Module : analyse et conception des bases de données :

# Documentation sur MCD, MLD et SQL :

## Modèle conceptuel des données

Le modèle conceptuel des données (**MCD**) a pour but d'écrire de façon formelle les données qui seront utilisées par le système d'information. Il s'agit donc d'une représentation des données, facile à comprendre, permettant de décrire le système d'information à l'aide d'entités.

### Entités et classe d'entité

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

On appelle **classe d'entité** un ensemble composé d'entités de même type, c'est-à-dire dont la définition est la même.

Le classement des entités au sein d'une classe s'appelle *classification*. Une entité est une *instanciation* de la classe. Chaque entité est composée de propriétés, données élémentaires permettant de la décrire.

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.

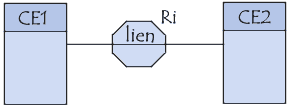
|  |
| --- |
| Libellé |
| Liste des propriétés   * ……. * ……. * ……. * ……. |

### Relations et classes de relation

Une relation (appelée aussi parfois *association*) représente les liens  
sémantiques qui peuvent exister entre plusieurs entités. Une classe de relation  
contient donc toutes les relations de même type (qui relient donc des entités  
appartenant à des mêmes classes d'entité). Une classe de relation  
peut lier plus de deux classes d'entité. Voici les dénominations des classes  
de relation selon le nombre d'intervenants :

* une classe de relation **récursive** (ou *réflexive*) relie la même classe d'entité
* une classe de relation **binaire** relie deux classes d'entité
* une classe de relation **ternaire** relie trois classes d'entité
* une classe de relation **n-aire** relie n classes d'entité

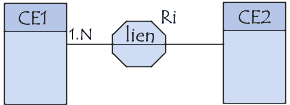
Les classes de relations sont représentées par des hexagones (parfois des ellipses) dont l'intitulé  
  
décrit le type de relation qui relie les classes d'entité (généralement un verbe).  
On définit pour chaque classe de relation un identificateur de la forme Ri permettant de désigner  
de façon unique la classe de relation à laquelle il est associé.



On peut éventuellement ajouter des propriétés aux classes de relation.  
La cardinalité

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



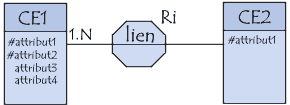
Une cardinalité 1.N signifie que chaque entité appartenant à une classe d'entité participe au moins une fois à la relation.  
  
Une cardinalité 0.N signifie que chaque entité appartenant à une classe d'entité ne participe pas forcément à la relation.

### Les identifiants

Un identifiant est un ensemble de propriétés (une ou plusieurs) permettant de désigner une et une seule entité. La définition originale est la suivante :

L'identifiant est une propriété particulière d'un objet telle  
qu'il n'existe pas deux occurrences de cet objet pour lesquelles cette propriété  
  
pourrait prendre une même valeur.

Les attributs d'une classe d'entité permettant de désigner de façon unique chaque  
instance de cette entité sont appelés *identifiants absolus*.  
  
Le modèle conceptuel des données propose de faire précéder d'un *#* les identifiants (parfois de les souligner).



Ainsi, chaque classe d'entité doit posséder au moins un attribut identifiant, et l'ensemble de ses attributs identifiants doivent être renseignés à la création de l'entité.

### Agrégation (ou identification relative)

Lorsqu'un identifiant est constitué uniquement d'attributs intrinsèques à une entité, c'est-à-dire ne faisant référence à aucune autre entité, on le nomme **identifiant absolu**.

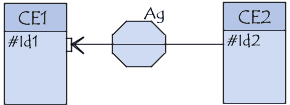
Les entités comportant des identifiants absolus peuvent être définies indépendamment des autres occurrences d'entités, on dit que ces entités sont indépendantes.

Certaines entités ne peuvent toutefois être identifiées que par l'intermédiaire  
d'autres entités, c'est la raison pour laquelle on parle d'**identification relative**.

Ainsi, l'**agrégation** (appelée aussi *identification relative*) permet  
de spécifier qu'une entité est nécessaire pour en identifier une autre.

* la classe d'entité permettant d'identifier est appelée *classe d'entité agrégeante*
* la classe d'entité identifiée est appelée *classe d'entité agrégée*

La représentation de ce type de relation est la suivante :



## Le modèle logique des données

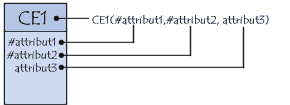
Le modèle logique des données consiste à décrire la structure de données utilisée sans faire référence à un langage de programmation. Il s'agit donc de préciser le type de données utilisées lors des traitements.

Ainsi, le modèle logique est dépendant du type de base de données utilisé.

## Le modèle relationnel

### Traduction d'une classe d'entité

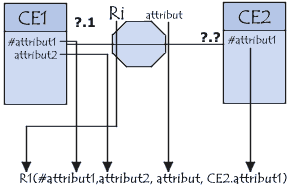
Chaque classe d'entité du modèle conceptuel devient une table dans le modèle logique. Les identifiants de la classe d'entité sont appelé *clés de la table*, tandis que les attributs standards deviennent des attributs de la table, c'est-à-dire des colonnes.



### Traduction d'une classe de relation

Le passage du modèle conceptuel au modèle logique au niveau des classes de relation se fait selon les cardinalités des classes d'entité participant à la relation :

* si une des classes d'entités possède une cardinalité faible :  
  la table aura comme attributs, les attributs de la classe ayant une cardinalité faible, puis le (ou les) attribut(s) de relation et enfin les attributs de la seconde classe précédé du nom de la classe
* si les deux classes d'entités possèdent une cardinalité forte :  
  la table aura comme attributs, les attributs des deux classes de relation précédés des noms des classes respectives, puis le (ou les) attribut(s) de relation



### Traduction d'une classe d'agrégation

Dans le cas de la présence d'une classe d'agrégation, la classe d'entité agrégée a comme attributs supplémentaires les attributs de la classe d'entité agrégeant.

# SQL: Qu'est-ce que SQL?

SQL (Structured Query Langage) est un langage de base de données relationnelle. Il permet:

1. La création de base et des tables.
2. L'ajout d'enregistrements sous forme de lignes.
3. L'interrogation de la base.
4. La mise à jour.
5. Le changement de structure de la table: ajout, suppression de colonnes.
6. La gestion de droits d'utilisateurs de la base.

La version la plus connue sur le Web est MySQL, une implémentation gratuite qui s'utilise notamment avec PHP, mais SQL est aussi le langage de nombreux autres logiciels de base de données parmi lesquelles PostgreSQL, Oracle, DB2, Access et SQL Server...

Les principales commandes sont:

* CONNECT pour se connecter à une base de données.
* CREATE pour créer une nouvelle base ou une table.
* INSERT pour ajouter des données.
* SELECT pour faire une requête sur le contenu.

Il est possible en SQL de faire des programmes procéduraux avec des itérations et des conditions.

On peut accéder à une base de données en passant des commandes comme on le fait en PHP, ou avec un logiciel visuel tel que phpMyAdmin fonctionnant sur le serveur ou localement avec XAMPP ainsi qu'avec de nombreux autres logiciels serveurs locaux.

Dans ce tutoriel, on utilisera MySQL avec PHP ainsi que l'interface phpMyAdmin.