DOCUMENTATION SUR MCD MLD ET SQL

Sonatel Academy
Fatou Niang

INTRODUCTION

Le **système d'information** ou SI peut être défini comme étant l'ensemble des moyens humains, matériels et immatériels mis en œuvre afin de gérer l'information au sein d'une unité une entreprise par exemple.

I) Notion Analyse et Conception

Merise

Merise est une méthode française née dans les années 70, développée initialement par Hubert Tardieu. Elle fut ensuite mise en avant dans les années 80, à la demande du Ministère de l'Industrie qui souhaitait une méthode de conception des SI.

MERISE est donc une méthode d'analyse et de conception des SI basée sur le principe de la séparation des données et des traitements. Elle possède un certain nombre de modèles (ou schémas) qui sont répartis sur 3 niveaux :

- Le niveau **conceptuel**,
- Le niveau **logique** ou **organisationnel**,
- Le niveau **physique**.

UML

UML, c’est l’acronyme anglais pour « Unified Modeling Language ». On le traduit par « Langage de modélisation unifié ». La notation UML est un **langage visuel** constitué d’un ensemble de schémas, appelés des **diagrammes**, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour **représenter** le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d’être effectuées par le logiciel, etc.

UML est né de la fusion des trois méthodes qui ont influencé la modélisation objet au milieu des années 90 : OMT, Booch et OOSE.

UML est surtout utilisé lorsqu’on prévoit de développer des applications avec une démarche objet (développement en Java, en C++, etc.).

Types de diagrames UML

-**Diagrammes UML structurels** (Diagramme de classe, Diagramme de composants, Diagramme de deploiement, Diagramme d’objet, Diagramme de paquetages);

-**Diagrammes UML comportementaux**(Diagramme d’activites, Diagramme de communication, Diagramme d'interaction, Diagramme de sequence, Diagramme d’etat-transition, Diagramme de temps, Diagramme de cas d’utilisation);

Etude comparative entre UML et MERISE

MERISE (Méthode d’Etude et de Réalisation Informatique pour les Systèmes d’Entreprise) est une méthode d'analyse et de réalisaton des systèmes d'information qui est élaborée en plusieurs étapes: schéma directeur, étude préalable, étude détaillée et la réalisation.
Alors que UML (Unified Modeling Langage), est un langage de modélisation des systèmes standard, qui utilise des diagrammes pour représenter chaque aspect d'un systèmes c’est-à-dire : statique, dynamique,....en s'appuyant sur la notion d'orienté objet qui est un véritable atout pour ce langage.

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. Il s'agit donc d'une représentation des données, facilement compréhensible, permettant de décrire le système d'information à l'aide d'entités. Le MCD est basé sur deux notions principales : les **entités** et les **associations**, d'où sa seconde appellation : le **schéma Entité/Association**.

A) MCD

-Entite

Chaque entité est unique et est décrite par un ensemble de propriétés encore appelées attributs ou caractéristiques. Une des propriétés de l'entité est l'identifiant. Cette propriété doit posséder des occurrences uniques et doit être source des dépendances fonctionnelles avec toutes les autres propriétés de l'entité. Bien souvent, on utilise une donnée de type entier qui s'incrémente pour chaque occurrence, ou encore un code unique spécifique du contexte.

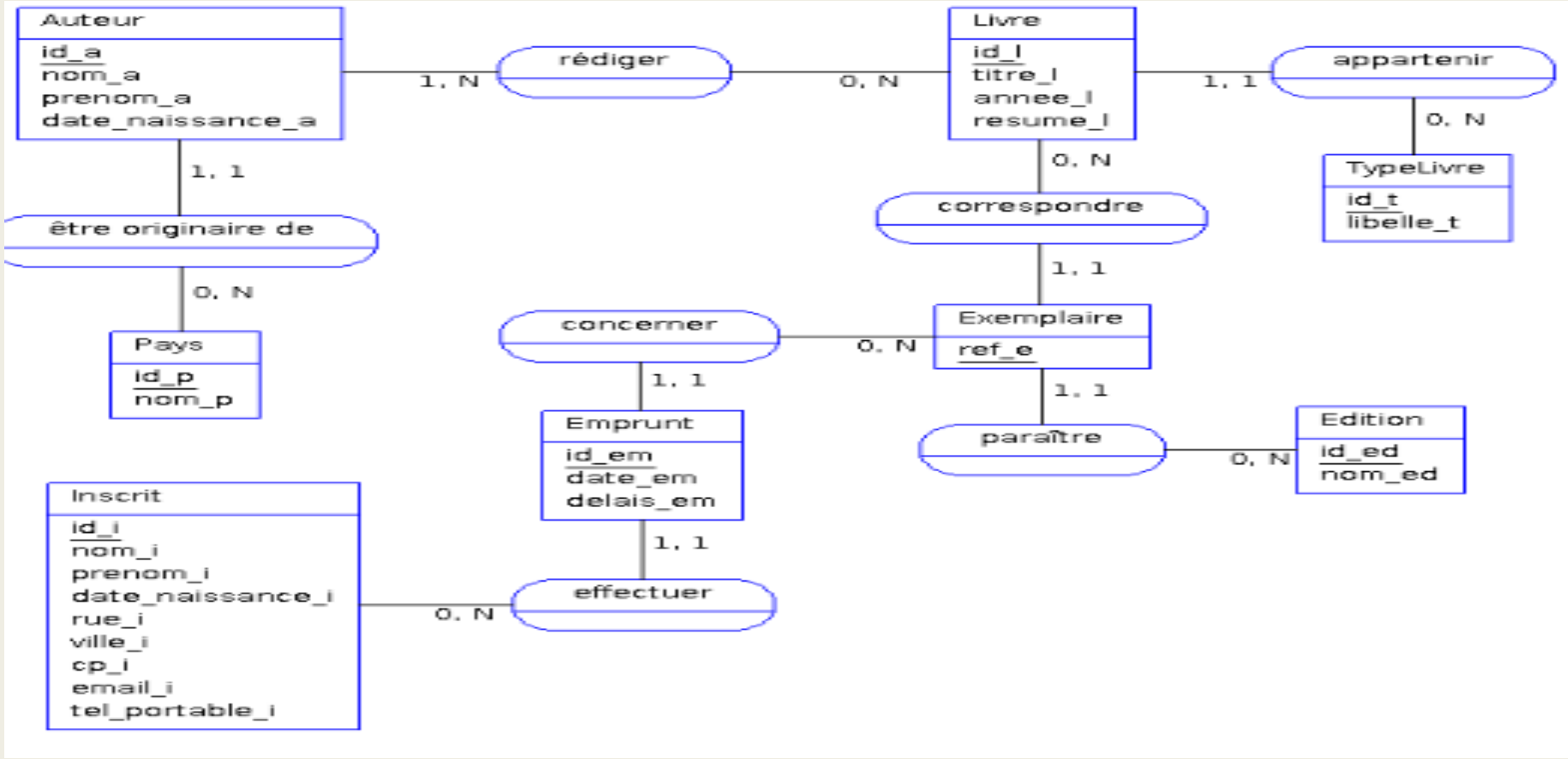
A) MCD

- **Les attributs :** Elles decrivent chaque entite, sesont les donnees de chaque entite;
- **Les Occurrences**: sont parfois appelés tuples. Par ailleurs, la table d'occurrence peut être comparée à l'instance d'une relation (implantation relationnelle d'une entité ou association) à un moment donné
- **Les Cardinalités**

De manière plus générale, les cardinalités d’une entité dans une association expriment le nombre de fois qu’une occurrence de cette entité peut être impliquée dans une occurrence de l'association, au minimum et au maximum.

Les cardinalités traduisent des règles de gestion. Ce sont des règles propres à l'organisation étudiée, qui sont décidées par les gestionnaires et décideurs. Ces règles expriment des contraintes sur le modèle.

Formalisme de Représentation



B) MLD

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.

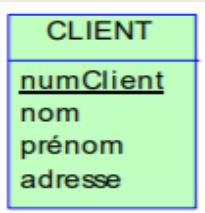
Règles de passage du MCD au MLD

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. Un index est un fichier interne au SGBD. L'utilisateur standard n'a pas besoin d'y accéder. L'index a pour but d'accélérer les traitements de recherche, de tri, de filtre et notamment sur les tables avec de nombreux enregistrements. La contrepartie est que l'index nécessite de l'espace mémoire et surtout, les temps d'insertion, de suppression d'enregistrements sont plus importants car il faut mettre à jour à la fois la table et l'index.

Règles de passage du MCD au MLD

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



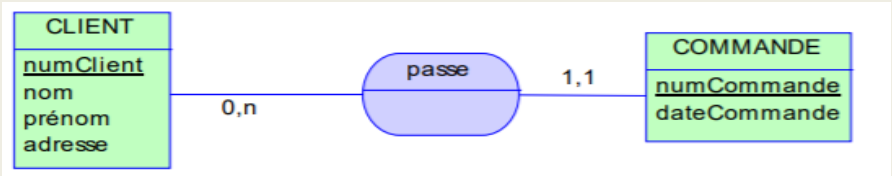
CLIENT(numClient , nom , prenom , adresse)
numClient : clé primaire de la table CLIENT

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
...	...	...	...

Règles de passage du MCD au MLD

- **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.



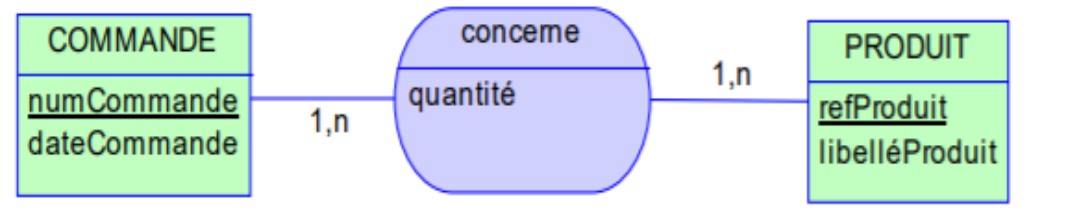
CLIENT(numClient , nom , prenom , adresse) numClient : clé primaire de la table CLIENT
COMMANDE(numCommande ,dateCommande , #numClient)

numCommande : clé primaire de la table COMMANDE #numClient : clé étrangère qui référence numClient de la table CLIENT

Règles de passage du MCD au MLD

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.



IV) SQL

Langage de définition des données (LDD)

Le langage de définition de données (LDD, ou *Data Definition 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

- **CREATE** : Il est utilisé pour créer une nouvelle table dans la base de données.
- **DROP**: Il est utilisé pour supprimer à la fois la structure et l'enregistrement stockés dans la table.
- **ALTER**: Il est utilisé pour modifier la structure de la base de données. Ce changement pourrait consister soit à modifier les caractéristiques d'un attribut existant, soit probablement à ajouter un nouvel attribut.
- **RENAME**: sert a renommer une table
- **TRUNCATE**: Il est utilisé pour supprimer toutes les lignes de la table et libérer l'espace contenant la table.

Langage de manipulation de données (LMD)

Les commandes DML sont utilisées pour modifier la base de données. Il est responsable de toute forme de modifications dans la base de données.

La commande de DML n'est pas validée automatiquement, ce qui signifie qu'elle ne peut pas enregistrer de manière permanente toutes les modifications dans la base de données. Ils peuvent être annulés.

- **INSERT** : instruction INSERT est une requête SQL. Il est utilisé pour insérer des données dans la ligne d'une table.
- **UPDATE** : Cette commande est utilisée pour mettre à jour ou modifier la valeur d'une colonne dans la table.
- **DELETE** : Il est utilisé pour supprimer une ou plusieurs lignes d'une table.

Langage d'interrogation de données (LID)

- **SELECT** : C'est la même chose que l'opération de projection de l'algèbre relationnelle. Il est utilisé pour sélectionner l'attribut en fonction de la condition décrite par la clause WHERE.
- **Jointure** : Les jointures en SQL permettent d'associer plusieurs tables dans une même requête. Cela permet d'exploiter la puissance des bases de données relationnelles pour obtenir des résultats qui combinent les données de plusieurs tables de manière efficace.
- **Projection** : Une projection est une instruction permettant de sélectionner un ensemble de colonnes dans une table. Soit la table *VOITURE* suivante :La sélection de toutes les colonnes de la table se fait par l'instruction :

```
SELECT * FROM VOITURE
```

Langage d'interrogation de données (LID)

- **Sous requête :**Effectuer une sous-requête consiste à effectuer une requête à l'intérieur d'une autre, ou en d'autres termes d'utiliser une requête afin d'en réaliser une autre (on entend parfois le terme de *requêtes en cascade*). Une sous-requête doit être placée à la suite d'une clause *WHERE* ou *HAVING*, et doit remplacer une constante ou un groupe de constantes qui permettraient en temps normal d'exprimer la qualification. Lorsque la sous-requête remplace une constante utilisée avec des opérateurs classiques, elle doit obligatoirement renvoyer une seule réponse (une table d'une ligne et une colonne).
- **Alias (AS)** est utilisé pour donner un nom d'alias à une table ou une colonne, qui peut également être une table d'ensemble de résultats. Ceci est très utile en cas de requêtes volumineuses ou complexes. L'alias est principalement utilisé pour donner un nom d'alias court pour une colonne ou une table avec des noms complexes

Langage d'interrogation de données (LID)

- **Commande**

- UNION : La commande UNION de SQL permet de mettre bout-à-bout les résultats de plusieurs requêtes utilisant elles-même la commande SELECT. C’est donc une commande qui permet de concaténer les résultats de 2 requêtes ou plus. Pour l’utiliser il est nécessaire que chacune des requêtes à concaténer retournes le même nombre de colonnes, avec les mêmes types de données et dans le même ordre.

- GROUP BY La clause Group by est utilisée pour regrouper les résultats d'une SELECT requête en fonction d'une ou plusieurs colonnes. Il est également utilisé avec les fonctions SQL pour regrouper le résultat d'une ou plusieurs tables.

Langage d'interrogation de données (LID)

- **Commande**

- ORDER BY : est utilisée avec l' SELECT instruction pour organiser les données récupérées dans l'ordre trié. L' **ordre par** la clause par défaut trie les données récupérées dans l' ordre croissant . Pour trier les données par ordre décroissant, le DESC mot clé est utilisé avec la clause Order by.

- HAVING: La condition HAVING en SQL est presque similaire à WHERE à la seule différence que HAVING permet de filtrer en utilisant des fonctions telles que SUM(), COUNT(), AVG(), MIN() ou MAX().

Langage d'interrogation de données (LID)

- **Fonction**

SQL fournit de nombreuses fonctions intégrées pour effectuer des opérations sur les données. Ces fonctions sont utiles lors de l'exécution de calculs mathématiques, de concaténations de chaînes, de sous-chaînes, etc. Les fonctions SQL sont divisées en deux catégories,

- **Fonctions d'agrégation** renvoient une seule valeur après avoir effectué des calculs sur un groupe de valeurs. Voici quelques-unes des fonctions Aggregrate fréquemment utilisées.

- **DISTINCT -**
- **COUNT:** Count renvoie le nombre de lignes présentes dans le tableau en fonction d'une condition ou sans condition.

exemple:
```
SELECT COUNT(name) FROM Emp WHERE salary = 8000;
```

- **LIMIT**

Langage d'interrogation de données (LID)

- LIKE est utilisée dans la condition de la requête SQL avec la WHERE. LIKE compare les données avec une expression utilisant des opérateurs génériques pour faire correspondre le modèle donné dans la condition.Il existe deux opérateurs génériques utilisés dans la LIKE.

•**Signe de pourcentage %** : représente zéro, un ou plusieurs caractères.

•**Signe de soulignement _** : ne représente qu'un seul caractère.

```
SELECT * FROM Student WHERE s_name LIKE 'A%';
```

- IN & NOT IN : Les opérateurs IN, NOT IN dans SQL sont utilisés avec les instructions / requêtes SELECT, UPDATE et DELETE pour sélectionner, mettre à jour et supprimer uniquement des enregistrements particuliers dans une table qui remplissent la condition donnée dans la clause WHERE et les conditions données dans les opérateurs IN, NOT IN. C'est-à-dire qu'il filtre les enregistrements d'une table selon la condition. La syntaxe des opérateurs SQL IN et NOT IN est donnée ci-dessous.

Langage d'interrogation de données (LID)

- **BETWEEN** L’opérateur BETWEEN est utilisé dans une requête SQL pour sélectionner un intervalle de données dans une requête utilisant WHERE. L’intervalle peut être constitué de chaînes de caractères, de nombres ou de dates. L’exemple le plus concret consiste par exemple à récupérer uniquement les enregistrements entre 2 dates définies.

- **AVG :** Moyenne renvoie la valeur moyenne après l'avoir calculée à partir des valeurs d'une colonne numérique

```
SELECT avg(salary) from Emp;
```

Langage d'interrogation de données (LID)

- **MIN:**La fonction MIN renvoie la valeur minimale d'une colonne sélectionnée de la table.

```
SELECT MIN(column_name) from table-name;
```

- **MAX :**La fonction MAX renvoie la valeur maximale de la colonne sélectionnée du tableau.

```
SELECT MAX(column_name) from table-name;
```

- SUM La fonction SUM renvoie la somme totale des valeurs numériques des colonnes sélectionnées.

```
SELECT SUM(column_name) from table-name;
```

Les outils

- **JMerise**