Module 3

MODÈLE RELATIONNEL

SEJEN
POWERING DECISION MAKING

Introduction

- Un modèle ancien qui a fait ses preuves
  - Introduit par **Codd** dans les années **70**
  - C'est un **modèle ensembliste** simple et rigoureux
  - **représentation des données** sous **formes tabulaires**, assez naturelle pour des non informaticiens
  - basé sur des **outils mathématiques** (théorie des ensembles) particulièrement **adapté à la description de schéma conceptuel de données**.

Modèle relationnel – La découverte

- Objectifs non initialement prévus mais atteints
  - Développement de langages de manipulation des données non procéduraux basés sur des théories(mathématiques) solides
  - Algèbre relationnelle et logique
  - Modélisation et manipulation de données tabulaires mais extensibilité pour pouvoir modéliser et manipuler des données complexes
  - Développement de standard pour la description et la manipulation des données complexes.
    - Le langage SQL qui a été normalisé au niveau international
- L'avenir
  - Des extensions « objets » au modèle relationnel et au langage SQL existent ou sont en cours de développement.

Concepts de base – théorie des relations

- Théorie des relations
  - Le **modèle relationnel** est fondé sur la théorie mathématique des **relations**
  - Cette **théorie** se **construit** à partir de la **théorie des ensembles**.
- 3 notions importantes
  - Domaine (de valeurs)
  - Relation
  - Attribut
- Notions complémentaires
  - Produit cartésien
  - Tuple

Concepts de base – Domaine

- Domaine
  - **Ensemble de valeurs** (atomiques) caractérisé par un nom
    - Les **domaines** sont les **ensembles** dans lesquels les **données prennent valeur**
  - Comme un ensemble un domaine peut être défini :
    - **En extension** : on donne la **liste des valeurs qui le composent**
      - domaine des « éditeurs de BD » = {Delcourt, Les Humanoïdes Associés, Soleil, ...}
    - **En intension** : on définit le **domaine par une propriété caractéristique** que respecte les éléments appartenant au domaine.
      - domaine des « entiers positifs » = {ensemble des entiers x tel que x >= 0}

Concepts de base – Domaine et produit cartésien

- Domaine et produit cartésien
  - Associé à la notion de **domaine**, on a la notion de **produit cartésien**
    - Le produit cartésien d'un ensemble de domaines **D1, D2, ... ,Dn** , noté **D1 x D2 x ... x Dn**, est l'ensemble des vecteurs <**v1, v2, ..., vn**> tel que pour **i** variant de **1 à n**, **vi** est une valeur du domaine **Di**.
- Exemple :
  - D1 : domaine Fonte = {Courier, Arial, Times}
  - D2 : domaine ElémentTexte = {Titre 1, Titre 2, Paragraphe}

fonte	élémentTexte
Courier	Titre 1
Courier	Titre 2
Courier	Paragraphe
Arial	Titre 1
Arial	Titre 2
Arial	Paragraphe
Times	Titre 1
Times	Titre 2
Times	Paragraphe

Concepts de base – Relation, n-uplets et tuples

- Qu’est-ce qu’une relation ?
  - Une relation est un **sous-ensemble** du **produit cartésien** d'une **liste de domaines**
  - Une relation est caractérisée par un nom
  - Une relation peut être vue comme un tableau à deux dimensions dont les colonnes correspondent aux domaines.
    - On utilise le terme relation ou table
    - Les lignes d'une relation correspondent à des **n-uplets** de valeurs que l'on nomme en général **tuple**
      - un **tuple** est une ligne d'une relation qui correspond à un enregistrement;
  - Afin de permettre la définition de **plusieurs colonnes** sur le **même domaine** et de rendre **l'ordre des colonnes** sans signification, on associe un nom à chaque colonne : c'est la notion d'**attribut**.

Concepts de base – exemple de relation

- « Titre de BD » et « Editeurs de BD » suivant
  - – D1 = {Trolls dans la brume, La porte du ciel, Sueurs froides}
  - –D2 = {Soleil, Les Humanoïdes Associés, Delcourt}
- La **relation** ou **table** « Titres chez Editeurs » associant les titres et les éditeurs est la suivante :

Titre chez les Editeurs	Titre de BD	Editeur de BD
	Trolls dans la brume	Soelei
	La porte du ciel	Les Humanoïdes Associés
	Sueurs froides	Delcourt

Concepts de base – relation et attribut

- – Relation et attribut
  - Comme nous l'avons souligné précédemment afin de permettre la définition de plusieurs colonnes sur le même domaine et de rendre l'ordre des colonnes sans signification, on associe un nom à chaque colonne
    - c'est la notion d'attribut.
  - Un **attribut** est la colonne d'une **relation** caractérisée par un **nom**.

Concepts de base – Variation des valeurs

- Variation des valeurs dans le temps
  - Le **modèle relationnel** permet de décrire des données dont les **valeurs évoluent dans le temps**
    - – Des tuples vont êtres ajoutés, d'autres supprimés et certains verront certaines de leurs valeurs modifiées.
  - Néanmoins, la structure d'une relation caractérisée par les trois concepts de **domaine**, **relation** et **attribut** est invariante
    - Cette structure est capturée dans le schéma de la relation
    - Le **schéma de la relation** est le **nom de la relation** suivi de la **liste des attributs** et de la **définition de leurs domaines**.
    - Un **schéma de relation** est notée sous la forme :
      - R (A1 : D1, A2 : D2, ..., Ai : Di,..., An : Dn) avec R le nom de la relation, Ai le nom de l'attribut i et Di le domaine i.

Concepts de base – Schéma d’une relation

- Intension et extension
  - Le **schéma d'une relation** représente son **intention**, i.e. les **propriétés** (en tous les cas au moins une partie) **communes** et **invariantes** des **tuples** qu'elle contient ou va contenir
    - l'intention est une description des données
  - – Une table représente au contraire l'extension d'une relation
    - i.e. une vue des tables qu'elle contient à un moment donné
    - l'extension représente un état de la base
    - L'extension d'une relation R est également appelée **instance** de R
- Exemple de schéma de relation
  - **BD** (TITRE: String, EDITEUR: String, SCENARISTE : String, DESSINATEUR : String, NUMERO: Integer)

Règles d’intégrité – unicité d’une clé

- Règles d’intégrité
  - Ce sont les contraintes que doivent vérifiées les données de la base.
- Règle minimale : unicité des clés
  - Il ne peut exister 2 fois le même **tuple** dans une relation
    - afin d'assurer cette unicité des **tuples**, la notion de **clé** est utilisée :
      - Une **clé** est l'ensemble minimal d'attributs dont la connaissance des valeurs permet d'identifier un tuple unique de la relation considérée.
      - Toute **relation** possède au moins une **clé**, car la connaissance de tous les attributs permet d'identifier un **tuple** unique.
      - S'il existe plusieurs clés, on en choisit « arbitrairement » une qui est appelée **clé primaire**. Les autres clés possibles sont appelés **clés secondaires** ou **alternatives**.

Règles d’intégrité – autres contraintes

- 3 autres types de règles d'intégrité sont souvent ajoutés
  - les contraintes de références
    - Une **contrainte de référence** existe à chaque fois qu'une **relation comprend des attributs** définis sur le même domaine avec **un lien sémantique** avec les **attributs** d'une **autre relation**
  - les contraintes d'entité
    - Elles imposent de vérifier qu'**une clé primaire** est **unique** et ne **possède pas de valeurs « nulles »** (c'est-à-dire des valeurs indéterminées ou inconnues)
  - les contraintes de domaine
    - Elles font référence au **type de définition** des attributs. Elle concerne le **contrôle syntaxique** et **sémantique** d'une donnée.

Règles d’intégrité – dépendance fonctionnelle

- DF : Dépendance Fonctionnelle
  - Il y a **dépendance fonctionnelle** d'un attribut (ou groupe d'attributs) **A** d'une relation **R** vers un attribut (ou groupe d'attributs) **B** si à une valeur de **A** correspond au plus une valeur de **B**
    - On dit que A détermine B
    - Cette **dépendance fonctionnelle** est **noté A -> B**
  - Plus formellement, **A-> B (**lire **« A détermine B ») si et seulement si pour tous tuples t1 et t2** de la **relation R si t1[A] = t2 [A] alors t1[B] = t2[B]** avec **ti[X]** le sous-tuple de R composé des valeurs de l'attribut X**.**
- Autre formulation
  - Un attribut (ou groupe d’attributs) **B** dépend fonctionnellement d’un attribut (ou groupe d’attributs) **A**, si, étant donné une valeur de **A**, il lui correspond une valeur unique de **B** (quel que soit l’instant considéré).

Règles d’intégrité – clés, le retour

- Clés candidates, primaires et secondaires
  - Formellement la clé d'une relation est le sous-ensemble **X** des attributs d'une relation **R(A1, A2, ...,An)** tel que
    - **X -> A1, A2, ..., An**
    - Il n'existe pas de sous-ensemble **Y X⊂** tel que **Y -> A1, A2, ..., An**
      - cette dernière condition implique qu'une clé doit être la plus « **petite possible** » en nombre d'attributs, qu'il ne peut y avoir **que des attributs** réellement **utiles pour définir l'unicité du tuple**.
  - Une **clé candidate** est un **attribut** (ou **groupes d'attributs**) dont les valeurs déterminent de façon univoque chaque **tuple** d'une **relation**
  - Une **clé primaire** est la **clé candidate** choisie pour identifier chaque **tuple** d'une relation.
  - Les **clés candidates non retenues** sont appelées **clés secondaires**.

Règles d’intégrité – clé étrangère

- Clé étrangère
  - Une **clé étrangère** est un **attribut** ou un **groupe d'attributs** défini sur un **domaine primaire** et qui n'est pas une **clé primaire**.
    - Une **clé étrangère** fait **toujours référence** à une **clé primaire définie** sur le **même domaine**
    - En clair, une **clé étrangère** dans une **relation** est un **attribut** (ou un **groupe d'attributs**) qui est une **clé primaire** dans une **autre relation**
- Clés étrangères et contraintes de références
  - Une **contrainte de référence** existe à chaque fois qu'une **relation** comprend des **attributs sémantiques** définis sur le même domaine avec **lien sémantique** avec les **autres attributs** d'une **autre relation**.
    - Ce type de contrainte doit être vérifié lors de la mise à jour des données

Règles d’intégrité – Contraintes de références

- Contraintes de références / référentielles
  - Les **clés étrangères** sont un des moyens pour représenter des **contraintes de références** (ou **contraintes référentielles**).
    - Une **contrainte référentielle** est une **contrainte d'intégrité** portant sur une relation R1, consistant à imposer que la valeur d'un groupe d'attributs de R1 apparaisse comme valeur de clé dans une autre relation R2

Règles d’intégrité – Valeur nulle et entité

- Valeur nulle
  - Lors de l'insertion de tuple dans une relation il arrive que la valeur pour un attribut soit inconnu ou inapplicable.
    - On est alors amené à introduire dans la relation une valeur conventionnelle, nommée **valeur nulle**.
  - La **valeur nulle** est une valeur conventionnelle introduite dans une relation pour représenter une information inconnue ou non applicable
- Contrainte d'entité
  - La **contrainte d'entité** est une **contrainte d'intégrité** qui impose que toute relation possède une **clé primaire** et que **tout attribut participant** à cette **clé primaire** soit **non nul**.

Conception du schéma des bases

- Un objectif
  - structurer le domaine d'application pour pouvoir le représenter sous forme de types et de tables.
  - La représentation doit être
    - **juste** pour éviter les **erreurs sémantiques** (notamment dans les réponses aux requêtes)
    - **complète** pour permettre les développements des programmes d’application souhaités
    - **évolutive** pour pouvoir supporter la prise en compte rapide de nouvelles demandes, l'expression de nouveaux besoins

Démarche de conception

- Une démarche en 5 étapes
  - G.Gardarin proposent une démarche en étapes.
  - L'idée est de procéder par abstractions successives, du problème vers l'utilisateur
    - (1) **Perception du monde réel** et **capture des besoins**
    - (2) **Elaboration** du **schéma conceptuel**
    - (3) **Conception** du **schéma logique**
    - (4) **Affinement** du **schéma logique**
    - (5) **Elaboration** du **schéma physique**

Démarche de conception – Besoin utilisateur

- **Etude des problèmes des utilisateurs et compréhension de leur besoin**
  - Relève du « Génie Logiciel ». Cela fait partie des phases **recueil des besoins** et **rédaction de cahier des charges**.
    - Il consiste à construire un ou plusieurs modèle Entité-Association exprimant les vues externes de la base
  - C'est une phase particulièrement importante ; elle consiste dans des cas réels à plus qu'une simple construction d'un modèle E/A.
  - – **Remarque** : ici nous utilisons le modèle E/A. Cependant d'autres modèles existent pour cette étape.
    - Actuellement on utilise le langage de modélisation **UML** et de ce qui est nommé les **cas d'utilisation** (**Use Case** en anglais)

Démarche de conception – Schéma conceptuel

- Elaboration du schéma conceptuel
  - Il s'agit ici d'intégrer en un seul **schéma conceptuel global**, **complet**, **cohérent** et **sans redondance**, l'ensemble des schémas produits à l'étape précédente
  - Des aller-retours avec l'étape précédente sont souvent nécessaires.

Démarche de conception – Schéma logique

- Conception du schéma logique
  - Dans cette étape, on réalise la transformation du **schéma conceptuel** produit à l'étape précédente en structures de données **implémentables** sur le système choisi.
  - Cela consiste pour nous à passer du **modèle E/A au modèle relationnel**. Nous verrons la méthode de transformation dans la suite de ce cours.
- Affinement du schéma logique
  - Il faut vérifier que le schéma produit est un bon schéma
    - En première approximation un « **bon schéma** » est un schéma **sans oublis ni redondances** d'informations
  - Pour définir des « bons » schémas (i.e. des schémas respectant certaines propriétés particulières), le modèle relationnel s'appuie sur la « **normalisation** ».

Démarche de conception – Schéma physique

- Elaboration du schéma physique
  - **Définition** des **structures physiques** de stockage des données.

Passage du modèle E/A au modèle relationnel - Notation

- Notations :
  - Associations de type 1:1 (un à un)
    - aucune des cardinalités maximales n'est n.
  - Association de type 1:n (un à plusieurs)
    - une des deux cardinalités maximales est n.
  - Association de type n:m (plusieurs à plusieurs) :
    - les deux cardinalités maximales sont n

Passage du modèle E/A au modèle relationnel – Les règles

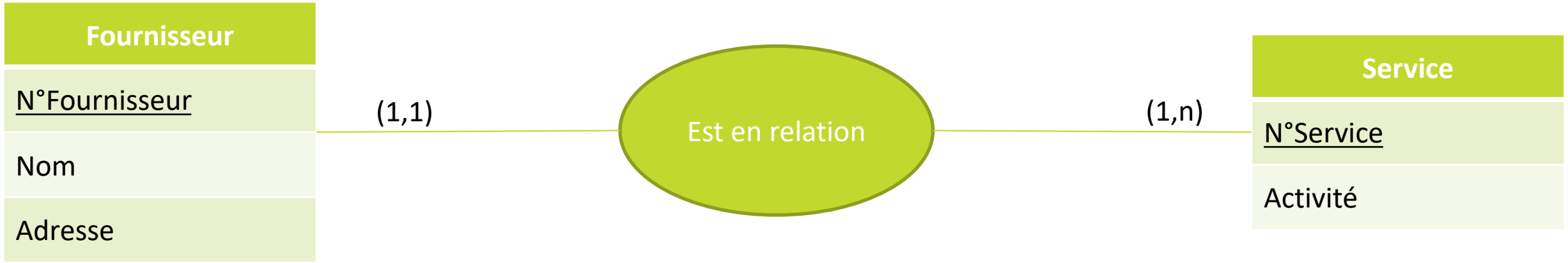
- Règle de conversion d'une entité
  - Toute entité se transforme en relation
    - Les **propriétés de l'entité** deviennent les **attributs de la relation**
    - La **clé** (ou **identifiant**) de l'**entité** devient la **clé primaire** de la **relation**.

Fournisseur
N°Fournisseur
Nom
Adresse

Fournisseur (N°Fournisseur, Nom, Adresse)

Passage du modèle E/A au modèle relationnel – Les règles

- Règle de conversion d'une association binaire (1:n)
  - L'association « se transforme » en **clé étrangère**
    - La **clé primaire** de la relation issue de l'**entité** ayant la **cardinalité maximum n** devient une **clé étrangère** dans la relation issue de l'**entité** de **cardinalité maximum 1**.

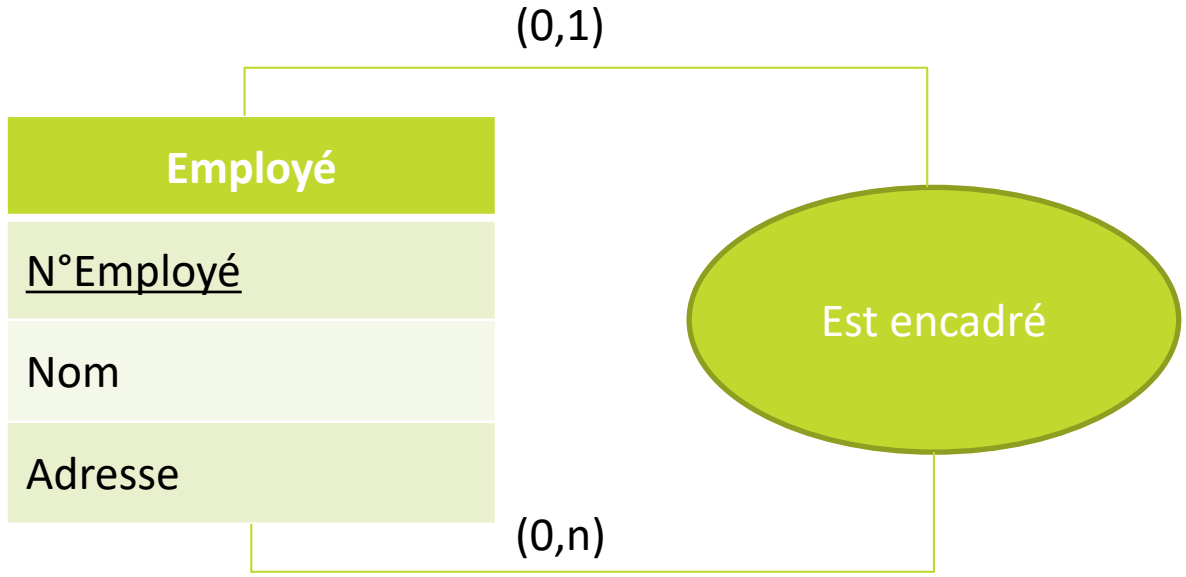


Service (N°Service, Activité)
Fournisseur (N°Fournisseur, Nom, Adresse, #N°Service)

SEJEN
POWERING DECISION MAKING

Passage du modèle E/A au modèle relationnel – Les règles

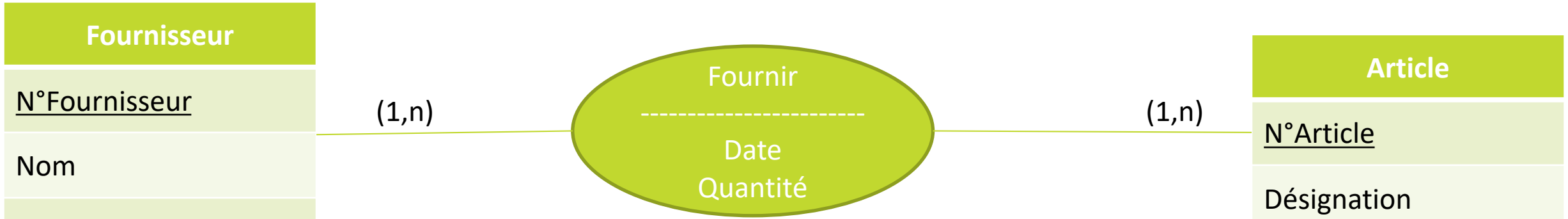
- Règle de conversion d'une association réflexive (1:n)
  - L'association « se transforme » en **clé étrangère**
  - La **clé primaire** de la relation issue de l'entité est **dupliqué** sous forme de **clé étrangère** pour traduire l'**association**.



Employé (N°Employé, Nom, Adresse, #N°Employe-Encadrant)

Passage du modèle E/A au modèle relationnel – Les règles

- **Règle de conversion d'une association binaire ou n-aire (n:n)**
  - L'association se transforme en relation
  - Les propriétés de l'association deviennent les attributs de la relation.
  - La **clé primaire** de la relation qui représente l'association est constituée par la **concaténation des clés primaires** des **relations** issues des entités liées à l'association.

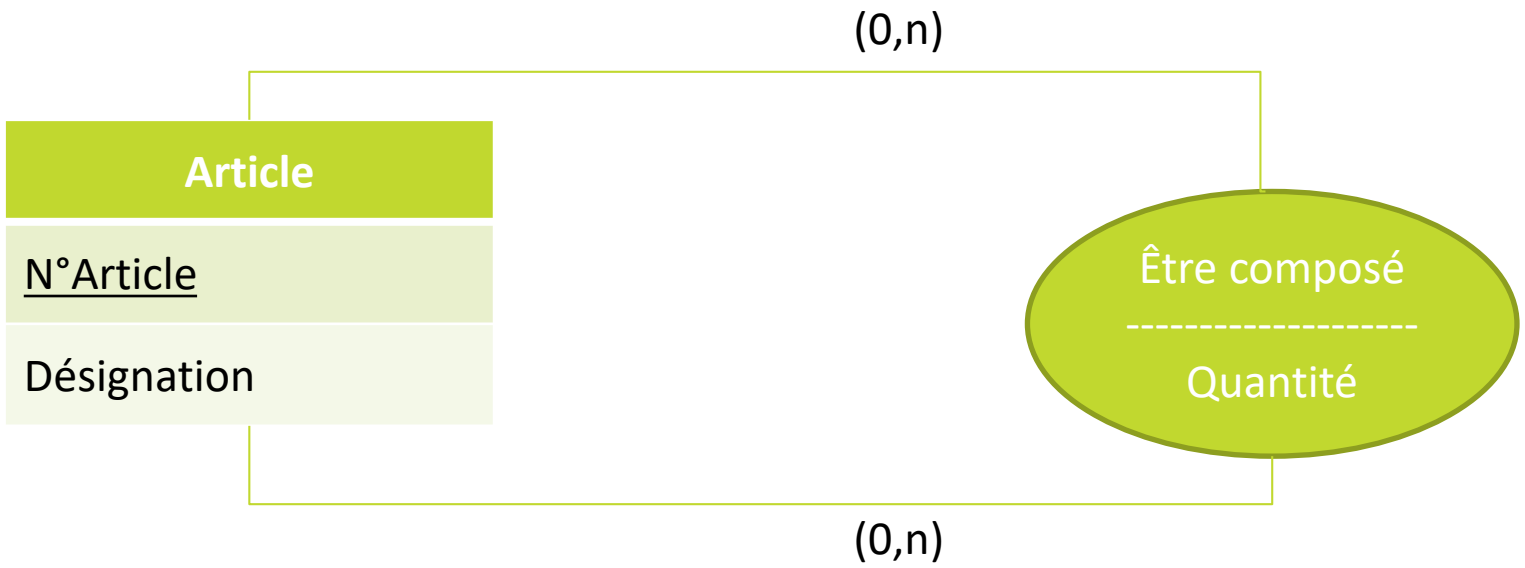


Fournisseur (N°Fournisseur, Nom, Adresse)
Article (N°Article, Désignation)
Fournir (#N°Fpurnisseur, #N°Article, Date, Quantité)

SEJEN
POWERING DECISION MAKING

Passage du modèle E/A au modèle relationnel – Les règles

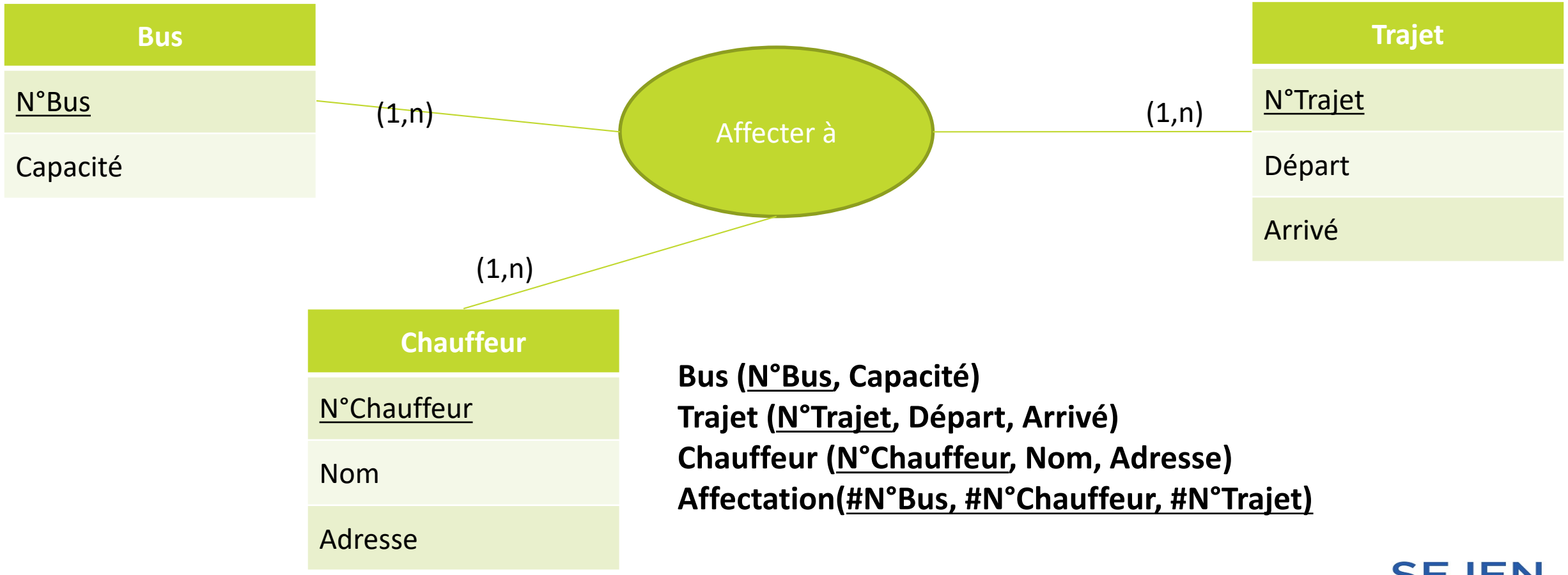
- Règle de conversion d'une association réflexive n-aire (n:n)



Article (N°Article, Désignation)
Etre composé (#N°Article-Composant, #N°Article-Composé, Quantité)

Passage du modèle E/A au modèle relationnel – Les règles

- – Règle de conversion d'une association ternaire n-aire (n:n)



Bus (N°Bus, Capacité)
Trajet (N°Trajet, Départ, Arrivé)
Chauffeur (N°Chauffeur, Nom, Adresse)
Affectation(#N°Bus, #N°Chauffeur, #N°Trajet)

SEJEN
POWERING DECISION MAKING

Problèmes de conception

- Une mauvaise conception pose des problèmes de
  - Redondance
    - L'**adresse** et le **nom** de chaque fournisseur sont répétés autant de fois qu'il y a de produits fournis par ce fournisseur.
  - Anomalies de mises à jour
    - Si l'adresse du fournisseur change, il faut la modifier pour chacune des entrées nom/adresse/produit/prix. Il peut y avoir des oublis.
  - Anomalies d'insertion
    - On ne peut insérer dans la base qu'un fournisseur ayant des produits référencés
  - Anomalies de suppression
    - Si on supprime les produits d'un fournisseur, on risque de perdre l'adresse de ce fournisseur dans la base

Normalisation – Rappels sur les DF

- Dépendance fonctionnelle
  - Soit $R(A_1,A_2, \ldots, A_n)$ un schéma de relation, et X et Y des sous-ensemble de $A_1,A_2, \ldots, A_n$. On dit que X détermine Y ou que Y dépend fonctionnellement de X si et seulement si, des valeur identiques de X implique des valeurs identiques de Y. On le note $X \rightarrow Y$.
  - Une dépendance fonctionnelle élémentaire est une dépendance fonctionnelle de la forme X → A, où A est un attribut unique n'appartenant pas à X et où il n'existe pas X' inclus au sens strict dans X (i.e. $X' \subset X$) tel que X' → A
  - Une dépendance fonctionnelle X → A est une dépendance fonctionnelle directe s'il n'existe aucun attribut B tel que l'on puisse avoir X → B et B → A.

Normalisation – Les formes normales

- Les formes normales
  - Les formes normales sont différents stades de qualité qui permettent d'éviter la redondance dans les bases de données relationnelles afin d'éviter ou de limiter : les pertes de données, les incohérences au sein des données, l'effondrement des performances des traitements.
  - La normalisation dans le modèle relationnel s'appuie sur la notion des dépendances fonctionnelles.
  - Les dépendances fonctionnelles permettent de définir les premières formes normales (1FN, 2FN, 3FN, BCFN).

Normalisation – Les formes normales

- 1FN
  - Une relation est en première forme normale si et seulement si, tout attribut contient une valeur atomique.
- 2FN
  - Une relation est en deuxième forme normale si, et seulement si, elle est en première forme normale et si toutes les dépendances fonctionnelles entre la clé et les autres attributs sont élémentaires.
- 3FN
  - Une relation est en troisième forme normale si, et seulement si elle est en deuxième forme normale et tout attribut n'appartenant pas à la clé n'est pas en dépendance fonctionnelle directe avec un ensemble d'attributs non-clé.
- BCFN
  - Une relation est en forme normale de Boyce-Codd (BCNF) si, et seulement si, les seules dépendances fonctionnelles élémentaires sont celles dans lesquelles une clé détermine un attribut non-clé.

SEJEN

POWERING DECISION MAKING