Conception de Bases de Données : Modèle Entité-Association

Introduction

Dans le monde actuel, les bases de données sont omniprésentes et essentielles au fonctionnement de nombreuses applications et systèmes d'information. Elles permettent de stocker, d'organiser et de gérer de grandes quantités de données de manière efficace et structurée. Une conception rigoureuse de ces bases de données est cruciale pour garantir la performance, la fiabilité et la maintenabilité des systèmes qui les utilisent.

Le modèle Entité-Association (E/A) est un outil puissant et largement utilisé pour la conception conceptuelle des bases de données. Il offre une approche intuitive et graphique pour représenter les données et leurs relations, facilitant ainsi la communication entre les concepteurs, les développeurs et les utilisateurs finaux.

Ce manuel a pour objectif de vous guider à travers les concepts fondamentaux du modèle Entité-Association et de vous fournir les connaissances nécessaires pour concevoir vos propres modèles E/A. À travers ce cours, vous apprendrez les étapes clés de la conception d'une base de données, les composants du modèle E/A, et les principes essentiels pour une conception efficace. Avec les exemples et illustrations fournis, vous serez en mesure de comprendre et d'appliquer ce modèle dans vos projets de conception de bases de données.

1 Conception d'une Base de Données

1.1 Objectifs de la Conception d'une Base de Données

La conception d'une base de données répond à un besoin fondamental : celui de développer des applications qui manipulent et exploitent de manière intensive de grandes masses de données. Ces données doivent être persistantes, c'est-à-dire qu'elles doivent survivre à l'exécution des programmes et être disponibles à tout moment. Les bases de données sont donc au cœur des systèmes d'information modernes.

On retrouve des bases de données dans de nombreux domaines et applications, par exemple :

- La gestion des stocks et des employés dans une entreprise.
- Les agences de voyages pour la gestion des réservations et des clients.
- Les services hospitaliers pour le suivi des patients et des dossiers médicaux.
- Les systèmes de marketing pour l'analyse des comportements des consommateurs.
- Le traitement des infractions et la gestion des dossiers judiciaires.
- Les cinémas pour la gestion des films et des séances.
- Les agences spatiales pour le stockage et l'analyse des données spatiales.
- La scolarité à l'université pour la gestion des étudiants et des cours.
- Les laboratoires de recherche pour l'organisation et l'analyse des données expérimentales.
- ... et bien d'autres encore.

1.2 Place de la Conception de BD dans le Cycle de Vie d'un Logiciel

La conception d'une base de données est une étape cruciale dans le processus de développement d'un logiciel. Elle fait partie du génie logiciel, mais se distingue du développement de programmes classique. Alors que le génie logiciel classique se concentre sur le développement des programmes et des algorithmes, la conception de base de données se focalise sur la structure et l'organisation des données.

La conception d'une base de données est donc la partie du processus de génie logiciel qui produit le schéma de la base de données.

1.3 Les Trois Étapes de la Conception

La conception d'une base de données se déroule généralement en trois étapes principales, comme illustré à la Figure **??** :

1. **Modélisation Conceptuelle** : Cette première étape consiste à comprendre et à analyser le domaine d'application et les besoins des utilisateurs. L'objectif est d'identifier les concepts fondamentaux du domaine, leurs propriétés et les relations qui les unissent. Le résultat de cette étape est un modèle conceptuel, une représentation abstraite des données, indépendante du modèle de données et du Système de Gestion de Base de Données (SGBD) qui sera utilisé. Le modèle Entité-Association est typiquement utilisé à cette étape.

2. **Modélisation Logique** : Cette étape consiste à traduire le modèle conceptuel en un schéma logique, en utilisant un modèle de données spécifique, comme le modèle relationnel, le modèle orienté-objet ou le modèle semi-structuré. Le choix du modèle de données est souvent influencé par le SGBD cible. Le schéma logique reste indépendant du SGBD spécifique.

3. **Modélisation Physique** : La dernière étape consiste à définir la structure de stockage physique des données, en tenant compte des spécificités du SGBD choisi et des performances attendues. Cela inclut la définition des index, des paramètres de configuration du SGBD, et d'autres optimisations physiques.

1.4 Difficultés de la Conception

La conception d'une base de données est une tâche complexe et souvent difficile. Plusieurs facteurs contribuent à cette complexité :

- **Complexité du domaine d'application** : Le monde réel est complexe et riche en exceptions. Il est souvent difficile de modéliser fidèlement cette complexité dans un modèle de base de données.

- **Interaction avec les experts du domaine et les futurs utilisateurs** : La conception d'une base de données nécessite une collaboration étroite avec les experts du domaine d'application et les futurs utilisateurs du système. Il est essentiel de bien comprendre leurs besoins et leurs attentes. Cela implique d'analyser et de comprendre le domaine, les besoins métier et d'identifier les données nécessaires aux traitements, tant pour les besoins actuels que futurs.

- **Abstraction et simplification** : Coder le monde réel, avec sa complexité et ses exceptions, à l'aide d'un modèle informatique sans flexibilité oblige à abstraire et à simplifier. Il faut donc trouver un juste milieu entre la fidélité au réel et la simplicité du modèle.

- **Gestion de la complexité et du volume d'informations** : Il faut maîtriser la complexité et le nombre d'informations à représenter, qui peuvent être très importants.

- **Performances** : Il est crucial de ne jamais perdre de vue les performances du système. La base de données doit être conçue de manière à garantir des temps de réponse acceptables, même avec de grandes quantités de données et un nombre important d'utilisateurs.

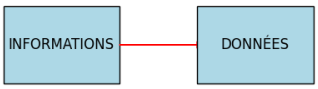

Figure 1: Passage des Informations aux Données

La Figure **??** illustre le passage des informations du monde réel aux données structurées dans une base de données. C'est ce passage complexe que la conception de base de données doit faciliter et optimiser.

2 Le Modèle Entité-Association (E/A)

2.1 Présentation du Modèle E/A

Le modèle Entité-Association (E/A) a été introduit par Peter Chen en 1976. Il propose un ensemble de concepts et de symboles graphiques pour modéliser les données d'une application de manière conceptuelle. L'objectif est de fournir une représentation claire et intuitive des données, indépendante de toute considération technique liée à l'implémentation.

Le modèle E/A repose sur trois concepts centraux :

- **Entité** : Représente un objet ou un concept du monde réel qui a une existence propre et qui est pertinent pour l'application.
- **Attribut** : Décrit une propriété ou une caractéristique d'une entité.
- **Association** : Représente une relation ou un lien entre deux ou plusieurs entités.

En plus de ces concepts fondamentaux, le modèle E/A inclut également des notions plus avancées comme :

- Les contraintes de cardinalité, qui précisent la nature et le nombre de liens entre les entités.
- Les clés et les entités faibles, qui permettent de gérer l'identification des entités.
- Quelques règles de conception pour garantir la qualité et la cohérence du modèle.

2.2 Concepts Fondamentaux

2.2.1 Entité (ou Type Entité)

Definition 2.1 (Entité)**.** Une **entité** (ou type entité) représente un ensemble d'objets qui sont centraux dans le domaine de l'application et pour lesquels des informations doivent être stockées. Une entité correspond à une catégorie d'objets ou de concepts du monde réel que l'on souhaite représenter dans la base de données.

Les entités peuvent représenter des objets concrets (comme les acteurs, les films, les clients, les produits, etc.) ou des concepts abstraits (comme les événements, les transactions, les catégories, etc.).

Example 2.2 (Exemples d'entités)**.** Dans le domaine du cinéma, on peut identifier les entités suivantes :

- **Acteur** : représente l'ensemble des acteurs de cinéma.
- **Film** : représente l'ensemble des films.
- **Tournage** : peut représenter l'ensemble des tournages de films (si l'on souhaite modéliser des informations spécifiques sur les tournages).

Graphiquement, une entité est représentée par un rectangle, contenant le nom de l'entité.

Figure 2: Entités Acteur et Film

Figure 3: Entité Tournage

2.2.2 Attribut

Definition 2.3 (Attribut)**.** Un **attribut** est une propriété ou une caractéristique descriptive d'une entité. Il sert à préciser et à qualifier une entité, en décrivant les informations que l'on souhaite stocker pour chaque instance de cette entité.

Chaque entité est décrite par un ensemble d'attributs. Un attribut est spécifié par un nom et un ensemble de valeurs possibles, appelé domaine de valeurs.

Example 2.4 (Exemples d'attributs pour l'entité Acteur)**.** Pour l'entité **Acteur**, on peut définir les attributs suivants :

- **Prénom** : le prénom de l'acteur (domaine de valeurs : chaîne de caractères).
- **Nom** : le nom de famille de l'acteur (domaine de valeurs : chaîne de caractères).
- **Date de naissance (Ddn)** : la date de naissance de l'acteur (domaine de valeurs : date).

Les attributs peuvent être de différents types :

- **Attribut simple** : un attribut dont la valeur est atomique et indivisible (ex : Nom, Prénom).
- **Attribut complexe** : un attribut dont la valeur est composée de plusieurs parties (ex : Date de naissance, qui peut être composée du jour, du mois et de l'année).

Dans un diagramme E/A, les attributs sont généralement listés à l'intérieur du rectangle représentant l'entité.

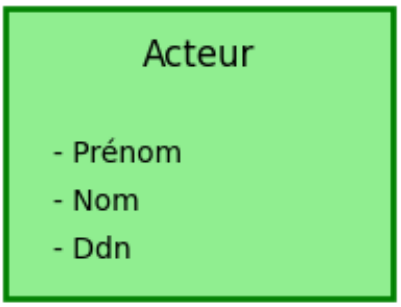

Figure 4: Attributs de l'entité Acteur

2.2.3 Association (ou Type d'Association)

Definition 2.5 (Association)**.** Une **association** (ou type d'association) représente un lien, une relation ou une interaction entre deux ou plusieurs entités. Elle permet de modéliser des liens concrets entre les instances de ces entités.

Les associations peuvent être binaires (entre deux entités), ternaires (entre trois entités), ou n-aires (entre n entités). Les associations binaires sont les plus fréquentes.

Example 2.6 (Exemple d'association binaire)**.** Dans le domaine du cinéma, on peut définir l'association **Joue** entre les entités **Acteur** et **Film**. Cette association représente le fait qu'un acteur joue dans un film.

Graphiquement, une association est représentée par un losange, relié par des traits aux entités qu'elle associe. Le nom de l'association est écrit à l'intérieur du losange.

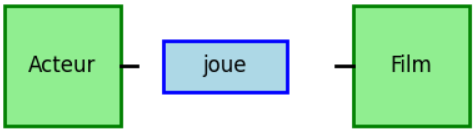

Figure 5: Association Joue entre Acteur et Film

Une association est spécifiée par un nom, les entités qu'elle associe et, éventuellement, des **rôles** et des attributs.

2.2.4 Rôles

Dans une association, chaque entité participante joue un rôle spécifique. Le rôle précise la fonction de l'entité dans le contexte de l'association. Les rôles sont particulièrement utiles pour clarifier le sens des associations, surtout lorsque plusieurs associations existent entre les mêmes entités ou lorsque l'association est réflexive (une entité est associée à elle-même).

Dans l'exemple de l'association **Joue**, on pourrait préciser les rôles : l'acteur joue le rôle de Acteur et le film a comme rôle Film dans lequel il joue. Cependant, dans cet exemple simple, les rôles sont implicites et ne sont pas obligatoires.

Dans certains cas, une association peut avoir des attributs propres. Par exemple, si l'on considère l'association **Joue** et que l'on souhaite stocker le rôle spécifique joué par un acteur dans un film, on peut ajouter un attribut **Rôle** à l'association elle-même.

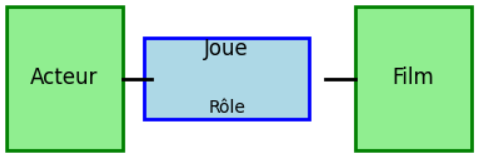

Figure 6: Association Joue avec l'attribut Rôle

2.3 Contraintes de Cardinalité

Les contraintes de cardinalité permettent de préciser la nature et le nombre de liens qui peuvent exister entre les instances des entités participant à une association. Elles apportent une précision importante au modèle E/A et permettent de refléter plus fidèlement les règles de gestion du domaine d'application.

La cardinalité s'exprime en termes de nombre minimal et maximal d'instances d'une entité qui peuvent être liées à une instance d'une autre entité via une association. On note généralement la cardinalité sous la forme **(min, max)**, où :

- **min** : le nombre minimum d'instances de l'entité cible auxquelles une instance de l'entité source doit être liée.
- **max** : le nombre maximum d'instances de l'entité cible auxquelles une instance de l'entité source peut être liée.

Les valeurs possibles pour **min** sont 0 ou 1 (ou parfois des valeurs plus grandes, mais rarement utilisées dans la modélisation conceptuelle). Les valeurs possibles pour **max** sont 1 ou **m** (pour plusieurs, many en anglais).

Example 2.7 (Exemple de contraintes de cardinalité pour l'association Joue)**.** Considérons l'association **Joue** entre **Acteur** et **Film**. On peut définir les contraintes de cardinalité suivantes :

- Un acteur peut jouer dans zéro, un ou plusieurs films. Donc, du côté de l'entité **Acteur**, la cardinalité est **(0, m)**.
- Un film peut avoir zéro, un ou plusieurs acteurs qui jouent dedans (par exemple, un film

d'animation peut ne pas avoir d'acteurs réels). Donc, du côté de l'entité **Film**, la cardinalité est également **(0, m)**.

Graphiquement, les contraintes de cardinalité sont indiquées sur les traits reliant l'association aux entités, sous la forme **(min, max)**.

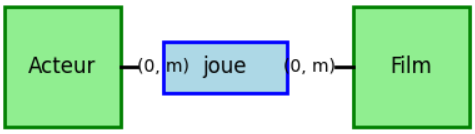

Figure 7: Contraintes de Cardinalité pour l'association Joue

2.4 Clés (ou Identifiants) d'une Entité

Definition 2.8 (Clé d'une entité)**.** Une **clé** (ou identifiant) d'une entité est un attribut (ou un ensemble d'attributs) qui permet d'identifier de manière unique chaque instance de cette entité. Elle garantit l'unicité des instances au sein d'une entité.

Il est essentiel de définir au moins une clé pour chaque entité afin de pouvoir distinguer et manipuler les instances de cette entité.

Example 2.9 (Clé pour l'entité Acteur)**.** Pour l'entité **Acteur**, on pourrait envisager plusieurs attributs comme clés potentielles :

- **Nom** : Le nom de famille seul n'est pas suffisant, car plusieurs acteurs peuvent avoir le même nom.
- **Prénom** : Le prénom seul n'est pas suffisant non plus pour la même raison.
- **Nom et Prénom** : La combinaison du nom et du prénom pourrait être une cléCandidate possible, mais il pourrait encore y avoir des homonymes.
- **Numéro d'identification unique (Num_A)** : Si l'on attribue un numéro unique à chaque acteur, cet attribut devient une cléCandidate idéale.

Dans un diagramme E/A, la clé primaire d'une entité est souvent soulignée ou indiquée d'une manière spécifique (par exemple, en la plaçant en premier dans la liste des attributs).

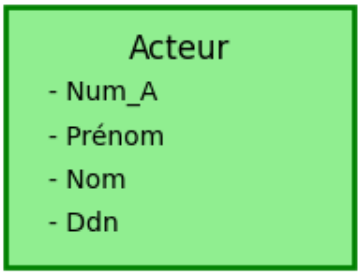

Figure 8: Clé de l'entité Acteur (Num_A)

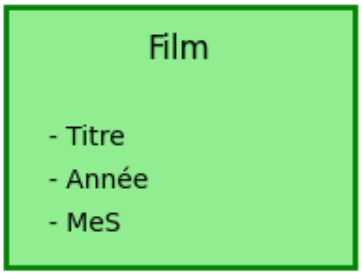

Figure 9: Clé de l'entité Film (Titre, Année)

2.5 Entités Faibles (Identification Relative)

Definition 2.10 (Entité Faible)**.** Une **entité faible** est une entité dont l'existence dépend de l'existence d'une autre entité, appelée entité forte ou entité propriétaire. Une entité faible ne peut pas être identifiée de manière unique par ses propres attributs, mais elle est identifiée relativement à l'entité forte dont elle dépend.

Les entités faibles sont souvent utilisées pour modéliser des informations qui sont des détails ou des compléments d'information relatifs à une autre entité.

Example 2.11 (Exemple d'entité faible : Salle de cinéma)**.** Considérons le domaine des cinémas. Une **Salle** de cinéma n'a pas d'existence indépendante d'un **Cinéma**. Une salle est toujours dans un cinéma. Pour identifier une salle de manière unique, il faut connaître le cinéma auquel elle appartient et son numéro de salle au sein de ce cinéma. Le numéro de salle est local au cinéma.

Dans ce cas, **Salle** est une entité faible, et **Cinéma** est l'entité forte ou propriétaire. L'identification d'une instance de **Salle** se fait de manière relative à une instance de **Cinéma**. La clé de **Salle** sera donc composée de la clé de **Cinéma** (par exemple, un identifiant unique de cinéma) et d'un attribut propre à **Salle** (par exemple, le numéro de salle).

Dans un diagramme E/A, une entité faible est souvent représentée par un rectangle avec un contour double. L'association entre l'entité faible et l'entité forte est généralement une association identifiante, souvent de cardinalité (1,1) du côté de l'entité faible.

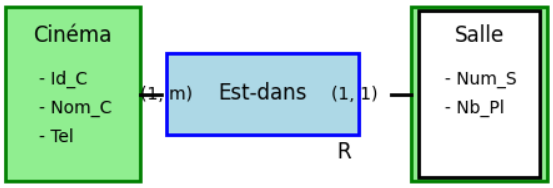

Figure 10: Entité Faible Salle et Entité Forte Cinéma

2.6 Héritage (Is-a)

Definition 2.12 (Héritage)**.** L'**héritage** (ou spécialisation/généralisation) est un concept qui permet d'organiser les entités en hiérarchies, en introduisant des relations de type est-un (is-a) entre les entités. L'héritage permet de modéliser des situations où certaines entités sont des cas particuliers ou des sous-types d'entités plus générales.

L'héritage est utile pour factoriser les attributs communs à plusieurs entités et pour représenter des hiérarchies de concepts.

Example 2.13 (Exemple d'héritage : Film, Documentaire, Animation)**.** Dans le domaine du cinéma, on peut considérer que les entités **Documentaire** et **Animation** sont des types particuliers de l'entité plus générale **Film**. On peut dire : un documentaire est un film et un film d'animation est un film. Il y a donc une relation d'héritage entre **Film** (entité super-classe) et **Documentaire** et **Animation** (entités sous-classes).

Les entités sous-classes héritent des attributs de l'entité super-classe et peuvent avoir des attributs spécifiques supplémentaires. Par exemple, l'entité **Film** peut avoir des attributs comme **Titre**, **Année**, **Metteur en scène**, tandis que **Documentaire** pourrait avoir un attribut spécifique comme **Sujet**, et **Animation** un attribut comme **Technique d'animation**.

Graphiquement, l'héritage est souvent représenté par un triangle pointant vers l'entité super-classe, avec la mention is-a ou un symbole similaire.

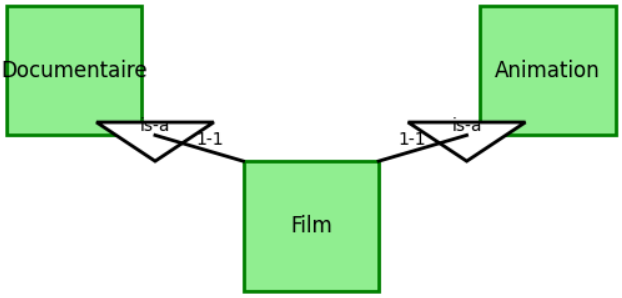

Figure 11: Héritage : Film, Documentaire et Animation

3 Principes de Conception

Une bonne conception de base de données repose sur le respect de certains principes fondamentaux. Voici quelques principes clés à suivre lors de la conception d'un modèle E/A :

3.1 Se Focaliser sur les Besoins de l'Application

Le principe le plus important est de toujours garder à l'esprit les besoins de l'application et des utilisateurs. Le modèle de données doit refléter la réalité du domaine d'application, mais il doit surtout être utile pour l'application pour laquelle il est conçu.

- **Entités et attributs pertinents** : Les entités et les attributs doivent être choisis en fonction de leur utilité pour l'application. Il est inutile d'introduire des entités ou des attributs qui ne seront pas utilisés ou qui n'apportent pas de valeur ajoutée. Par exemple, pour une application de gestion de cinémathèque, un attribut GENRE pour l'entité **Acteur** n'est probablement pas pertinent. De même, pour une application de type Amazon, stocker l'adresse de chaque acteur n'a aucun intérêt. En revanche, pour un studio de cinéma, stocker l'adresse et le portable de chaque acteur peut être important.
- **Associations significatives** : Les associations doivent refléter les relations importantes entre les entités, celles qui sont pertinentes pour l'application. Il faut s'interroger sur la nécessité de chaque association et sur sa cardinalité. Par exemple, est-il nécessaire d'avoir deux associations entre l'entité **Studio** et l'entité **Film**, l'une indiquant qu'un studio produit un film, et l'autre qu'un studio a participé au tournage ? Cela dépend des besoins de l'application.

3.2 Faire Simple

La simplicité est un principe essentiel en conception. Un modèle simple est plus facile à comprendre, à mettre en œuvre et à maintenir. Il est important de :

- **Eviter la complexité inutile** : Ne pas chercher à modéliser tous les détails et toutes les subtilités du domaine d'application. Se concentrer sur l'essentiel et simplifier au maximum.
- **Choisir des notations simples et précises** : Utiliser les notations du modèle E/A de manière rigoureuse et précise, mais sans les complexifier inutilement.

3.3 Eviter les Redondances

La redondance, c'est-à-dire la répétition de la même information sous différentes formes, est à éviter au maximum dans une base de données. La redondance entraîne des problèmes d'efficacité (gaspillage d'espace de stockage, temps d'accès plus longs) et de qualité des données (risques d'incohérences si les informations redondantes ne sont pas mises à jour de manière synchronisée).

Il faut donc s'efforcer de :

- **Identifier et éliminer les redondances** : Analyser attentivement le modèle E/A pour détecter les redondances potentielles et les supprimer. Cela peut passer par une restructuration du modèle, une décomposition des entités, ou l'introduction de nouvelles entités ou associations. Faire la chasse aux redondances est une activité importante de la conception.
- **Normalisation** : Les techniques de normalisation des bases de données, qui seront étudiées plus tard, permettent de réduire au maximum la redondance et d'améliorer la qualité des données.

3.4 Choisir Judicieusement les Entités, Associations et Attributs

Le choix des entités, des associations et des attributs est crucial pour la qualité du modèle E/A. Il faut se poser les bonnes questions et faire les bons compromis.

- **Attribut versus Entité** : Dans certains cas, on peut hésiter à modéliser une information comme un attribut ou comme une entité à part entière. Par exemple, dans l'exemple de l'association **Joue**, on pourrait considérer que le Rôle joué par un acteur dans un film est un simple attribut de l'association, ou bien le modéliser comme une entité **Rôle** liée à l'association **Joue**. Le choix dépendra de la complexité de l'information à représenter et des besoins de l'application. Si un acteur peut jouer plusieurs rôles dans un même film et si l'on souhaite décrire des informations spécifiques sur chaque rôle (comme un texte de rôle, un costume, etc.), il sera plus judicieux de modéliser **Rôle** comme une entité. Sinon, un simple attribut **Rôle** dans l'association **Joue** peut suffire.
- **Associations pertinentes** : S'assurer que chaque association modélise une relation significative et pertinente pour l'application. Eviter les associations inutiles ou ambiguës.
- **Attributs descriptifs** : Choisir des attributs qui décrivent de manière précise et complète les entités et les associations, en fonction des besoins de l'application.

4 Conclusion

Ce manuel a introduit les concepts fondamentaux de la conception de bases de données en utilisant le modèle Entité-Association (E/A). Nous avons vu que la conception d'une base de données est un processus complexe et itératif, qui nécessite une compréhension approfondie du domaine d'application, une collaboration étroite avec les experts du domaine, et le respect de principes de conception clés comme la simplicité, la pertinence et l'absence de redondance.

Le modèle E/A est un outil puissant et flexible pour la modélisation conceptuelle des données. Il permet de représenter de manière intuitive les entités, leurs attributs et les relations qui les unissent. Les diagrammes E/A facilitent la communication et la compréhension du modèle par tous les acteurs du projet (concepteurs, développeurs, utilisateurs finaux).

Bien que le modèle E/A soit largement utilisé et efficace, il existe d'autres approches pour la modélisation conceptuelle, comme le diagramme UML (Unified Modeling Language) ou les modèles sémantiques et les ontologies. Ces approches peuvent être plus adaptées à certains types d'applications ou de domaines. La recherche continue dans le domaine de la modélisation des données, comme en témoigne la conférence ER (Entity-Relationship) dédiée à ce sujet.

La conception de bases de données est un métier à part entière, qui requiert des compétences techniques, mais aussi des qualités d'analyse, de communication et de compromis. Les outils graphiques de modélisation, comme WinDesign, Looping, Visual Paradigm ou Lucidchart, peuvent faciliter la tâche du concepteur, mais ils ne remplacent pas la réflexion et l'expertise humaine.

L'étape de modélisation conceptuelle n'est que la première étape de la conception d'une base de données. Elle est suivie des étapes de modélisation logique et physique, qui consistent à traduire le modèle conceptuel en un schéma implémentable dans un SGBD spécifique. Le choix du modèle de données (relationnel, orienté-objet, semi-structuré) et du SGBD (Oracle, MySQL, PostgreSQL, etc.) dépendra des besoins de l'application, des contraintes techniques et des performances attendues.

La conception de bases de données est un domaine en constante évolution, en réponse aux nouveaux besoins des applications et aux progrès des technologies. Les bases de données NoSQL, les bases de données orientées graphes, les bases de données en mémoire, sont autant d'exemples de nouvelles approches et technologies qui viennent enrichir le paysage des bases de données. La maîtrise des principes de conception fondamentaux, comme ceux présentés dans ce manuel, reste cependant essentielle pour aborder sereinement les défis de la conception de bases de données, aujourd'hui et demain.