Merise

La méthode Merise permet (entre autres) de conceptualiser les données. Cette méthode permet :

- De se poser les bonnes questions.
- De s'assurer avec le client qu'on a bien compris son besoin.
- De simplifier le travail en équipe pour communiquer sur la gestion des données.
- De limiter la dette technique en perdant un peu de temps sur la conception.

Les étapes :

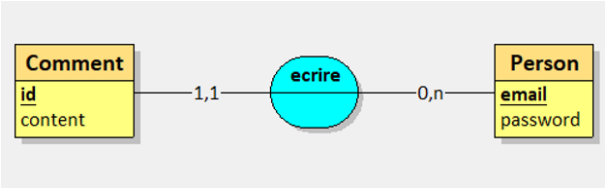
- Expression des besoins
- MCD
- MLD
- MPD

Logiciel : Looping, SQL PowerArchitect, SAP powerDesigner, MySQL Workbench, DBDesigner, AnalyseSI.

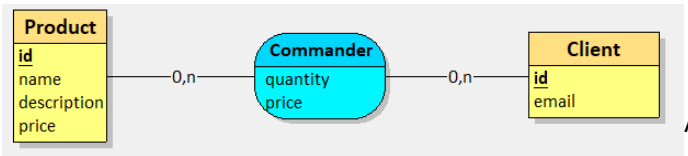
Modèle conceptuel de données (MCD)

Le MCD représente des entités (classes persistées en BDD) et les associations qui les lient.

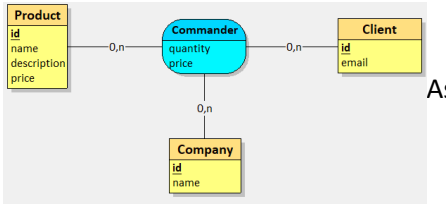
- Les entités sont représentées dans un cadre avec leurs attributs persistés en BDD (nom, code, type, longueur, obligatoire, primaire).
  La clef primaire y est soulignée : Clé primaire.
- Les associations sont nommée et représentées par un ovale.
  Certaines associations (porteuses de données) possèdent des attributs d'association.
- Les liens sont représentés par une ligne sur laquelle figure une cardinalité (0,1 / 1,1 / 1,N / N,N).



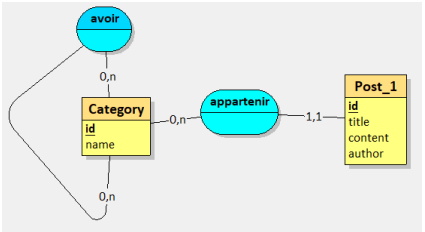
Association binaire



Association porteuse de données



Association ternaire



Association réflexive

A partir du MCD, on peut construire le modèle logique de données (MLD).

Modèle Logique de données (MLD)

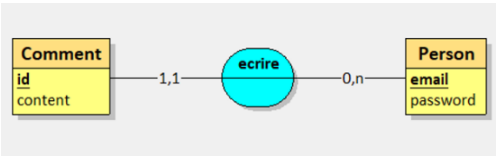
Pour obtenir le modèle logique de données à partir du MCD, on doit :

- Transformer les entités en tables.
- Transformer les attributs en champs.
- Écrire sur une nouvelle ligne le nom de table et ses champs entre parenthèses.
- Appliquer pour chaque association la règle qui lui convient.

➢ Règle pour la cardinalité 1/N : relation « one to many » ou « many to one »

Si on a 1 comme cardinalité maximum d'un côté et N de l'autre, il faut **ajouter une clef étrangère** précédée de # (du côté de la cardinalité 0,1 ou 1,1) **qui pointe vers la clef primaire** de l'autre table (du côté de la cardinalité 0,N ou 1,N).

Exemple :

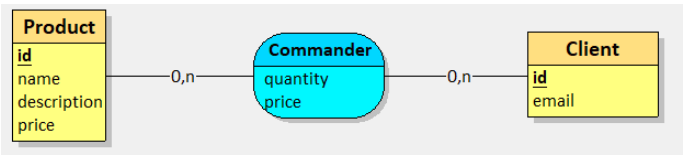


MLD
person(email, password)
comment(id, content, #peron_email)

➢ Règle pour cardinalité N/N : relation « many to many »

Si on a N comme cardinalité maximum des deux côtés, il faut **ajouter une table intermédiaire** (table de jointure) contenant deux clefs **qui pointent vers les clefs primaires** de chaque table.
Ce couple de deux clefs peut constituer une clef primaire de cette table de jointure.
La table de jointure peut être porteuse de données.

Exemple :



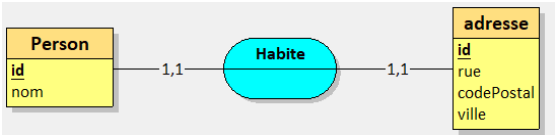
MLD
product(id, name, description, price)
commande(product_id, client_id, quantity, price)
client(id, email)

➢ Règle pour cardinalité 1/1 : relation « one to one »

Dans ce cas, il y a deux options possibles et il convient de bien réfléchir afin de choisir la plus pertinente.

- On fusionne les deux tables en gardant la table qui est fonctionnellement la plus importante et en y intégrant les champs de l'autre table (sauf l'id).

  Exemple :



MLD
person((id, nom, rue, codePostal, ville)

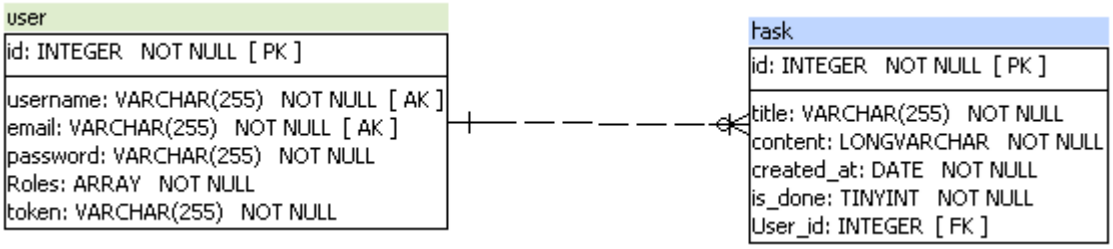
- En cas de distinction fonctionnelle forte ou d'évolution possible, on procède comme pour la cardinalité 1/N en ajoutant une clef étrangère dans une table pointant sur la clef primaire de l'autre table.

A partir du MLD, on peut :

- Construire le modèle physique de données (MPD).
- Écrire les requêtes SQL pour créer la base de données.

Modèle physique de données

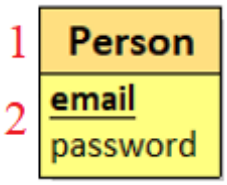
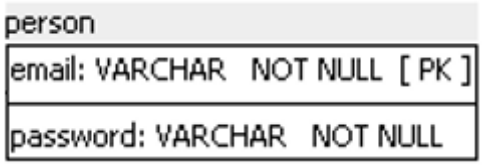
Le modèle physique de données se construit à partir du MLD. Il représente la structure réelle de la base de données.



Remarque :

- PK : primary key
- FK : foreigner key
- AK : contrainte d'unicité

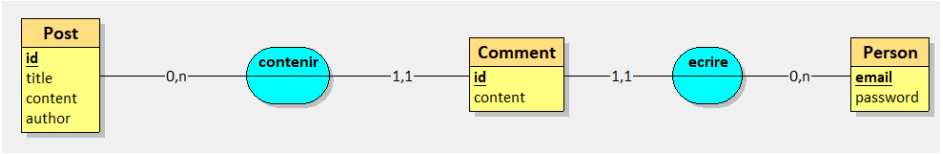
Attention le MPD ressemble grandement au MCD, mais il faut bien noter ce qui les différencie :

	MCD	MPD
		
1	Entité (classe **du code**) qui se note MyENtity.	Table **de la BDD** qui se note my_table.
2	Attributs de la classe qui seront persistés en BDD.	Champs (colonne) représentant la structure de la BDD.
	Une instance de l'entité ⟺	Une ligne de la BDD (tulpe)
	Ne contient pas les clefs étrangères	Contient les clefs étrangères (application des règles)

Exercice 1 :

Une application, publiant des articles, permet à un utilisateur qui s'identifie avec un email et un mot de passe de les commenter. Les articles se composent d'un titre, d'un contenu et d'un auteur.
En utilisant la méthode Merise, construire la base de données de cette application.

Corrigé



MCD

MLD

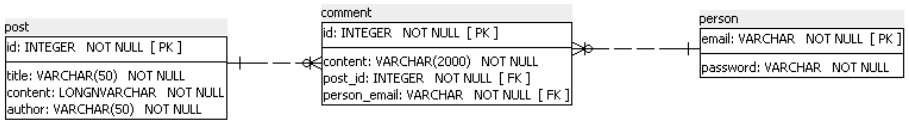
Post(id *INT*, title *VARCHAR(50)*, content *TEXT*, author *VARCHAR(50)*);
Person(email *VARCHAR(50)*, password *VARCHAR(50)*);
Comment(id *INT*, content *VARCHAR(2000), #person_email, #post_id*);

script SQL

```
CREATE TABLE Post(
  id INT,
  title VARCHAR(50) NOT NULL,
  content TEXT NOT NULL,
  author VARCHAR(50) NOT NULL,
  PRIMARY KEY(id)
);

CREATE TABLE Person(
  email VARCHAR(50),
  password VARCHAR(50) NOT NULL,
  PRIMARY KEY(email)
);

CREATE TABLE Comment(
  id INT,
  content VARCHAR(2000) NOT NULL,
  email VARCHAR(50) NOT NULL,
  id_1 INT NOT NULL,
  PRIMARY KEY(id),
  FOREIGN KEY(email) REFERENCES Person(email),
  FOREIGN KEY(id_1) REFERENCES Post(id)
);
```

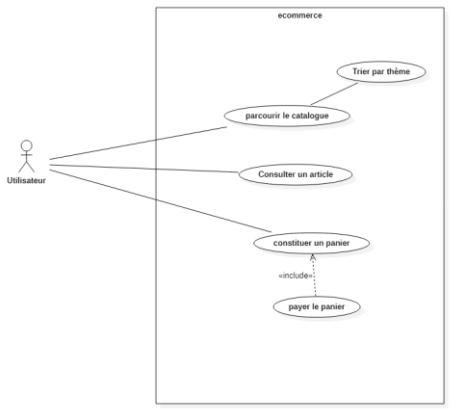


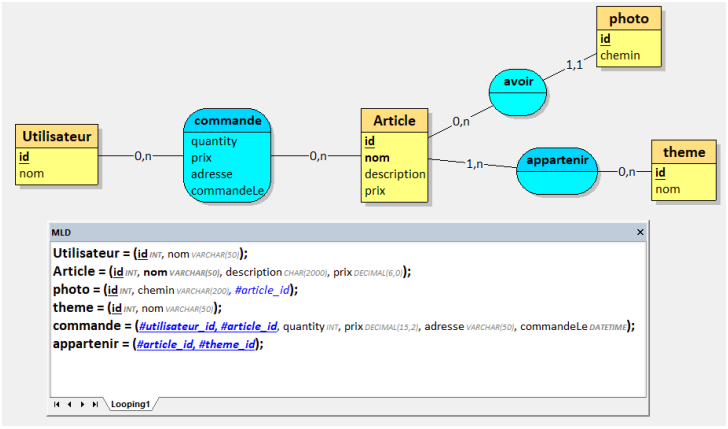
MPD

Exercice 2 :

Un client souhaite créer un site internet où les utilisateurs pourront consulter et acheter des articles.
Tous les articles auront un nom, une description, un prix et éventuellement une ou plusieurs photos.
Il y aurait un menu dans lequel les articles seraient classés par thèmes. Par exemple, une balançoire serait dans le thème « jardin » et « jeux pour enfants ».

1) Faire le diagramme de cas d'utilisation UML de ce site en se limitant aux informations données.
2) Réfléchir à la conception de la base de données de ce site en construisant un MCD et un MLD, pour obtenir finalement le MPD.
3) Créer la base de données dans un SGBD.





Script SQL

```
CREATE TABLE Utilisateur(
  id INT,
  nom VARCHAR(50),
  PRIMARY KEY(id)
);

CREATE TABLE Article(
  id INT,
  nom VARCHAR(50) NOT NULL,
  description CHAR(2000),
  prix DECIMAL(6,0),
  PRIMARY KEY(id),
  UNIQUE(nom)
);

CREATE TABLE photo(
  id INT,
  chemin VARCHAR(200),
  id_1 INT NOT NULL,
  PRIMARY KEY(id),
  FOREIGN KEY(id_1) REFERENCES Article(id)
);

CREATE TABLE theme(
  id INT,
  nom VARCHAR(50),
  PRIMARY KEY(id)
);

CREATE TABLE commande(
  id INT,
  id_1 INT,
  quantity INT,
  prix DECIMAL(15,2),
  adresse VARCHAR(50),
  commandeLe DATETIME NOT NULL,
  PRIMARY KEY(id, id_1),
  FOREIGN KEY(id) REFERENCES Utilisateur(id),
  FOREIGN KEY(id_1) REFERENCES Article(id)
);

CREATE TABLE appartenir(
  id INT,
  id_1 INT,
  PRIMARY KEY(id, id_1),
  FOREIGN KEY(id) REFERENCES Article(id),
  FOREIGN KEY(id_1) REFERENCES theme(id)
);
```