Analyse et Conception de Base de Données

INTRODUCTION

Une base de données informatisée est un ensemble structuré de données enregistrées sur des supports accessibles par l'ordinateur, représentant des informations du monde réel et pouvant être interrogées et mises à jour par une communauté d'utilisateurs.

Le résultat de la conception d'une base de données informatisée est une description des données. Par description on entend définir les propriétés d'ensembles d'objets modélisés dans la base de données et non pas d'objets particuliers. Les objets particuliers sont créés par des programmes d'applications ou des langages de manipulation lors des insertions et des mises à jour des données.

Cette description des données est réalisée en utilisant un modèle de données. Ce dernier est un outil formel utilisé pour comprendre l'organisation logique des données.

La gestion et l'accès à une base de données sont assurés par un ensemble de programmes qui constituent le *Système de gestion de base de données* (SGBD).

I) Notion Analyse et Conception

Afin de créer votre base de données, vous allez utiliser un langage de modélisation. Grâce à ce langage, vous pouvez non seulement décrire le domaine fonctionnel représenté dans votre base de données, mais aussi l'ensemble du système que vous êtes en train de concevoir et ses interactions avec son environnement et les utilisateurs !

1) Merise

Merise est une méthodologie de modélisation à usage général dans le domaine du développement de systèmes d'information, du génie logiciel et de la gestion de projet.

Introduit pour la première fois au début des années 1980, il était largement utilisé en France. Il a été développé et perfectionné à un point tel que la plupart des grandes organisations gouvernementales, commerciales et industrielles françaises l'ont adopté. Merise procède à un traitement séparé des données et des processus, où la vue des données est modélisée en trois étapes: de la conception à la physique en passant par la logique. De même, la vue axée sur les processus passe par les trois étapes conceptuelle, organisationnelle et opérationnelle. Ces étapes du processus de modélisation sont parallèles aux étapes du cycle de vie: planification stratégique, étude préliminaire, étude détaillée, développement, mise en œuvre et maintenance. C'est une méthode d'analyse basée sur le modèle entité-relation. En utilisant Merise, vous pouvez concevoir des tables avec des relations pour créer une base de données relationnelle.

2) UML

UML signifie « *Unified Modeling Language* ». Il s'agit d'un langage de modélisation graphique normalisé permettant de représenter les multiples aspects de la conception d'un système.

Il propose plusieurs types de diagrammes, chacun permettant de décrire les différentes facettes du système (fonctionnalité, architecture logique ou physique...).

Bien qu'UML soit destiné à décrire l'organisation d'un système avec une *approche orientée objet* (AOO), et non une base de données relationnelle, il peut servir de base à sa conception en modélisant le domaine fonctionnel (grâce au diagramme de classes).

3) Etude Comparative

- Niveaux d’abstraction

L’approche Merise : Le cycle d'abstraction permet de sérier les niveaux de préoccupations lors de la description ou de l'analyse du système. Les trois niveaux retenus correspondent à des degrés de stabilité et d'invariance de moins en moins élevés.

Le niveau conceptuel,

Le niveau logique,

Le niveau physique.

L'approche UML propose différentes notions (cas d'utilisation, paquetage, classe, composant, nœud) et différents diagrammes pour modéliser le système aux différents niveaux d'abstraction.

- Approche fonctionnelle

L'approche Merise propose une approche descendante où le système réel est décomposé en activités, elles-mêmes déclinées en fonctions. Les fonctions sont composées de règles de gestion, elles-mêmes regroupées en opérations. Ces règles de gestion au niveau conceptuel génèrent des modules décomposés en modules plus simples et ainsi de suite jusqu'à obtenir des modules élémentaires... Les limites d'une telle approche résident dans le fait que les modules sont difficilement extensibles et exploitables pour de nouveaux systèmes.
L'approche UML: Les fonctions cèdent la place aux cas d'utilisation qui permettent de situer les besoins de l'utilisateur dans le contexte réel. A chaque scénario correspond des diagrammes d'interaction entre les objets du système et non pas un diagramme de fonction.

- Dualité des données -traitements

L'approche Merise propose de considérer le système réel selon deux points de vue: un point de vue statique (les données), un point de vue dynamique (les traitements). Il s'agit d'avoir une vision duale du système réel pour bénéficier de l'impression de relief qui en résulte, et donc consolider et valider le système final.

II) Concepts Analyse et Conception

A) MCD

Le modèle conceptuel des données (**MCD**) a pour but d'écrire de façon formelle les données qui seront utilisées par le système d'information.

1) Les entités

Une entité est la représentation d'un élément matériel ou immatériel ayant un rôle dans le système que l'on désire décrire.

Prenons par exemple une Ford Fiesta, une *Renault* Laguna et une *Peugeot 306*. Il s'agit de 3 entités faisant partie d'une classe d'entité que l'on pourrait appeler *voiture*. La *Ford Fiesta* est

donc une instanciation de la classe *voiture*. Chaque entité peut posséder les propriétés *couleur*, *année* et *modèle*.

Les classes d'entités sont représentées par un rectangle. Ce rectangle est séparé en deux champs :

- le champ du haut contient le libellé. Ce libellé est généralement une abréviation pour une raison de simplification de l'écriture. Il s'agit par contre de vérifier qu'à chaque classe d'entité correspond un et un seul libellé, et réciproquement
- le champ du bas contient la liste des propriétés de la classe d'entité.

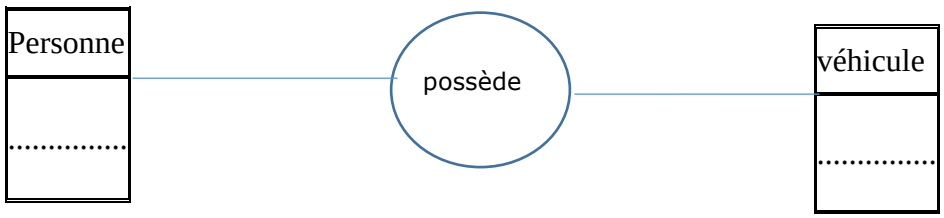
2) Les attributs

Les attributs sont les éléments d'information de base attachés à une entité. Lorsque vous générez un Modèle Physique de Données (MPD) à partir d'un Modèle Conceptuel de Données (MCD), les attributs d'entité deviennent des colonnes de tables dans le MPD.

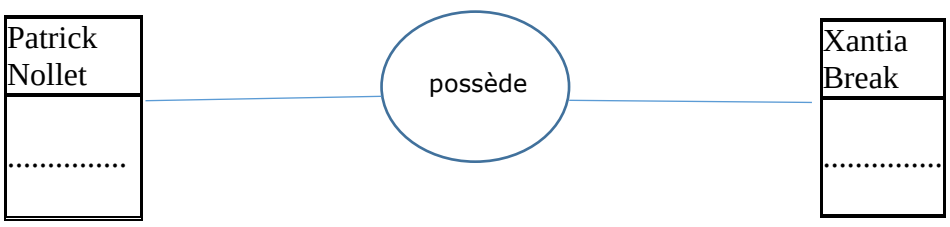
3) Les Occurrences

On définit l'occurrence ou l'instance la réalisation particulière d'une propriété d'une association ou d'une entité.

Exemple :



Une occurrence du modèle



4) Les Cardinalités

Les cardinalités permettent de caractériser le lien qui existe entre une entité et la relation à laquelle elle est reliée. La cardinalité d'une relation est composée d'un couple comportant une borne maximale et une borne minimale, intervalle dans lequel la cardinalité d'une entité peut prendre sa valeur :

- la borne minimale (généralement 0 ou 1) décrit le nombre minimum de fois qu'une entité peut participer à une relation
- la borne maximale (généralement 1 ou n) décrit le nombre maximum de fois qu'une entité peut participer à une relation

Une cardinalité 1.N signifie que chaque entité appartenant à une classe d'entité participe au moins une fois à la relation.
Une cardinalité 0.N signifie que chaque entité appartenant à une classe d'entité ne participe pas forcément à la relation.

5) Formalisme de Représentation

III) MLD

Le modèle logique des données

Il est aussi appelé modèle relationnel (lorsqu'on travaille avec une base de données relationnelle. On emploie souvent

l'abréviation suivante : MLD : Modèle logique des données Et
quelquefois, les abréviations suivantes sont employées :

- MLDR : Modèle logique de données relationnelles
- MRD : Modèle relationnel de données
- MLRD : Modèle relationnel logique de données

Le MCD (Modèle Conceptuel de Données) ne peut pas être implanté dans une base de données sans modification.

Il est obligatoire de transformer ce modèle. On dit qu'on effectue un passage du modèle conceptuel de données vers le modèle logique de données.

Le MLD pourra être implanté dans une base de données relationnelle.

Règles de passage du MCD au MLD :

Règle numéro 1 :

a) **Une entité du MCD devient une relation, c'est à dire une table.**

Dans un SGBD (Système de Gestion de base de données) de type relationnel, une table est une structure tabulaire dont chaque ligne correspond aux données d'un objet enregistré (d'où le terme enregistrement) et où chaque colonne correspond à une propriété de cet objet.

Une table contiendra donc un ensemble d'enregistrements.

Une ligne correspond à un enregistrement.

Une colonne correspond à un champ.

La valeur prise par un champ pour un enregistrement donné est située à l'intersection ligne-colonne correspondant à enregistrement-champ. Il n'y a pas de limite théorique au nombre d'enregistrements que peut contenir une table. Par contre, la limite est liée à l'espace de stockage.

b) **Son identifiant devient la clé primaire de la relation.**

La clé primaire permet d'identifier de façon unique un enregistrement dans la table.

Les valeurs de la clé primaire sont donc uniques.

Les valeurs de la clé primaire sont obligatoirement non nulles.

Dans la plupart des SGBDR (Système de Gestion de Base de Données Relationnelle), le fait de définir une clé primaire donne lieu automatiquement à la création d'un index.

c) **Les autres propriétés deviennent les attributs de la relation.**

CLIENT

numClient
nom
prénom
adresse

Exemple :

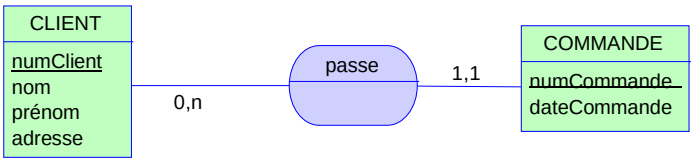
CLIENT(numClient , nom , prenom , adresse) numClient : clé primaire de la table CLIENT

numClient	Nom	Prenom	adresse
1	Dupont	Pierre	5 rue de Paris 93000 Saint-Denis
2	Durand	Raymond	68 rue Alphonse Daudet 77540 Noisy le grand
3	Dupuis	Elisa	1, boulevard Louis Blériot 94800 Villejuif
4	Dubois	Raymonde	15bis, rue de la Gaité 75014 Paris
…	…	…	…

Règle numéro 2 :

Une association de type 1:N (c'est à dire qui a les cardinalités maximales positionnées à « 1 » d'une côté de l'association et à « n » de l'autre côté) se traduit par la création d'une clé étrangère dans la relation correspondante à l'entité côté « 1 ». Cette clé étrangère référence la clé primaire de la relation correspondant à l'autre entité.

Exemple :



CLIENT(numClient , nom , prenom , adresse)

numClient : clé primaire de la table CLIENT

COMMANDE(numCommande ,dateCommande , #numClient) numCommande : clé primaire de la table COMMANDE

#numClient : clé étrangère qui référence numClient de la table CLIENT

Table CLIENT :

numClient	Nom	Prenom	adresse
1	Dupont	Pierre	5 rue de Paris 93000 Saint-Denis

2	Durand	Raymond	68 rue Alphonse Daudet 77540 Noisy le grand
3	Dupuis	Elisa	1, boulevard Louis Blériot 94800 Villejuif
4	Dubois	Raymonde	15bis, rue de la Gaité 75014 Paris
…	…	…	…

Table COMMANDE :

numCommande	dateCommande	numClient
11	1/02/2014	1
62	1/02/2014	3
423	2/02/2014	3
554	3/02/2014	2
…	…	…

Même si les tables COMMANDE et CLIENT sont 2 tables distinctes, il est possible de retrouver toutes les informations des 2 tables de façon cohérente grâce à la clé étrangère.

Exemple de questions auxquelles il est possible de répondre :

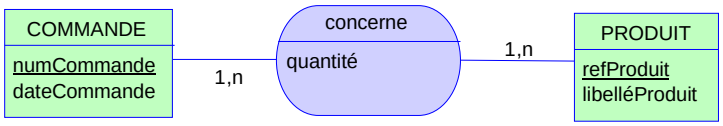
- Quel est le nom du client qui a passé la commande 11 ?
- Quels sont les noms des clients qui ont commandé le 1/02/2014 ?
- Combien de commandes a passé Elisa Dupuis ?
- Quelle est l'adresse du client qui a passé la commande 423 ?

Règle numéro 3 :

Une association de type N :N (c'est à dire qui a les cardinalités maximales positionnées à « N » des 2 côtés de l'association) se traduit par la création d'une table dont la clé primaire est composée des clés étrangères référençant les relations correspondant aux entités liées par l'association.

Les éventuelles propriétés de l'association deviennent des attributs de la relation.

Exemple :



COMMANDE(numCommande
,dateCommande) numCommande : clé

primaire de la table COMMANDE
PRODUIT(refProduit, libelleProduit)
refProduit : clé primaire de la table
PRODUIT

CONCERNE(#numCommande , #refProduit , quantité)
#numCommande , #refProduit : clé primaire composée de la table CONCERNE

#numCommande : clé étrangère qui référence numCommande de la table COMMANDE

#refProduit : clé étrangère qui référence refProduit de la table PRODUIT

Si le nom du MCD n'est pas significatif, on peut renommer le nom de la table.

Dans notre exemple, plutôt que d'appeler la table « CONCERNE », on la nommera « LIGNE_DE_COMMANDE ».

LIGNE_DE_COMMANDE (#numCommande , #refProduit , quantité)

#numCommande , #refProduit : clé primaire composée de la table CONCERNE

#numCommande : clé étrangère qui référence numCommande de la table COMMANDE

#refProduit : clé étrangère qui référence refProduit de la table PRODUIT

Table COMMANDE :

numCommande	**dateCommande**
11	1/02/2014
62	1/02/2014
423	2/02/2014
554	3/02/2014
…	…

Table PRODUIT :

refProduit	**libelleProduit**
C24	Chocolat
B12	Bière
L22	Lait
…	…

TableLIGNE_DE_COMMANDE :

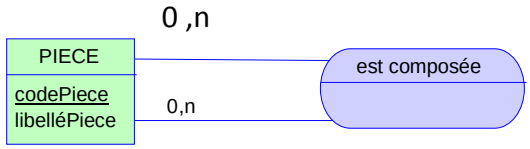
numCommande	**refProduit**	**Quantite**

11	C24	3
62	B12	3
62	C24	8
423	C24	8765
…	…	…

Associations ternaires : Les règles définies ci-dessus s'appliquent aux associations ternaires.

Associations réflexives : Les règles définies ci-dessus s'appliquent aux associations réflexives.

Exemple :



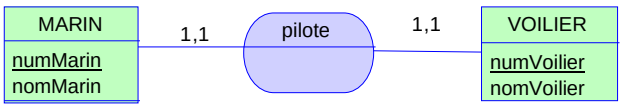
PIECE(codePiece ,libellePiece)

COMPOSITION(refProduit, libelleProduit)

COMPOSITION (#pieceComposee , #pieceComposante)

Cas particuliers : associations 1,1 : On entend par association 1,1 une association dont les cardinalités maximales sont à 1 de chaque côté

Exemple 1 : Dans le cadre d'une course à la voile en solitaire, représentez le schéma relationnel après avoir fait le schéma Entité-Relations pour les informations suivantes : numero du marin, nom du marin, numéro du voilier, nom du voilier.



Si fonctionnellement, le marin est le plus important…

MARIN(numMarin , nomMarin , numVoilier ,
nomVoilier) Clé primaire : numMarin

OU

Si fonctionnellement, le voilier est le plus important…

VOILIER(numVoilier , nomVoilier , numMarin , nomMarin)

Clé primaire : numVoilier

OU

Si le modèle peut évoluer ou si on a une distinction fonctionnelle forte entre marin et voilier…

VOILIER(numVoilier , nomVoilier , numMarin)

Clé primaire : numVoilier

Clé étrangère : numMarin qui référence numMarin de la table MARIN

et

MARIN(numMarin , nomMarin)

Clé primaire : numMarin

OU

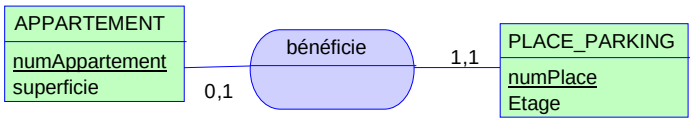
VOILIER(numVoilier , nomVoilier)

MARIN(numMarin , nomMarin , numVoilier)

Clé primaire : numMarin

Clé étrangère : numVoilier qui référence numVoilier de la table VOILIER

Exemple 2 : Dans un immeuble, un appartement peut bénéficier d'une place de parking ou pas mais jamais de plusieurs. Travail à faire : Représentez le schéma relationnel après avoir fait le schéma Entité-Relations



APPARTEMENT (numAppartement , superficie)

Clé primaire : numAppartement

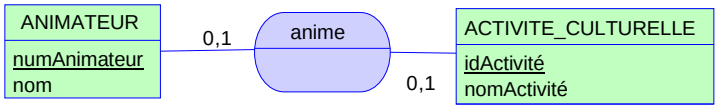
PLACE_PARKING (numPlace , Etage , numAppartement)

Clé primaire : numPlace

Clé étrangère : numAppartement qui référence numAppartement de la table APPARTEMENT

Exemple 3 : Une activité culturelle peut disposer d'un animateur ou pas mais jamais de plusieurs. Un animateur peut s'occuper au maximum d'une activité culturelle.

Travail à faire : Représentez le schéma relationnel après avoir fait le schéma Entité-Relations



Il faut évaluer l'importance de la cardinalité minimale à 0 (zéro de chaque côté).

Si le pourcentage d'animateurs qui n'animent pas est peu important, on traitera le 0 comme un 1 en plaçant une clé étrangère dans la table « Activité culturelle ».

Si le pourcentage d'activités culturelles sans animateur est peu important, on traitera le 0 comme un 1 en plaçant une clé étrangère dans la table « Animateur ».

Si le pourcentage d'animateurs qui n'animent pas est important et que le pourcentage d'activités culturelles sans animateur est important, on traitera l'association comme si les cardinalités maximales étaient à N de chaque côté. Dans ce cas, on obtient une table supplémentaire mais qui contiendra peu d'enregistrements.

ANIMATEUR (numAnimateur , nom)

Clé primaire : numAnimateur

ACTIVITE_CULTURELLE (idActivite , nomActivite)

Clé primaire : idActivite

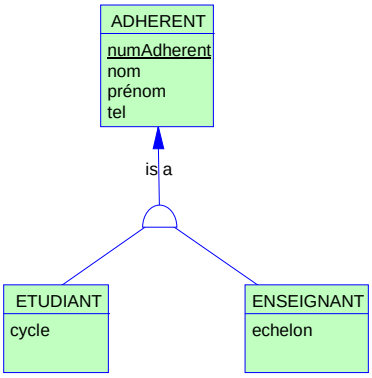
ANIMER (numAnimateur , idActivite)

Clé primaire : nimAnimateur + idActivite

Clé étrangère : numAnimateur qui référence numAnimateur de la table ANIMATEUR Clé étrangère : idActivite qui référence idActivite de la table ACTIVITE_CULTURELLE **Les héritages :**

Cas numéro 1 : La spécialisation

Exemple 1 : Les adhérents d'une bibliothèque universitaire sont des enseignants ou des étudiants.



Passage au MLD :

- L'entité mère se transforme en table
- Les entités filles se transforment en table
- L'identifiant de l'entité mère devient la clé primaire de la table qui correspond à l'entité mère et aux tables qui correspondent aux entités filles
- Les propriétés des entités se transforment en attributs des tables
- Les clés primaires des « tables filles » sont aussi des clés étrangères qui référencent la clé primaire de la « table mère »
- Un champ est ajouté dans la « table mère » pour permettre de typer les occurrences, c'est-à-dire d'identifier quelle est la « table fille » concernée.

ADHERENT (numAdherent , nom, prénom, tel, type)

Clé primaire : numAdherent

ETUDIANT (numAdherent , cycle)

Clé primaire : numAdherent

Clé étrangère : numAdherent qui référence numAdherent de la table ADHERENT

ENSEIGNANT (numAdherent , echelon)

Clé primaire : numAdherent

Clé étrangère : numAdherent qui référence numAdherent de la table ADHERENT

I. SQL

SQL (Structured Query Langage) **est** un langage de **base de données** relationnelle. Il permet : La création de **base** et des tables. L'ajout d'enregistrements sous forme de lignes

(C'est-à-dire faire des manipulations de nos table)

Les instructions SQL sont regroupées en catégories en fonction de leur utilité et des entités manipulées

1) Le langage de définition de données (LDD, ou *Data Définition Language*, soit DDL en anglais)

Est un langage orienté au niveau de la structure de la base de données. Le LDD permet de créer, modifier, supprimer des objets. Il permet également de définir le domaine des données (nombre, chaîne de caractères, date, booléen…) et d'ajouter des contraintes de valeur sur les données. Il permet enfin d'autoriser ou d'interdire l'accès aux données et d'activer ou de désactiver l'audit pour un utilisateur donné.

Les instructions du LDD sont : CREATE, ALTER, DROP, AUDIT, …

2) Le langage de manipulation de données (LMD, ou *Data Manipulation Language*, soit DML en anglais)

Est l'ensemble des commandes concernant la manipulation des données dans une base de données. Le LMD permet l'ajout, la suppression et la modification de lignes, la visualisation du contenu des tables et leur verrouillage.

Les instructions du LMD sont : INSERT, UPDATE, DELETE, SELECT.

Ces éléments doivent être validés par une transaction pour qu'ils soient pris en compte.

3) Le langage de protections d'accès (ou *Data Control Language*, soit DCL en anglais)

S'occupe de gérer les droits d'accès aux tables.

Les instructions du DCL sont : GRANT, REVOKE.

```
CREATE TABLE etudiant
(ID         INTEGER         NOT NULL PRIMARY KEY,
 nom        CHAR(20)        NOT_NULL),
  prenom      CHAR(60)        NOT_NULL,
  datee       DATE            NOT NULL DEFAULT CURRENT_DATE
telrphone       CHAR(20)    ) ;
```

```
INSERTION DANS UN TABLE

INSERT INTO client (prenom, nom, ville, age)
 VALUES
 (‘1’,DOUDOU, LO, 'DAKAR', 24),
 (‘2’,MOUHAMED, ‘DIOP’, 'LOUGA', 36),
 (‘3’,'MAME DIARRA', 'DIOP', 'touba', 18),
 (‘4’,'MADY', 'NDAYE', 'DAKAR', 68);

SELECTION DNS UN TABLE VOICI QUELQUE EXAMPLE
```

1. La liste des nom et prénom des clients.

```
SELECT nom,prenom FROM client
```

2. Liste de tout les clients

   SELECT * FROM client

3. La liste des clients habite au dakar

```
SELECT * FROM client WHERE ville="dakar"
```

4. La liste des clients leurs age <=18ans

```
SELECT * FROM client WHERE age<= 16.
```

La commande DELETE en SQL permet de supprimer des lignes dans une table. En utilisant cette commande associé à WHERE il est possible de sélectionner les lignes concernées qui seront supprimées.

```
DELETE FROM client WHERE ID = 3
```