

[Projet n°6]

Solution technique d'un système de gestion de pizzeria

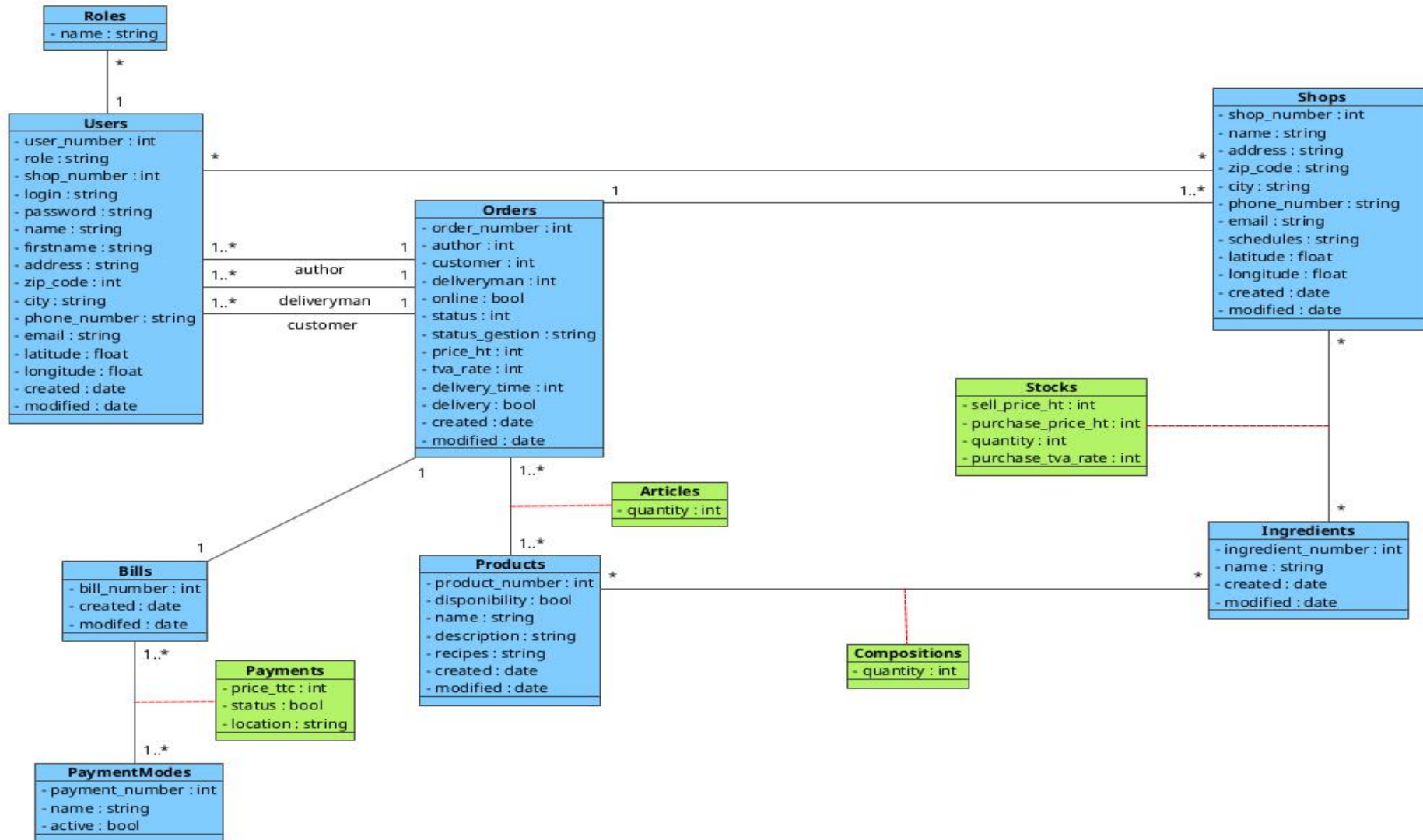
Yohan SOLON



SOMMAIRE

Domaine fonctionnelle.....	3
Composant et interactions.....	4
Diagramme de déploiement.....	13

Domaine fonctionnelle



Composant et interactions

- Association Roles / Users :



Définitions des classes :

- Roles : Représente le rôle que un utilisateur peut avoir
- Users : Représente un utilisateur de l'application

Interprétation de la relation :

- Un rôle peut être assigné à un ou plusieurs utilisateur(s)
- Un utilisateur a un rôle

Table de correspondance :

Users	
Id [PK]	name
1	Jean
2	Elise
3	Pierre

Roles
name [PK]
Client
Employé

Users		
Id [PK]	name	role_name [FK]
1	Jean	Client
2	Elise	Employé
3	Pierre	Employé

Tuples des tables "users" et "roles"

Pour définir quel rôle possède chaque utilisateur nous ajoutons un nouvel attribut "role_name" en tant que clé étrangère FK dans la table Users et on y ajoute la valeur de la clé primaire du rôle correspondant

Ajout de la clé étrangère "role_name" dans la table Users

- Association Users / Shops, relation plusieurs-à-plusieurs



Définitions des classes :

- Shops : Représente un magasin
- Users : Représente un utilisateur “employé” de l’application

Interprétation de la relation :

- Un magasin peut avoir plusieurs employé-e(s)
- Un / des employé-e(s) peut être assigné-e(s) dans plusieurs magasins

Table de correspondance :

Users		Shops		Employees	
Id [PK]	name	Id [PK]	name	user_id [PFK]	shop_id [PFK]
1	Jean	1	Mag_1	1	1
2	Elise	2	Mag_2	2	2
3	Pierre			3	2
				3	1

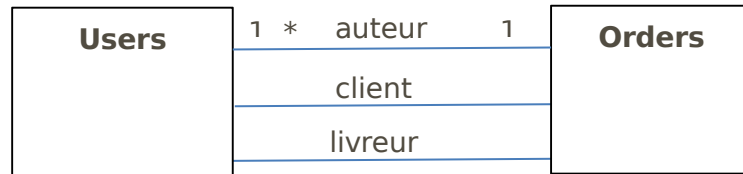
Tuples des tables “users” et “shops”

Pour répondre à la multiplicité des informations, nous mettrons en place une table qui matérialisera les différentes liaisons, afin de définir les employés d’un magasin.

La clé primaire de de cette table sera composé de clé étrangère point sur les tables users et shops.

Ajout d'une table employees matérialisant la relation « plusieurs à plusieurs » entre les tables users et shops

- Association Users / Orders, relations un-à-plusieurs



Définitions des classes :

- Orders : Représente une commande
- Users : Représente un utilisateur de l'application

Interprétation des relations :

- La première association indique l'auteur de la commande
- La deuxième association le client de la commande
- La troisième association l'employé en charge de donner la livraison

Table de correspondance :

Users		Orders		Orders			
Id [PK]	name	Id [PK]		Id [PK]	author [FK]	customer [FK]	deliveryman [FK]
1	Jean	1		1	2	1	3
2	Elise	2		2	2
3	Pierre						

Ajout des clefs étrangères dans la table Orders

Tuples des tables "users" et "orders"

Pour définir dans quel association est un utilisateur nous ajoutons de nouveaux attributs "author", "customer", "deliveryman" en tant que clé étrangère FK dans la table Orders et on y ajoute la valeur de la clé primaire de l'utilisateur correspondant

- Association Orders / Shops relation un-à-plusieurs



Définitions des classes :

- Orders : Représente une commande
- Shops : Représente un magasin

Interprétation des relations :

- Une commande est assigné à un magasin
- Un magasin peut avoir 0 ou plusieurs commandes

Table de correspondance :

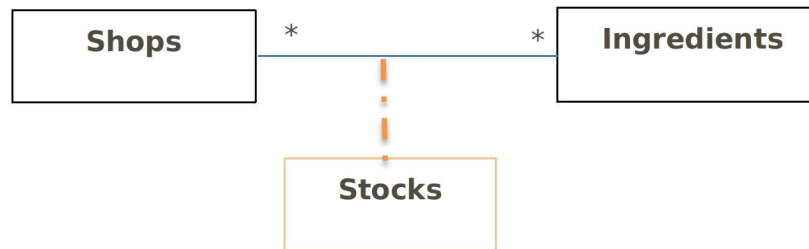
Orders		Shops		Orders	
Id [PK]		Id [PK]	name	Id [PK]	shop_id [FK]
1		1	Mag_1	1	2
2		2	Mag_2	2	2

Tuples des tables “orders” et “shops”

Ajout de la clef étrangère dans la table Orders

Pour définir dans quel magasin est associer une commande nous ajoutons un nouvel attribut “shop_id” en tant que clé étrangère FK dans la table Orders et on y ajoute la valeur de la clé primaire du magasin correspondant,

- Associations Shops / Ingrédients relation plusieurs-à-plusieurs



Définitions des classes :

- Shops : Représente un magasin
- Ingrédients : Représente un ingrédient
- Classe d'association stocks

Interprétation de la relation :

- Un magasins peut contenir un stock d'ingrédients
- Un ingrédient peut être dans plusieurs stocks

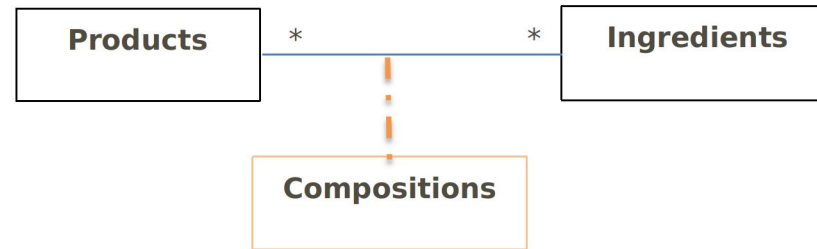
Shops			Ingrédients			Stocks		
Id [PK]	name		Id [PK]	name		shop_Id [PFK]	Ingredient_id [PFK]	quantity
1	Mag_1		1	Ingredient 00		1	1	10
2	Mag_2		2	Ingredient 01		2	1	50

Tuples des tables "shops" et "ingrédients"

Pour définir l'association de nos 2 tables la table stocks est constitué d'une clef primaire composé de deux clefs étrangère des valeurs des clefs primaires des table Shops et Ingrédients et indique ça quantité.

Classe d'association stocks sur une base de d'une table de relation plusieurs-à-plusieurs

- Association Products / Ingredients relation plusieurs-à-plusieurs



Définitions des classes :

- Products : Représente un produit
- Ingredients : Représente un ingrédient

Interprétation des relations :

- Un produit est composés de plusieurs ingrédients
- Un ingrédient peut composés différents produits
- La classe d'association compositions contient les quantité d'ingrédients pour un produit

Products	
Id [PK]	name
1	product_1
2	product_2

Ingrédients	
Id [PK]	name
1	Ingredient 00
2	Ingredient 01

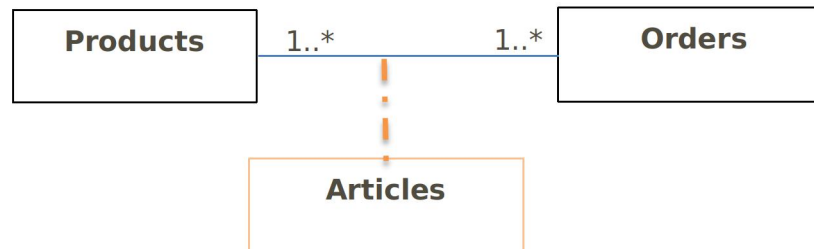
Compositions		
product_Id [PFK]	Ingredient_id [PFK]	quantity
1	1	4
2	2	5

Classe d'association compositions

Tuples des tables “products” et “ingredients”

La classe d'association compositions est constitué d'une clef primaire composé des clefs étrangère des tables products et ingrédients ayant comme valeur leurs clefs primaires et la quantité d'ingrédients par association.

- Association Products / Orders, relation plusieurs-à-plusieurs



Définitions des classes :

- Products : Représente un produit
- Orders : Représente une commande

Interprétation des relations :

- Un commande est composés de 1 à plusieurs articles
- Un produits peut composés 1 à commandes
- La classe d'association articles contient les quantité de produit que compose une commande

Products	
Id [PK]	name
1	product_1
2	product_2

Orders
Id [PK]
1
2

Articles		
product_id [PFK]	order_id [PFK]	quantity
1	1	4
2	1	5

Classe d'association articles

Contient la clef primaire composé des clefs étrangère product_id et order_id, ayant la valeurs des clef primaire de la table products et orders.

Tuples des tables "products" et "orders"

La classe d'association articles est chargé de faire la jonction entre les tables products et orders, constituant ainsi la liste de la quantité d'ingrédients nécessaire dans la constitution d'un produit.

- Associations Orders / Bills, relation un-à-un



Définitions des classes :

- Orders : Représente une commande
- Bills : Représente une facture

Interprétation des relations :

- Une commande à une facture
- Une facture correspond à une commande

Orders
Id [PK]
1
2

Bills
number [PK]
1
2

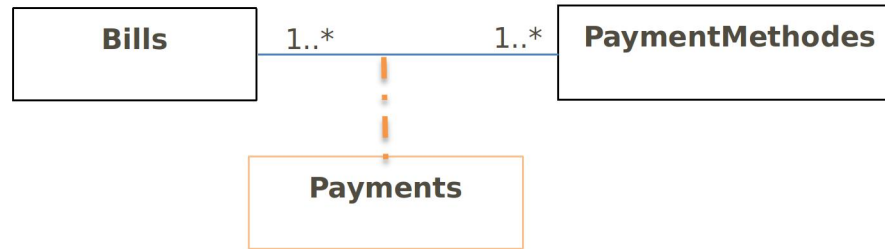
Bills	
number [PK]	order_id [FK]
1	1
2	2

Tuples des tables “orders” et “Bills”

Ajout de la clef étrangère dans la table Bills

Pour savoir à quel commande correspond une facture nous allons ajouter une clef étrangère dans la table Bills qui prendra comme valeur la clef primaire de la table orders.

- Association Bills / PaymentMethodes, relation plusieurs-à-plusieurs




Définitions des classes :

- Bills : Représente une facture
- PaymentMethodes : Représente les moyens de paiements

Interprétation des relations :

- Une factures a au minimum 1 à plusieurs mode de paiement
- Une méthodes de paiement peut apparaître dans 1 à plusieurs factures
- La classe d'association payments contient les différents paiement effectué pour une facture

Bills			PaymentMethodes			Payments	
number [PK]	order_id [FK]		id [PK]			bill_id [PFK]	payment_methode_id [PFK]
1	1		1			1	1
2	2		2			2	1

Tuples des tables “bills” et “PaymentMethodes”

La classe d'association payments est chargé de faire la jonction entre les tables bills et paymentMethodes, constituant ainsi la liste des moyens de paiement utilisés par le client.

Classe d'association Payments

Contient la clef primaire composé des clefs étrangère bill_id et payment_methode_id, ayant la valeurs des clef primaire de la table bills et paymentMethodes.

Diagramme de déploiement

