Chapitre III : Modèle relationnel

Introduction

Le modèle relationnel est créé au début des années 70 par Edgar Frank CODD, un mathématicien de formation et chercheur chez IBM. Les produits utilisant ce modèle ont commencé à être commercialisés vers les années 1980 et ont apporté de nouvelles fonctionnalités qui permettent un confort d'utilisation sans précédent. Ce modèle s'est imposé dans les entreprises avec l'adoption de ses concepts par les systèmes commerciaux. Ses principaux concepts sont les relations, leurs attributs, les domaines des attributs, les enregistrements… Un de ses principaux avantages est la manipulation ensembliste des relations par les opérations de l'algèbre relationnelle sur lesquelles sont construits des langages non procéduraux tels que SQL.

Le modèle relationnel ne manipule pas des structures de données figées, mais des valeurs : aucun chemin d'accès n'est préalablement défini, toute manipulation des données est possible. Ce modèle représente les informations dans des structures appelées relations.

C'est un modèle qui se base sur des fondements mathématiques tels que l'algèbre relationnelle et la logique des prédicats.

I. Notions fondamentales

I. 1. Domaine

C'est un ensemble de valeurs appartenant à un type de base (entier, réel, chaine de caractères, booléen, etc.) sur lesquelles portent éventuellement un certain nombre de contraintes. Ces contraintes permettent d'exclure des valeurs et de ne considérer qu'une partie des valeurs du type de base. Son nombre de valeurs peut être fini ou infini.

Exemple :

- **Types prédéfinis :** Entier, Réel, Booléen…
- **Types prédéfinis avec des contraintes supplémentaires :** Réels appartenant à l'intervalle [0, 20], Entiers x tels que 50 < x < 150, Chaines de 15 caractères au maximum, …
- **Types énumérés :**
  - ✓ Jour ouvrable = {"Lundi", "Mardi", "Mercredi", "Jeudi", "Vendredi"} ;
  - ✓ Mois = {Janvier, Février, Mars Avril, Mai, Juin, Juillet, Aout, Septembre, Octobre, Novembre, Décembre} ;

- ✓ Sexe = {"Masculin", "Féminin"}

I. 2. Attribut

Un attribut est caractérisé par un nom et prend ses valeurs dans un domaine de valeurs. Il correspond à une colonne d'une table (implémentation d'une relation) dans la base de données. Les noms des attributs d'une relation sont tous différents. Cependant, deux (ou plusieurs) relations différentes peuvent avoir des attributs de même nom. Par exemple le *nom* peut être attribut de la relation *Produit* et de la relation *Personne*.

Un attribut est dit composé s'il peut être remplacé par plusieurs attributs. Par exemple, l'adresse d'une personne peut être remplacée par les attributs *Ville, Quartier, Rue, NumVilla*.

I. 3. Enregistrement ou N_uplet

Un enregistrement est une ligne d'une table dans la base de données. Il est aussi appelé N_uplet ou tuple. C'est un ensemble de valeurs dont chacune appartient au domaine d'un des attributs de la relation.

Exemple :

Soit la relation *Bureau* suivante :

Bureau (Numero, #Centre, Nb_Inscrit, Type, President)

(1, 'Ecole Boucotte Sud', 528, 'Normal', 1254701) et (2, 'Ecole Tilène', 589, 'Pilote', 2145781) peuvent être des enregistrements de la relation *Bureau*.

Remarque :

A l'exception de la clé primaire, si une contrainte sur un attribut ne l'interdit pas :

- sa valeur peut être omise pour un enregistrement donné :
  Exemple : (3, 'CFP de Lindiane', , 1454250) est un enregistrement de la relation ***Bureau*** dont le nombre d'inscrits n'est pas connu ;
- sa valeur peut être remplacée par NULL si elle n'est pas connue :
  Exemple : (3, 'CFP de Lindiane', NULL, 1245782) est un enregistrement de la relation **Bureau** dont le nombre d'inscrits est remplacé par la valeur Null.

I. 4. Relation

Une relation est un sous-ensemble du produit cartésien d'une liste de domaines ayant un nom unique dans la base. Elle est représentée par un tableau à deux dimensions dont chaque

colonne correspond à un attribut et chaque ligne à un enregistrement. Ainsi, le nombre de colonnes correspond au nombre d'attributs. Le nombre d'attributs d'une relation est appelé degré de la relation.

Le produit cartésien d'un ensemble de domaines D_1, D_2, D_3, …, D_n noté $D_1xD_2xD_3x…xD_n$ est l'ensemble des enregistrements (ou tuples) (u_1, u_2, u_3, …, u_n) tels que u_i appartient à D_i.

Exemple :

Le produit cartésien des deux ensembles E_1 = {A, B, C, D} et E_2 = {α, β, μ} est l'ensemble E_1 x E_2 = {(A, α), (A, β), (A, μ), (B, α), (B, β), (B, μ), (C, α), (C, β), (C, μ), (D, α), (D, β), (D, μ)}.

I. 5. Formes normales

I. 5. 1. Notion de clé

I. 5. 1. 1. Clé candidate

Une clé candidate d'une relation est un sous-ensemble de son ensemble d'attributs dont les valeurs identifient de manière unique et sans ambiguïté un enregistrement et un seul.

Chaque relation a au moins une clé candidate et peut en avoir plusieurs. Les clés candidates ne sont pas forcément composées du même nombre d'attributs. Deux enregistrements différents ne peuvent pas avoir les mêmes valeurs d'une clé candidate donnée.

I. 5. 1. 2. Clé primaire

La clé primaire d'une relation est choisie parmi ses clés candidates. La clé candidate choisie doit être parmi celles composées du plus petit nombre d'attributs.

Une clé primaire permet de différentier les enregistrements d'une relation mais aussi d'éviter qu'un enregistrement puisse être dupliqué.

I. 5. 1. 3. Clé étrangère

Une clé étrangère d'une relation R_1 est la clé primaire d'une relation R_2 qui a migré vers R_1 lors du passage du modèle entité/association au modèle relationnel. Elle permet d'établir les liaisons entre leurs enregistrements et particulièrement le fait qu'un enregistrement de R_1 (enregistrement fils) dépende d'un enregistrement de R_2 (enregistrement père). La clé étrangère doit avoir le même domaine que la clé primaire qu'elle référence. On dit que R1 et R2 sont liées par une ***contrainte d'intégrité référentielle***.

I. 5. 2. Dépendance fonctionnelle

I. 5. 2. 1. Dépendance fonctionnelle élémentaire

Une dépendance fonctionnelle est élémentaire si sa source ne contient pas d'attributs superflus. On ne parle de DF élémentaire que lorsque la source de la DF contient 2 attributs au moins.

Exemple :

La DF A, B → C telle que la DF B → C existe n'est pas élémentaire car A est superflus.
La DF X → Y, Z est élémentaire.

I. 5. 2. 2. Dépendance fonctionnelle directe

La dépendance fonctionnelle A → B est directe s'il n'existe aucun attribut C tel que les DF A → C et C → B existent. En d'autres termes, cela signifie que la dépendance entre A et B ne peut pas être obtenue par transitivité.

I. 5. 3. Les formes normales

I. 5. 3. 1. Première forme normale

Une relation respecte la première forme normale notée (**1FN**) si :
- Tous ses attributs sont atomiques ;
- Tous ses attributs sont simples.

I. 5. 3. 2. Deuxième forme normale

Une relation respecte la deuxième forme normale notée (**2FN)** si :
- elle respect la première forme normale ;
- toutes les DF entre la clé primaire et les autres attributs sont élémentaires.

I. 5. 3. 3. Troisième forme normale

Une relation respecte la troisième forme normale notée (**3FN**) si :
- elle respect la deuxième forme normale ;
- toutes les DF entre la clé primaire et les autres attributs sont directes.

I. 5. 4. Normalisation d'une base de données

La normalisation du schéma relationnel d'une base de données permet de l'optimiser. Normaliser une base de données consiste à normaliser toutes ses relations. Une relation est

normalisée si elle respecte la troisième forme normale. Une base de données est normalisée si toutes ses relations sont normalisées. La normalisation d'une base de données a pour but de :

- limiter les redondances de données ;
- limiter les pertes de données ;
- éliminer les incohérences ;
- améliorer les performances des traitements.

II. Schéma et instance d'une base de données

II. 1. Schéma d'une base de données

Le schéma d'une relation est sa représentation logique à l'aide du modèle relationnel. Il est composé de son nom et de ses attributs avec leur domaine : R_1 (A_1 : D_1, A_2 : D_2, …, A_n : D_n). L'ensemble des schémas des relations d'une base de données donne son schéma.

II. 2. Instance d'une base de données

L'instance d'une relation à un instant t donné est l'ensemble des enregistrements qu'elle contient à cet instant. L'instance d'une base de données à un instant t donné correspond à l'ensemble des instances de ses relations à cet instant. C'est le contenu de la base à l'instant considéré.

III. Passage du modèle entité/association au modèle relationnel

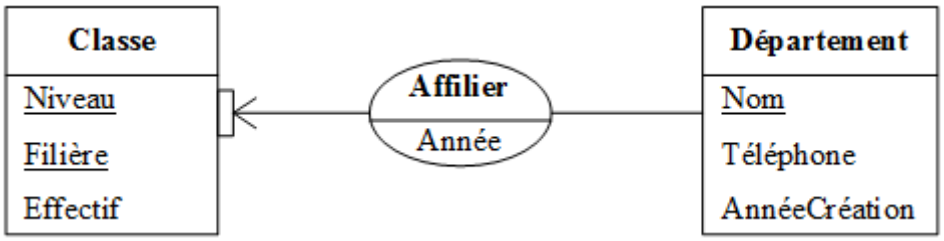
Le passage du modèle entité/association au modèle relationnel suit un certain nombre de règles portant pour la plupart sur les cardinalités des associations. Cependant il faut d'abord normaliser le modèle entité/association avant de passer au modèle relationnel.

III. 1. Règles portant sur les entités

R_1 : Chaque entité indépendante E est représentée par une relation R dont le schéma est celui de l'entité. L'identifiant de E devient la clé primaire de R.

R_2 : Chaque entité faible E est représentée par une relation R qui regroupe tous ses attributs ainsi que la clé primaire de la relation représentant l'entité forte associée. Cette clé participe à la clé primaire de la relation représentant l'entité faible.

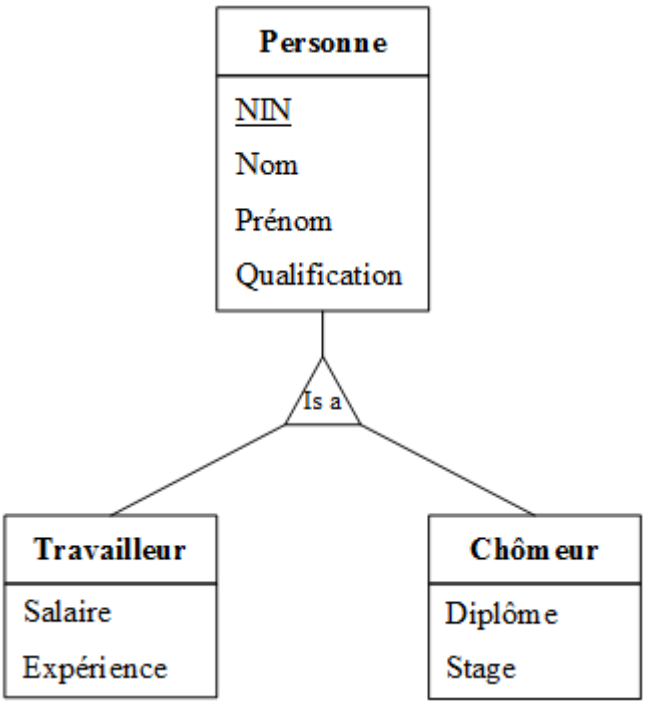
Exemple :



Classe (Niveau, Filière, #Département, Effectif, Année)

Département (Nom, Téléphone, AnnéeCréation)

R3 : Dans le cas de généralisation-spécialisation, il y a plusieurs manières de procéder :



1. **Méthode 1 :** Dans le cas d'une spécialisation incomplète, l'entité générale E est représentée par une relation R, chaque entité spécialisée E_i est représentée par une relation R_i. Les relations R_i représentant les entités spécialisées prennent la clé primaire de celle représentant l'entité générale et en font leur la clé primaire.

   Exemple :

   Personne (NIN, Nom, Prénom, Qualification)

   Travailleur (NIN, Nom, Prénom, Qualification, Salaire, Expérience)

   Chômeur (NIN, Nom, Prénom, Qualification, Diplôme, Stage)

2. **Méthode 2 :** Dans le cas d'une spécialisation complète, les entités spécialisées E_i sont représentées chacune par une relation R_i contenant les attributs spécifiques de l'entité E_i et tous les attributs de l'entité générale. L'identifiant de l'entité générale devient la

clé primaire de chaque relation R_i. Dans ce cas, l'entité générale ne sera pas représentée par une relation dans le modèle relationnel.

Exemple :

Travailleur (<u>NIN</u>, Nom, Prénom, Qualification, Salaire, Expérience)

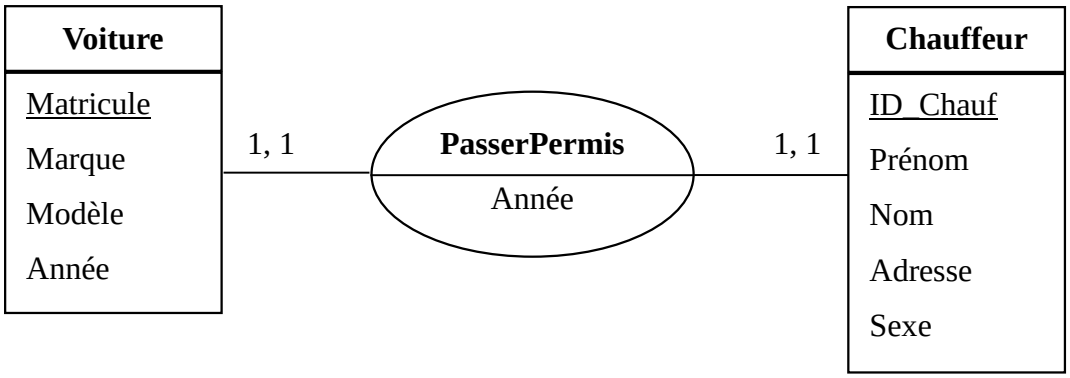
Chômeur (<u>NIN</u>, Nom, Prénom, Qualification, Diplôme, Stage)

III. 2. Règles portant sur les associations

R_1 : Pour chaque association binaire A entre les entités E et F (représentées par les relations R_E et R_F respectivement) avec des cardinalités (1, 1) dans les deux sens :

- **Méthode 1 :** Chaque entité (E et F) est représentée par une relation ainsi que l'association A qui les relie. Dans ce cas, la relation représentant l'association A héritera des clés primaires des deux relations R_E et R_F en plus de ses propres attributs éventuels. Les clés héritées composeront sa clé primaire.
- **Méthode 2 :** Une des deux relations représentant les entités reliées (R_E ou R_F) par l'association prend la clé primaire de l'autre comme clé étrangère et les attributs éventuels de l'association. Alors, l'association ne sera pas représentée par une relation.

Exemple :



1ère méthode :

Chauffeur (<u>Id_Chauf</u>, Nom, Prénom, Adresse, Sexe)

Voiture (<u>Matricule</u>, Marque, Modèle, Année)

PasserPermis (#<u>Id_Chauf</u>, #<u>Matricule</u>, Année)

2ème méthode :

Chauffeur (<u>Id_Chauf</u>, Nom, Prénom, Adresse, Sexe, #Matricule, Année)

Voiture (<u>Matricule</u>, Marque, Modèle, Année)

Ou

Chauffeur (<u>Id_Chauf</u>, Nom, Prénom, Adresse, Sexe)

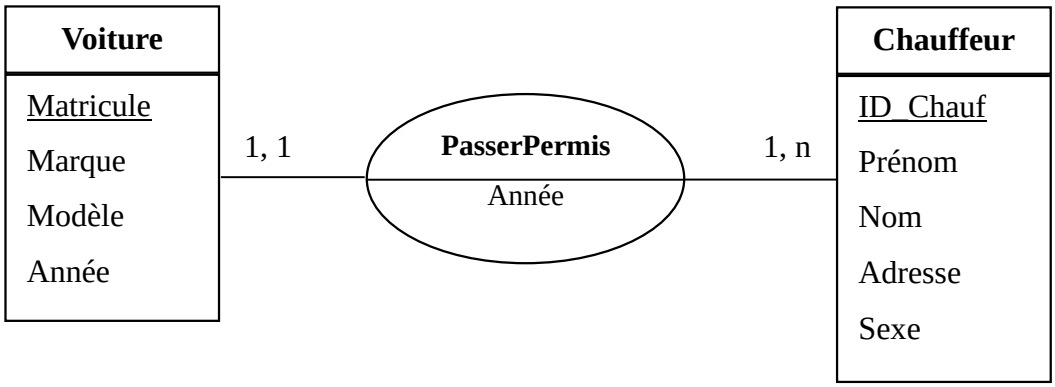
Voiture (Matricule, Marque, Modèle, Année, #Id_Chauf, AnnéePermis)

Remarque :

Il est aussi possible de les fusionner pour n'en faire qu'une seule relation qui regroupera donc tous les attributs des deux entités ainsi que les attributs éventuels de l'association.

R_2 : Pour chaque association binaire A entre E et F (représentées par les relations R_E et R_F respectivement) avec comme cardinalité (1, 1) du coté E et une cardinalité différente de (1, 1) du coté F, on inclut dans R_E la clé primaire de la relation R_F comme clé étrangère, ainsi que les attributs éventuels de l'association A.

Exemple :

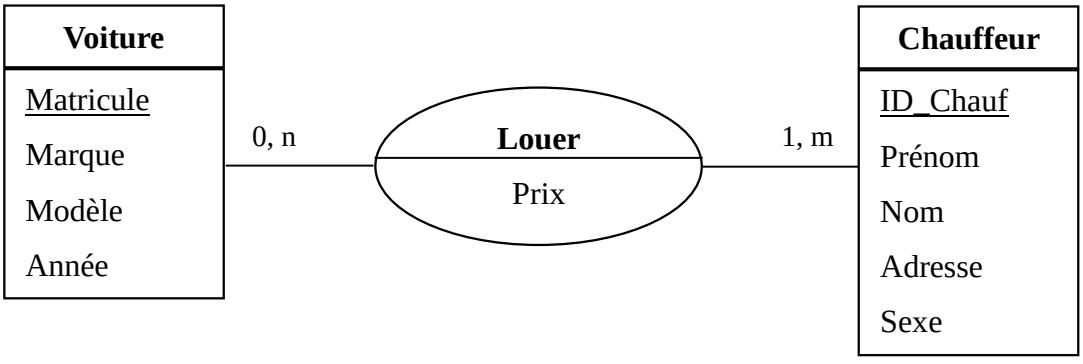


Chauffeur (Id_Chauf, Nom, Prénom, Adresse, Sexe)

Voiture (Matricule, Marque, Modèle, Année, Id_Chauf, AnnéePermis)

R_3 : Pour chaque association binaire A entre E et F avec des cardinalités différentes de (1, 1) des deux côtés, on crée les relations R_E et R_F qui représentent respectivement E et F en plus d'une relation R_A qui représente A. Les clés primaires des relations R_E et R_F migrent vers la relation R_A et y trouve les attributs éventuels de A. La clé primaire de R_A est la concaténation des clés étrangères venant des relations R_E et R_F. La relation R_A est appelée relation d'association.

Exemple :



Chauffeur (Id_Chauf, Nom, Prénom, Adresse, Sexe)

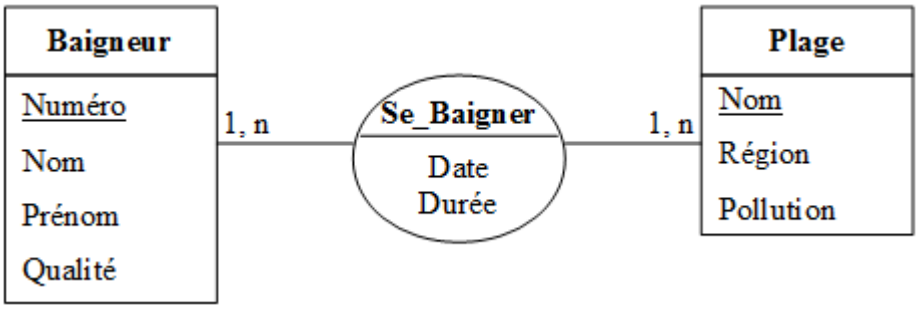
Voiture (Matricule, Marque, Modèle, Année)

Louer (Id_Chauf, Matricule, Prix)

Remarque :

Quand une association porteuse devient relation, il arrive souvent qu'un ou plusieurs de ses attributs participent à sa clé primaire en plus des clés étrangères héritées des entités qu'elle relie.

Exemple :



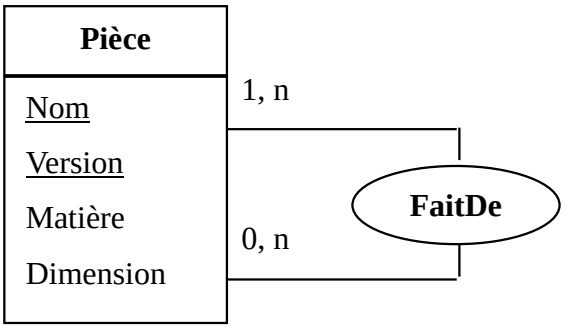
Baigneur (Numéro, Nom, Prénom, Qualité)

Plage (Nom, Région, Pollution)

Se_Baigner (#Baigneur, #Plage, Date, Durée)

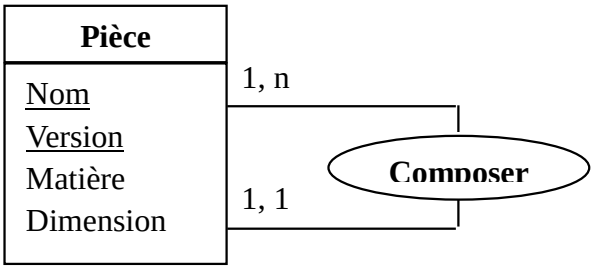
R_4 : Transformation des associations récursives : Une association récursive (réflexive) est une association binaire particulière. Cependant, sa transformation se fait en utilisant les mêmes règles que les associations binaires simples.

Exemple :



Pièce (Nom, Version, Matière, Dimension)

FaitDe (#Nom, #Version, #NomP, #VersionP)



Pièce (Nom, Version, Matière, Dimension, #NomC, #VersionC)

R_5 : Transformation des associations n-aires : Une association n-aire (n >= 3) est souvent représentée par une relation dans le modèle relationnel. Cependant, dans le cadre général, la transformation d'une telle association se fait de la même manière que les associations binaires.

IV. Règles d'intégrité

IV. 1. Règles d'intégrité structurelles

Les règles d'intégrité structurelles sont des assertions que doivent respecter le schéma et l'instance d'une base de données. Elles sont imposées par la méthode MERISE.

IV. 1. 1. Unicité des clés

Une relation est par définition un ensemble d'enregistrements sans doublon. Pour éviter d'avoir deux enregistrements similaires, on introduit la notion de clé primaire qui est composé d'un ou de plusieurs d'attributs pour lequel (lesquels) deux enregistrements différents ne peuvent pas prendre la même valeur.

IV. 1. 2. Contrainte de domaine

Les valeurs possibles d'un attribut sont restreintes à un ensemble de valeurs prédéfinies appelé domaine.

IV. 1. 3. Contrainte d'intégrité référentielle

Pour maintenir la liaison qu'a deux entités dans le modèle entité/association, on crée les contraintes d'intégrité référentielles dans le modèle relationnel. La contrainte d'intégrité référentielle est celle qui relie deux relations dont l'une a pris la clé primaire de l'autre comme clé étrangère lors du passage du modèle entité/association au modèle relationnelle.

IV. 1. 4. Valeurs nulles et clé

La valeur NULL est une valeur conventionnelle introduite dans une relation pour représenter une information inconnue ou inapplicable. Elle permet donc de résoudre le

problème des attributs facultatifs du modèle entité/association. Si l'administrateur de la base ne l'interdit pas pour un attribut, cette valeur peut être prise par tout attribut quel que soit son domaine à l'exception de la clé primaire.

Sa notation est : NULL

IV. 2. Règles provenant du cahier des charges

Les règles de gestion du système étudié peuvent introduire de nouvelles contraintes que les données doivent respecter. Ces contraintes sont spécifiques au système étudié et chaque système peut en avoir plusieurs. Elles sont mentionnées dans le cahier des charges.