud, N°-Téléphone)**.

**NB** : Si le nombre maximal des valeurs qu'un attribut multivalué peut prendre est *très limité*, dans ce cas on peut inclure cet attribut comme un *ensemble* d'attributs *simples*.

**Exemple :** Si chaque étudiant possède au maximum 02 numéros de téléphone, on peut écrire la relation *Etudiant* comme suit (sans ajouter la relation *Téléphones*) :

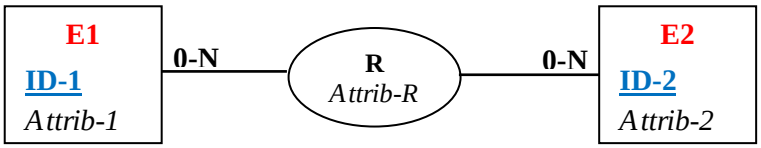
Etudiant (Code-Etud, Nom, Prénom, N°-Téléphone-1, N°-Téléphone-2).

**Etape 04 : Conversion des types d'associations binaires (0-N : 0-N) ou (1-N :1-N)**

Pour chaque type d'association S :

Créer une relation (Table) ***R*** qui inclut tous les attributs simples de ***S.***

Inclure les ***deux clés*** des entités participantes à ***S*** comme ***clé primaire*** de ***R***.



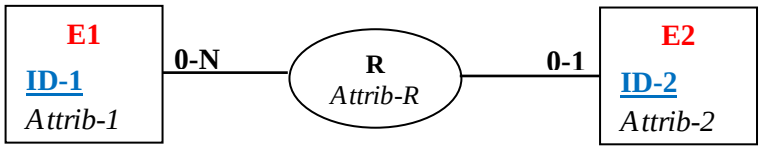

E1(**ID-1**, Attrib-1)

E2 (**ID-2**, Attrib-2)

R (**#ID-1, #ID-2**, Attrib-R)

**Etape 05 : Conversion des types d'associations binaires (0-N : 0-1) ou (0-N :1-1)**

Inclure la clé de l'entité ayant les cardinalités (0-N) ou (1-N) comme clé étrangère de la relation (table) correspondante à l'entité ayant les cardinalités (0-1) ou (1-1). Les attributs de l'association doivent être également inclus à cette relation.



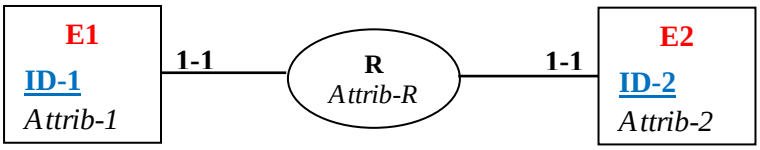

E1(**ID-1**, Attrib-1)

E2 (**ID-2**, Attrib-2, # **ID-1**, Attrib-R)

**Etape 06 : Conversion des types d'associations binaires (1-1 : 1-1)**

1. La clé étrangère.

   *Inclure la clé étrangère dans la relation (table) correspondante à l'une des entités (au choix du concepteur).*

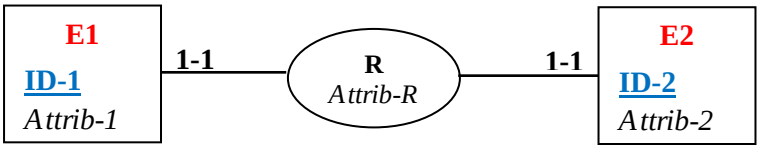



E1(**ID-1**, Attrib-1)

E2 (**ID-2,** Attrib-2, **# ID-1,** Attrib-R)

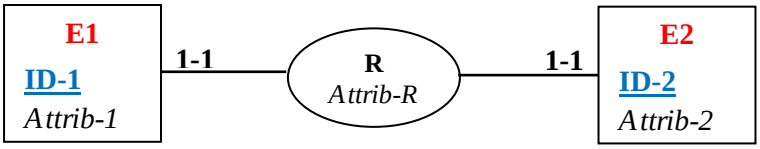
2. Les relations fusionnées.

   *Fusionner les attributs des deux entités en une seule relation R. La clé primaire de R est choisie parmi l'un des deux clés des entités participantes.*




E (**ID-1**, Attrib-1, ID-2, Attrib-2, Attrib-R)

3. Référence croisée (Nouvelle relation contenant les deux clés des entités participantes).




E1(**ID-1**, Attrib-1)

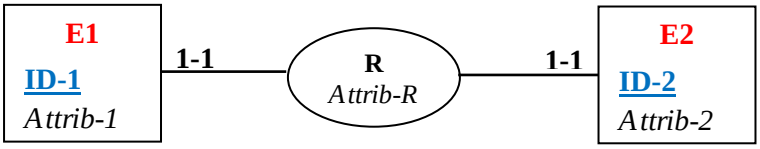
E2 (**ID-2,** Attrib-2)

R (**#ID-1, #ID-2**, Attrib-R)

**Etape 07 : Conversion des types d'associations binaires (0-1 : 0-1) :**

1. Référence croisée (Nouvelle relation ***R***) → Solution préférée.

   Inclure les ***deux clés*** des entités participantes comme ***clé primaire*** de ***R***.




E1(**ID-1**, Attrib-1)

E2 (**ID-2,** Attrib-2)

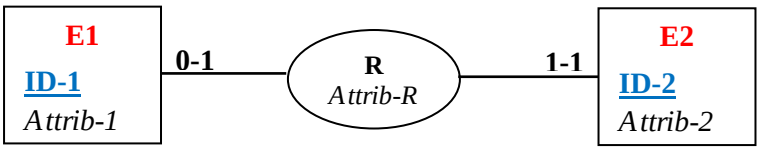
R (**#ID-1, #ID-2**, Attrib-R)

2. La clé étrangère.
3. Les relations fusionnées.

**Etape 08 : Conversion des types d'associations binaires (0-1 : 1-1) :**

La clé étrangère →***Solution préférée***.

Inclure la clé étrangère dans la relation correspondante à l'entité ayant les cardinalités (1-1).




E1(**ID-1**, Attrib-1)

E2 (**ID-2,** Attrib-2, **# ID-1,** Attrib-R)

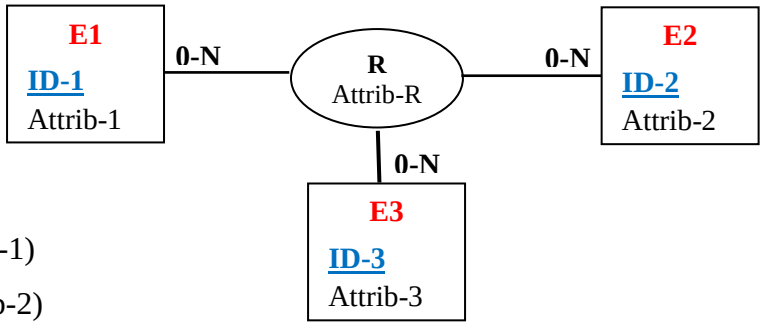
**Etape 08 : Conversion des types d'associations n-aires (n>2)**

**Cas 01 :**

Pour chaque type d'association ***S*** (relation) n-aire ayant des cardinalités 1-N sur tous les branches :

Créer une relation ***R*** ayant comme attributs les attributs de **R**.

La clé primaire de ***R*** est la concaténation des clés primaires des entités participantes à ***S***.




E1(**ID-1**, Attrib-1)

E2 (**ID-2,** Attrib-2)

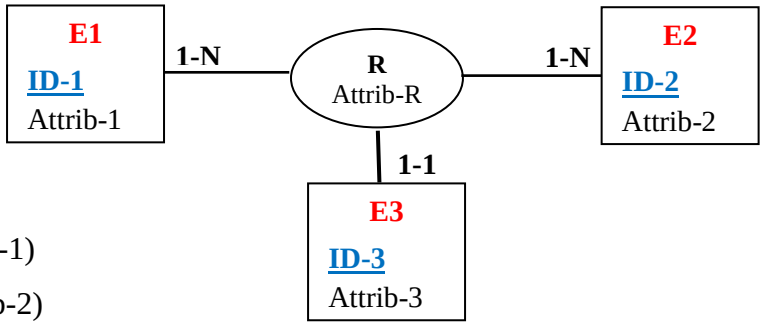
E3 (**ID-3,** Attrib-3)

R (**#ID-1, #ID-2, #ID-3**)

**Cas 02 :**

Pour chaque type d'association ***S*** (relation) n-aire ayant des cardinalités 1-1 sur une des branches :

Inclure les clés primaires ainsi que les attributs de ***S*** dans la relation correspondante à l'entité reliée par la branche 1-1.



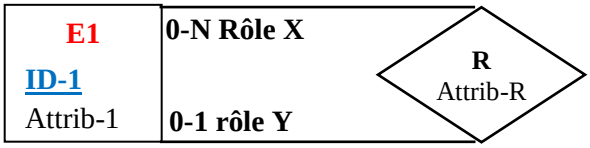

E1(**ID-1**, Attrib-1)

E2 (**ID-2,** Attrib-2)

E3 (**ID-3,** #ID-1, #ID-2, Attrib-3, Attrib-R)

**Etape 04 : Conversion des types de relations récursives avec des rôles (0-1 : 0-N)**

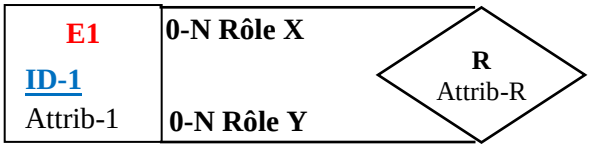
Inclure la clé primaire correspondante au nom rôle ayant la cardinalité N dans le relation R correspondante à l'entité participante à la relation récursive.




E1(**ID-1**, Attrib-1, **ID-1(X)**)

**Etape 04 : Conversion des types de relations récursives avec des rôles (0-N : 0-N)**

Créer une nouvelle relation R contenant les deux clés primaires correspondants aux noms des rôles de la relation récursive.




E1(**ID-1(X)**, Attrib-1)

R(#**ID-2(Y)**, #**ID-2**, Attrib-R)