

## 1. L'Analyse : Le Rôle Clé

- **Concept** : Tu es le pont entre le besoin du client (souvent flou) et la solution technique (précise). Ton but est de **comprendre, modéliser** et **valider**.
  - **Mindset examen** : Lis attentivement l'énoncé. Chaque phrase du "client" cache souvent une règle de gestion (cardinalité) ou une donnée.
- 

## 2. Cas d'Utilisation (Use Cases - UML)

Ce diagramme décrit **qui fait quoi** (fonctionnalités).

### Théorie Express

- **Acteur** : Entité externe (Personne ou autre système) qui interagit avec le système.
  - *Primaire (gauche)* : Initie l'action pour atteindre un but (ex: Client).
  - *Secondaire (droite)* : Sollicité par le système pour aider à réaliser le use case (ex: Serveur SMS, Système bancaire externe).
- **Use Case (Cas d'utilisation)** : Une fonctionnalité majeure, libellée par un **Verbe à l'infinitif + Complément** (ex: "Commander produits", pas juste "Commande").

### ⚠ Les Erreurs à NE PAS faire (Use Cases)

- **Confondre "Système" et "Acteur"** : Ta base de données ou ton logiciel ne sont PAS des acteurs. L'acteur est toujours *externe* au rectangle du système.
  - **Le Use Case "Login" tout seul** : "Se connecter" est rarement un but en soi. C'est souvent une **pré-condition** écrite dans la description textuelle, pas une bulle isolée (sauf si l'énoncé insiste lourdement dessus).
  - **Décrire des traitements internes** : Ne mets pas "Vérifier mot de passe" ou "Calculer montant" comme Use Case. Ce sont des étapes internes du système, pas des actions de l'utilisateur.
- 

## 3. UI / UX & Ergonomie

Tu devras probablement dessiner ou critiquer une interface.

### Théorie Express

- **UI (Interface) vs UX (Expérience)** : L'UI c'est ce qu'on voit (boutons, couleurs). L'UX c'est ce qu'on ressent (facilité, frustration).

- **Ergonomie (Utilisabilité) :** L'interface doit être efficace (moins de clics), efficiente et satisfaisante.

#### Critères de réussite (Checklist examen)

- **Navigation :** L'utilisateur doit toujours savoir où il est (fil d'ariane, titre) et comment revenir en arrière.
  - **Feedback :** Après une action (ex: sauvegarde), le système DOIT confirmer ("Sauvegarde réussie").
  - **Cohérence :** Même place pour les boutons "Retour" sur tous les écrans, mêmes couleurs.
  - **Composants adaptés :**
    - *Liste déroulante* pour un choix unique dans une longue liste (ex: Pays).
    - *Radio buttons* pour un choix unique parmi peu d'options (2-4) (ex: Homme/Femme).
    - *Checkbox* pour choix multiples.
- 

#### 4. Modèle Conceptuel de Données (MCD - Merise)

C'est souvent la partie la plus pondérée. Il s'agit de structurer les données statiques.

##### Théorie Express

- **Entité :** Un objet de gestion (ex: Client, Produit).
- **Association :** Un lien entre entités (ex: Acheter, Appartenir).
- **Identifiant :** Souligné. Propriété unique (ex: NumClient, pas Nom).
- **Cardinalités (Min, Max) :**
  - **0,1** : Optionnel, au max 1.
  - **1,1** : Obligatoire, toujours 1.
  - **0,n** : Optionnel, plusieurs possibles.
  - **1,n** : Obligatoire, au moins 1.

#### Le "Musée des Erreurs" (MCD Anti-Patterns)

Ces erreurs coûtent très cher à l'examen. Relis-les deux fois.

1. **Mettre une Clé Étrangère dans le MCD : INTERDIT.**

- *Erreur* : Mettre idClient comme propriété dans l'entité Facture.
- *Correction* : C'est l'association (le trait) entre Client et Facture qui porte cette information. Les clés étrangères n'apparaissent qu'au MLD.

## 2. La boucle temporelle (Unicité de l'association) :

- *Problème* : Si un Client achète un Produit. Si tu mets juste une association "Acheter", il ne peut l'acheter qu'une seule fois dans sa vie (car le couple idClient+idProduit est unique).
- *Correction* : Si l'action peut se répéter, il faut ajouter la Date ou l'heure **dans l'identifiant de l'association** ou transformer l'association en entité (ex: Commande).

## 3. L'attribut égaré (Mauvais placement) :

- *Erreur* : Mettre DateFacture dans l'association entre Client et Facture.
- *Règle* : Une propriété dans une association dépend de **toutes** les entités liées. Ici, la date dépend de la Facture unique, pas du couple Client-Facture. Elle doit être dans l'entité Facture.

## 4. Confusion Statique vs Dynamique :

- *Erreur* : Créer une entité "Validation" ou "Envoi".
- *Correction* : Le MCD stocke des données (Noms, Dates, Montants), pas des actions. Les actions sont gérées par les cas d'utilisation ou des changements d'état (ex: un attribut Statut dans Commande).

## 5. Propriété vs Entité :

- *Erreur* : Mettre "Pays" comme simple attribut texte si le système doit gérer une liste déroulante stricte.
- *Correction* : Créer une entité Pays et une association Habiter.

## Les Patterns utiles (Solutions types)

- **Historisation** : Si tu dois garder l'historique des prix, ne remplace pas le prix dans Produit. Crée une entité Prix liée à Produit avec une DateDebut.
- **Hiérarchie** : Un employé est dirigé par un autre employé. C'est une **association réflexive** (une boucle sur l'entité Employé).

---

## 5. Modèle Logique de Données (MLD - Relationnel)

Transformation du MCD en tables pour la base de données.

## Règles de transformation (Recette de cuisine)

1. **Entité -> Table** : Chaque entité devient une table. L'identifiant devient **Clé Primaire (PK)**.
2. **Association (1,n) - (1,1) ou (0,n) - (1,1)** :
  - C'est la règle du "Père et du Fils". Le côté "Plusieurs" (n) porte la clé étrangère.
  - *Exemple* : Un Client (1,n) a des Factures (1,1). La table Facture reçoit idClient comme **Clé Étrangère (FK)** (Not Null car 1,1).
3. **Association (0,n) - (0,n) (Many-to-Many)** :
  - On crée une **nouvelle table** de jointure.
  - Sa clé primaire est composée des deux clés étrangères (ex: idCommande, idProduit) + attributs portés par l'association (ex: Quantité).

## Derniers conseils pour demain

- **Vocabulaire** : Utilise les termes précis (Entité, Occurrence, Cardinalité, Clé Primaire).
- **Validation Croisée** : Vérifie que chaque champ de ton écran UI (ex: liste des pays) a bien une source de données dans ton MCD (entité Pays).

Respire un bon coup, relis bien les cardinalités, et ça va aller !

---

## Le Musée des Erreurs (MCD Patterns)

Ce sont des erreurs de conception structurelles qui montrent que tu n'as pas compris la philosophie "Conceptuelle" de Merise. Si tu en fais une, tu perds souvent la totalité des points de la question.

### 1. L'Intrus Logique (La Clé Étrangère dans le MCD)

C'est l'erreur la plus fréquente et la plus pénalisée.

- **L'erreur** : Mettre un champ idClient ou refProduit comme propriété à l'intérieur d'une entité Commande.
- **Pourquoi c'est faux** : Dans un MCD, les liens se font par les **traits** (associations), pas par les attributs. Les clés étrangères (FK) n'existent pas encore à ce stade, elles n'apparaissent qu'au niveau du MLD (Base de données).
- **L'exemple illustré** :

- **✗ Faux** : Entité Facture avec propriétés : idFacture, Date, Montant, idClient.
- **✓ Juste** : Entité Facture avec propriétés : idFacture, Date, Montant. Et une association (trait) vers l'entité Client.

## 2. La Boucle Temporelle (Non-Répétabilité de l'Association)

- **L'erreur** : Utiliser une association simple pour une action qui peut se répéter dans le temps entre les deux mêmes objets.
- **Le piège** : Un couple d'identifiants dans une association est **unique**. Si tu relies Client et Produit par l'association Acheter, le client Bob ne pourra acheter le Produit X qu'**une seule fois dans sa vie**.
- **L'exemple illustré** :
  - **✗ Faux** : Client -(0,n)----(Acheter)----(0,n)- Produit.
  - **✓ Juste (Option 1 - Date)** : Ajouter la propriété DateHeure comme **identifiant** de l'association Acheter (pour rendre le triplet unique).
  - **✓ Juste (Option 2 - Entité faible)** : Transformer l'association en une entité Commande ou LigneCommande. C'est souvent la solution préférée car on peut y stocker plus d'infos.



## 3. L'Attribut Égaré (Mauvais placement)

- **L'erreur** : Placer une propriété dans une association alors qu'elle dépend d'une seule entité (ou l'inverse).
- **La règle (Règle de la dépendance fonctionnelle)** : Pour qu'un attribut soit dans une association, sa valeur doit dépendre de la combinaison des entités liées.
- **L'exemple illustré** :
  - **✗ Faux** : Mettre DateFacture dans l'association Concerner entre Client et Facture. La date dépend uniquement de la facture, pas du client !
  - **✓ Juste** : DateFacture doit être une propriété de l'entité Facture.
  - *Contre-exemple* : Quantité est bien dans l'association Commander (entre Produit et Commande) car la quantité dépend de QUELLE commande on parle et de QUEL produit.



## 4. La Confusion Statique vs Dynamique (Action vs Donnée)

- **L'erreur** : Créer des entités pour des actions ou des traitements. Le MCD stocke des **données** (ce qui reste quand on éteint l'ordi), pas des processus.

- **L'exemple illustré :**

-  **Faux** : Créer une entité Validation, Envoi ou Calcul.
-  **Juste** : Utiliser une propriété Statut (ex: "En attente", "Validé", "Envoyé") dans l'entité concernée (ex: Commande), ou gérer cela via des dates (DateValidation, DateEnvoi).

## 5. La Propriété vs L'Entité (Le syndrome de la liste déroulante)

- **L'erreur** : Mettre une information comme simple texte alors qu'elle devrait être une entité à part entière.
- **Le test** : Si l'énoncé suggère une liste fermée de choix, une gestion centralisée, ou des infos supplémentaires sur ce champ, c'est une entité.
- **L'exemple illustré :**
  -  **Faux** : Employé avec propriété NomDépartement (Texte). Risque d'erreur de frappe ("Compta", "Comptabilité") et impossible de gérer le chef du département.
  -  **Juste** : Entité Employé liée à une Entité Département.