

Programmation orienté objet avancée  
*Dossier d’énoncé du travail de Java*

*Informatique de gestion*

*UE 230 : Projet informatique intégré*

Vous devez remplir chacune des zones bleues encadrées dans le document.

# Identification des étudiants du groupe

## Etudiant 1

Prénom et nom :

|  |
| --- |
| Loucas Pignataro |

Bloc (1, 2 ou 3) :

|  |
| --- |
| **2** |

Groupe (A, B, C …) :

|  |
| --- |
| **B** |

## Etudiant 2

Prénom et nom :

|  |
| --- |
| Alexandre Lambert |

Bloc (1, 2 ou 3) :

|  |
| --- |
| **2** |

Groupe (A, B, C …) :

|  |
| --- |
| **A** |

# Domaine d’application

## Sujet

|  |
| --- |
| Gestion d’une brasserie |

## Description du domaine d’application

|  |
| --- |
| Notre domaine d’application est la gestion d’une brasserie, les clients peuvent acheter de la marchandise sur place dans le magasin ou se la faire livrer. Ils interagissent avec les différents vendeurs pour se faire conseiller ou passer à l’encaissement de leurs articles. Dans le cas de l’encaissement, une facture est alors générée et le processus de vente est terminé. Dans le cas d’une livraison, le paiement peut se faire par virement ultérieur. L’utilisateur de l’application pourra enregistrer ses commandes ainsi que les différents vendeurs, produits, clients, etc dans la base de données. Il pourra effectuer différentes recherches ou encore calculer le montant total d’une commande. |

# Base de données

## Schéma entités-associations

|  |
| --- |
|  |

## Schéma des tables

|  |
| --- |
|  |

## Documentation des tables

Pour chacune des tables, remplissez les zones ci-dessous.

*N.B. Dans la description des colonnes, placez une croix dans "Oblig" si la colonne est obligatoire et placez une croix dans "ID" si la colonne est identifiante à elle seule (clé primaire ou secondaire (unique)). S’il s’agit d’une colonne clé étrangère, placez une croix dans "FK" et précisez le nom de la table référencée.*

Nom de la table :

|  |
| --- |
| **Customer** |

Clé primaire :

|  |
| --- |
| customerId |

Description des colonnes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de colonne | Type(longueur) | Oblig | ID | FK | Table référencée |
| customerId | string |  | X |  |  |
| firstname | string | X |  |  |  |
| lastname | string | X |  |  |  |
| accountNumber | string |  |  |  |  |
| payementType | string | X |  |  |  |
| deadline | int | X |  |  |  |
| streetBilling | string | X |  |  |  |
| streetNumberBilling | int | X |  |  |  |
| localityId | string |  |  | X | Locality |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Nom de la table :

|  |
| --- |
| **Locality** |

Clé primaire :

|  |
| --- |
| localityId |

Description des colonnes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de colonne | Type(longueur) | Oblig | ID | FK | Table référencée |
| localityId | string |  | X |  |  |
| zipCode | int | X |  |  |  |
| name | string | X |  |  |  |

Nom de la table :

|  |
| --- |
| **Seller** |

Clé primaire :

|  |
| --- |
| sellerId |

Description des colonnes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de colonne | Type(longueur) | Oblig | ID | FK | Table référencée |
| sellerId | string |  | X |  |  |
| firstName | string | X |  |  |  |
| lastName | string | X |  |  |  |
| hireDate | date | X |  |  |  |

Nom de la table :

|  |
| --- |
| **Order** |

Clé primaire :

|  |
| --- |
| orderID |

Description des colonnes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de colonne | Type(longueur) | Oblig | ID | FK | Table référencée |
| orderId | string |  | X |  |  |
| seller | string | X |  | X | Seller |
| customer | string | X |  | X | Customer |
| releaseDate | date | X |  |  |  |
| sendDate | date |  |  |  |  |
| withdrawDate | date |  |  |  |  |
| deadlinePayement | date | X |  |  |  |
| receiptDate | date |  |  |  |  |
| numberRemeinderPayment | int | X |  |  |  |
| isPaid | bool | X |  |  |  |
| isDelivered | bool | X |  |  |  |
| deleveryStreet | string |  |  |  |  |
| deleveryStreetNumber | string |  |  |  |  |
| localityId | string | X |  | X | Locality |

Nom de la table :

|  |
| --- |
| **DetailLine** |

Clé primaire :

|  |
| --- |
| Order + Product |

Description des colonnes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de colonne | Type(longueur) | Oblig | ID | FK | Table référencée |
| order | string | X |  | X | Order |
| product | string | X |  | X | Product |
| reduction | double |  |  |  |  |
| sellPrice | double | X |  |  |  |
| quantity | double | X |  |  |  |

Nom de la table :

|  |
| --- |
| **Product** |

Clé primaire :

|  |
| --- |
| code |

Description des colonnes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de colonne | Type(longueur) | Oblig | ID | FK | Table référencée |
| code | string |  | X |  |  |
| name | string | X |  |  |  |
| price | double | X |  |  |  |

# Fonctionnalités

## CRUD : Listing, insertion, suppression, modification

Vous devez prévoir les fonctionnalités complètes d’administration d’une table de la base de données. Les opérations CRUD (**C**reate, **R**ead, **U**pdate et **D**elete) doivent être faites sur une table qui contient :

* Des **colonnes de types différents** (au moins une colonne de type **texte**, au moins une colonne de type **numérique**, au moins une colonne de type **date** et au moins un **booléen**) ;
* Plusieurs **colonnes facultatives**;
* Au moins une **colonne clé étrangère** vers une autre table de la base de données.

*N.B. 4 Fonctionnalités seront donc implémentées sur une même table, à savoir le listing du contenu de la table (dans une JTable), un formulaire d’insertion d’une nouvelle ligne dans la table, la suppression d’une ou plusieurs lignes choisie(s) par l’utilisateur et la possibilité de modifier une ligne de la table (en remplissant d’abord les champs du formulaire d’insertion avec les valeurs de la ligne choisie, en permettant ensuite à l’utilisateur de modifier les champs du formulaire et enfin de sauver la ligne ainsi modifiée).*

Nom de la table pour les opérations CRUD :

|  |
| --- |
| Order |

*Vérification du respect des contraintes :*

Colonne de type date :

|  |
| --- |
| releaseDate |

Colonne de type booléen :

|  |
| --- |
| isPaid |

Colonnes facultatives :

|  |
| --- |
| sendDate, withdrawDate, receiptDate, deleveryStreet, deleveryStreetNumber |

Colonne clé étrangère (précisez vers quelle table) :

|  |
| --- |
| seller => vers la table Seller |

## Recherches

### Recherche 1

Titre de la recherche :

|  |
| --- |
| Commandes passées par les clients d’une localité donnée |

Objectif de la recherche :

|  |
| --- |
| Obtenir la liste des commandes avec le client ainsi que le en fonction d’une certaine localité |

Jointure entre les tables suivantes (**au moins 3**) :

|  |
| --- |
| **Table 1 : Customer**  **Table 2 : Order**  **Table 3 : Seller**  **Table 4 : Locality** |

Description des entrées (introduites par l’utilisateur) :

|  |  |
| --- | --- |
| Critère de recherche | Format (type de composant Swing) |
| Localité | JComboBox proposant la combinaison du code postal et du libellé (pas l’id) |

Sorties (dans une JTable) :

|  |  |
| --- | --- |
| Nom de la colonne | Provenant de la table (nom de la table) |
| lastName | Customer |
| firstName | Customer |
| orderId | Order |
| orderDate | Order |
| lastName | Seller |
| firstName | Seller |

### Recherche 2

Titre de la recherche :

|  |
| --- |
| Recherche des lignes de commandes passées entre 2 dates données |

Objectif de la recherche :

|  |
| --- |
| Obtenir un récapitulatif des commandes passées entre 2 dates |

Jointure entre les tables suivantes (**au moins 3**) :

|  |
| --- |
| **Table 1 : Order**  **Table 2 : DetailLine**  **Table 3 : Product** |

Description des entrées (introduites par l’utilisateur) :

|  |  |
| --- | --- |
| Critère de recherche | Format (type de composant Swing) |
| Intervalle de dates => 2 dates | 2 JSpinner |

Sorties (dans une JTable) :

|  |  |
| --- | --- |
| Nom de la colonne | Provenant de la table (nom de la table) |
| orderDate | Order |
| orderId | Order |
| quantity | DetailLine |
| sellPrice | DetailLine |
| name | Product |
|  |  |

### Recherche 3

Titre de la recherche :

|  |
| --- |
| Informations sur les clients qui ont commandé un produit donné |

Objectif de la recherche :

|  |
| --- |
| Obtenir des informations sur les clients(adresse, modalités de payement, etc) qui ont commandé un produit |

Jointure entre les tables suivantes (**au moins 3**) :

|  |
| --- |
| **Table 1 : Produit**  **Table 2 : LigneDetail**  **Table 3 : Commande**  Table 4 : Client  Table 5 : Localité |

Description des entrées (introduites par l’utilisateur) :

|  |  |
| --- | --- |
| Critère de recherche | Format (type de composant Swing) |
| produit | JComboBox reprenant la liste des libellés (et pas des id) des produits |

Sorties (dans une JTable) :

|  |  |
| --- | --- |
| Nom de la colonne | Provenant de la table (nom de la table) |
| deadlinePayment | Order |
| firstname | Customer |
| lastname | Customer |
| billingStreet | Customer |
| billingStreetNumber  zipCode | Customer  Locality |
| name | Locality |

## Tâche métier

Vous devez implémenter au moins une fonctionnalité relative à votre projet autre que les précédentes (exemples : planification, règles métier, statistiques…). Cette tâche sera implémentée dans la couche métier (calculs, règles métier…). L’exécution de cette tâche métier peut nécessiter éventuellement l’affichage de plusieurs écrans successifs, l’obtention d’informations auprès de l’utilisateur ou des accès en lecture et/ou écriture à la base de données.

Description de la tâche métier :

|  |
| --- |
| Calcul du prix total pour une commande |

Entrées (saisies auprès de l’utilisateur) :

|  |
| --- |
| JCombobox proposant la liste des noms et prénoms des clients existants dans la BD   * Choix d’un client * Affichage dans une JTable de toutes ses sommandes * Choix d’une commande * Affichage du prix total de la commande |

Sorties (affichage) :

|  |
| --- |
| Affichage d’une fenêtre indiquant « Le montant total de votre commande xx est de xx € » |

## Thread

Description du processus parallèle (autre que l’affichage de l’heure) :

|  |
| --- |
| Notre thread va gérer l’affichage d’un dessin d’une bière qui se remplit puis se vide en boucle |