Thème 1 : E-A, requête SQL, simulation MySQL

**E-commerce**

**Etat des lieux :**

Chaque entité dans un composant différent est une classe, chaque objet de la classe est une nouvelle instance dans la classe principale.

Cas de E-commerce : formulaire rempli par un client

E-commerce, on affecte un identifiant a une nouvelle instance de chaque table dans la BD, pour effectuer une commande, un client n’est pas forcement enregistrer comme membre

Le formulaire pour remplir les attributs des entités, un client peut ajouter plusieurs articles ce qui fait que la répartition des tables

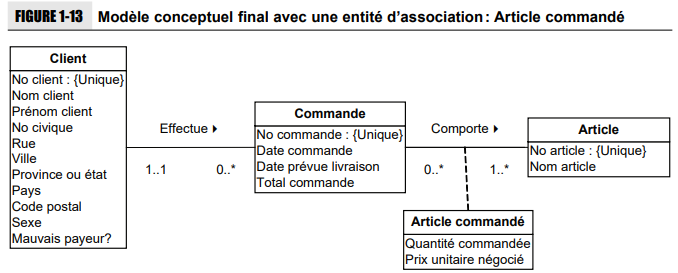
Cas interne : formulaire rempli par un administrateur

Bd : pourquoi pas une seule table, un client x dans la table Client de numéro x1 qui effectue une commande de x11 articles (exemple : le client numéro 2 effectue une commande de 10 articles), on a une clé unique numéro 2, on ne peut pas mettre plus d’un article dans une colonne ce qui revient aux dépendances fonctionnelles, la première forme normale.

**OCL :** (Object Constraint Language) est un langage qui permet d’exprimer des contraintes dans certains types de diagrammes UML, et en particulier dans les diagrammes de classes.

**Interface**: une classe spéciale dont toutes les méthodes sont abstraites

1. **E-A :** analyse



• certains fragments ont été reformulés (le numéro du client est unique et numéro article) ;

• certains fragments n’ont pas été utilisés (nombre article) ;

• les redondances (produit et article apparaissent comme des synonymes) ;

NB : pour un achat en ligne il est inutile de différencier un produit et un article mais dans le cas de stock cela est nécessaire.

• des formulations incorrectes (les exemples ne sont pas des objets)

La colonne Totale commande est une valeur calculée, elle n’est pas nécessaire

Les attributs ne changent pas, se sont des constantes, les éléments variables sont des objets

Client : une variable (on a plusieurs clients)

Commande : une variable (on a plusieurs commandes)

Article : une variable (on a plusieurs commandes)

**Contraint de multiplicité :**

Une occurrence : le client A(Napon,Emile,…) effectue la commande C( c11, 01/03/12, 01/04/12, 1) qui comporte 10 Articles AR( AR11, infinix)

Le client A(Napon,Emile,…) effectue la commande C( c12, 01/03/12, 01/05/12, 1) qui comporte 3 Articles AR( AR11, iphone)

Un seul client peut effectuer aucune ou plusieurs commandes

Si la commandes existe elle est liée à un seul client

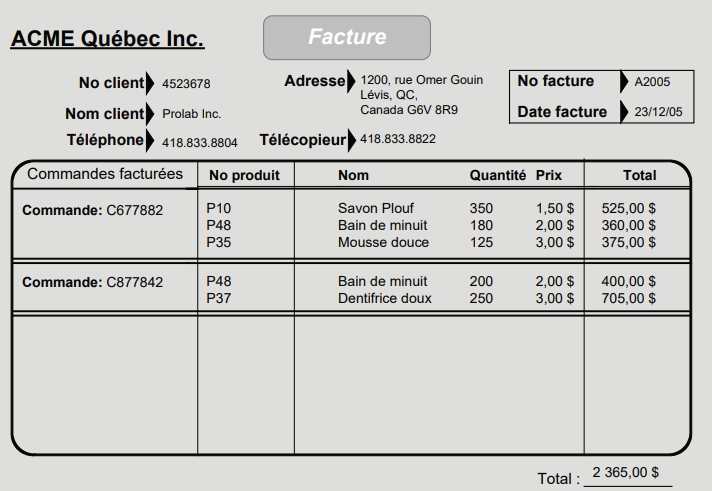
Une commande comporte un ou plusieurs articles

Un article dans le stock est commande ou non commande, on peut avoir la meme article dans une commande ( ex : 10 infinix dans la C( c11, 01/03/12, 01/04/12, 1)

1. **Diagramme de classe : méthode implicite et explicite**

**Exemple de facture :**

**Exemple :**

****

………..

Article\_commande

CalculerPrixArticle()

ajoutArticle ()

SommeTotal()

Article

Client

………..

………..

rechercherArticle()

S’enregistrer()

Commande

………..

commander()

**OCL :**

{OrdonnC} Les commandes d’un client sont ordonnées

intervallePrix exprime une contrainte sur le prix des articles

{addOnly} ajout d’un article a la liste dans l’administration

1. **Schéma**

Create table **Client (**

no\_client varchar(20) not null ,

nom varchar(20),

prenom varchar(20),

numero\_civique varchar(10),

rue varchar(30),

ville varchar(20),

province varchar(20),

pays varchar(20),

code\_postal varchar(20),

sexe varchar(20),

payeur boolean,

**primary key**(no\_client)

**) ;**

Create table **Commande(**

no\_commande varchar(20),

fk varchar(20),

Date\_commande Date,

Date\_livraison Date,

total\_commande int(5),

**primary key**(no\_commande),

**foreign key (**fk) **references** Client(no\_client)

**);**

Create table **Article (**

no\_article varchar(20) non null,

nom\_article varchar(20),

Primary key(no\_article)

**) ;**

Create table **Article\_commande(**

no\_comm varchar(20) non null,

no\_art varchar(20) non null,

Quantite\_commande int;

Prix\_unitaire float

**Primary key**(quantite\_commande,no\_comm,no\_art)

**Foreign key**(no\_comm) **references** Commande(no\_commande),

**Foreign key**(no\_art) **references** Article(no\_article),

**);**