III. Mise en place de la solution proposée

1. Modélisation : la méthode MERISE

Merise a principalement été utilisée en France dans les administrations publiques ou privés et par certaines entreprises de services du numérique. La méthode Merise est une méthode d'analyse, de conception et de réalisation de systèmes d'informations.

La méthode Merise se caractérise par ces différents axes :

- Identifier les acteurs du projet
- Réaliser le schéma directeur (objectifs/stratégie de l'entreprise, cadre du projet)
- Faire l'étude préalable (description des besoins et des attentes des utilisateurs, ébauche des divers modèles)
- Faire l'étude détaillée
- Réalisation de l'étude technique (moyens techniques nécessaires)
- Mise en production
- Gestion de la maintenance (informations sur les acteurs, les documentations et les formations)

La méthode MERISE se déroule en trois grandes étapes :

- Le modèle conceptuel de données (MCD) : Quelles relations lient les données entre elles ?
- Le modèle logique de données (MLD) : Quelles sont les liens qui découlent de ces relations ?
- Le modèle physique de données (MPD) : C'est l'implémentation en langage SQL (Structured Query Language) de la base de données pour un système de gestion choisit.

Nous avons décidé de choisir la méthode Merise pour sa fiabilité, sa rapidité et son efficacité. De plus, c'est sur cette méthode que les membres de l'équipe ont été formés.

Pour formaliser les différentes étapes de la méthode Merise nous avons choisi le logiciel JMERISE, distribué gratuitement au téléchargement. Ce logiciel permet de formaliser un MCD puis de le traduire automatiquement en MLD puis en script SQL (MPD). La conversion des différents modèles est rapide et efficace. En outre, JMERISE gère la version 2 de Merise 2 ; nous permettant notamment d'implémenter la notion d'héritage dans notre MCD.

a. Modèle conceptuel de données (MCD)

Pour pouvoir construire le modèle conceptuel de données[1] on procède en deux étapes :

- La création d'un dictionnaire de données (DD)
- La création d'une matrice des dépendances fonctionnelles (Matrice des DF)

[1] Tous les documents concernant la modélisation de la base de données sont disponibles dans le dossier modélisation sur le GitHub du projet.

Le dictionnaire de données ou recueil d'informations consiste à collecter les données. Nous avons donc répertorié l'ensemble des données que nous jugions nécessaire en analysant le sujet du projet.

Nous aboutissons au dictionnaire de données suivant :

Nom	Description	Type	Contraintes
IDClient	Identifiant du client	Numérique	Obligatoire
NomClient		Alphabétique	Obligatoire
PrenomClient		Alphabétique	Obligatoire
AdresseClient	Lieu d'habitation du client	varchar	Obligatoire
IDFournisseur	Identifiant du fournisseur	Numérique	Obligatoire
NomFournisseur		Alphanumérique	Obligatoire
NumeroFournisseur	Numéro de téléphone du fournisseur	Numérique	Obligatoire
IDIventeur	Identifiant de l'inventeur	Numérique	Obligatoire
NomInventeur		Alphabétique	Obligatoire
NumeroPermis	Permis de l'inventeur	Numérique	Obligatoire
IDDiluant	Identifiant du diluant	Numérique	Obligatoire
NomDiluant		Alphabétique	Obligatoire
IDRecipient		Numérique	Obligatoire
NomRecipient		Alphabétique	Obligatoire
StockRecipients	Quantité disponible de récipients en stock	Numérique	Obligatoire, >= 0
PrixRecipient		Décimale	Obligatoire
IDOnguent	Identifiant de l'onguent	Numérique	Obligatoire
NomOnguent		Alphabétique	Obligatoire
PrixOnguent		Décimale	Obligatoire
IDIngredient		Numérique	Obligatoire
NomIngredient		Alphabétique	Obligatoire
QuantiteIngredient	Quantité d'ingrédient dans la recette	Numérique	
FraicheurIngredient	Degré de fraicheur (en jour)	Numérique	Obligatoire, >= 0
PrixIngredient		Décimale	Obligatoire
StockIngredients	Quantité disponible d'ingrédients en stock	Numérique	Obligatoire, >= 0
IDPotion	Identifiant d'une potion	Numérique	Obligatoire
NomPotion		Alphabétique	Obligatoire
TemperaturePotion	Température de préparation	Numérique	
PrixPotion		Décimale	Obligatoire
FraicheurMax		Numérique	
FraicheurMin		Numérique	
IDCommande	Numéro de commande	Numérique	Obligatoire
FraicheurCommande	Fraicheur désirée par le client	Numérique	
QuantiteCommande	Quantité de la commande	Numérique	Obligatoire
DateCommande		date	Obligatoire
StatutCommande		Alphabétique	Obligatoire
PrixTotal		Décimale	Obligatoire
IDProposition	Identifiant de la proposition d'une nouvelle recette	Numérique	Obligatoire
NomPotionPropose		Alphabétique	Obligatoire
FraicheurMaxProposee		Numérique	
FraicheurMinProposee		Numérique	
QuantiteProposee		Numérique	Obligatoire
TemperatureProposee	Température de préparation de la recette proposée	Numérique	Obligatoire
ValidationRecetteProposee	Permet de valider ou non une recette proposée	Booléen	

Figure 4 - Dictionnaire de données (DD)

La deuxième phase de conception du MCD est l'analyse des dépendances. Les dépendances sont les liens qui lient les différentes données. Le document résultant de cette étape est un diagramme ou une matrice des dépendances.

A partir du dictionnaire précédant on construit la matrice des dépendances fonctionnelles suivantes :

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
1	IDClient																																												
2	NomClient	X																																											
3	PrenomClient	X																																										
4	AdresseClient	X																																										
5	IDFournisseur																																											
6	NomFournisseur					X																																						
7	NumeroFournisseur					X																																						
8	IDIventeur																																											
9	NomInventeur								X																																			
10	NumeroPermis								X																																			
11	IDDiluant																																											
12	NomDiluant											X																																
13	IDRecipient																																											
14	NomRecipient													X																														
15	StockRecipients													X																														
16	PrixRecipient													X																														
17	IDOnguent																																											
18	NomOnguent																	X																										
19	PrixOnguent																	X																										
20	IDIngredient																																											
21	NomIngredient																				X																							
22	QuantiteIngredient																				X																							
23	FraicheurIngredient																				X																							
24	PrixIngredient																				X																							
25	StockIngredients																				X																							
26	IDPotion																																											
27	NomPotion																										X																	
28	TemperaturePotion																										X																	
29	PrixPotion																										X																	
30	FraicheurMax																				C						C																	
31	FraicheurMin																				C						C																	
32	IDCommande																																											
33	FraicheurCommande																																X											
34	QuantiteCommande																																X											
35	DateCommande																																X											
36	StatutCommande																																X											
37	PrixTotal																																X											
38	IDProposition																																											
39	NomPotionPropose																																						X					
40	FraicheurMaxProposee																																						X					
41	FraicheurMinProposee																																						X					
42	QuantiteProposee																																						X					
43	TemperatureProposee																																						X					
44	ValidationRecetteProposee																																						X					

Figure 5 - Matrice des dépendances fonctionnelles

A l'aide de ces deux documents (DD et matrice des DF) on établit le modèle conceptuel de données sur le logiciel JMERISE.

On obtient le MCD suivant :

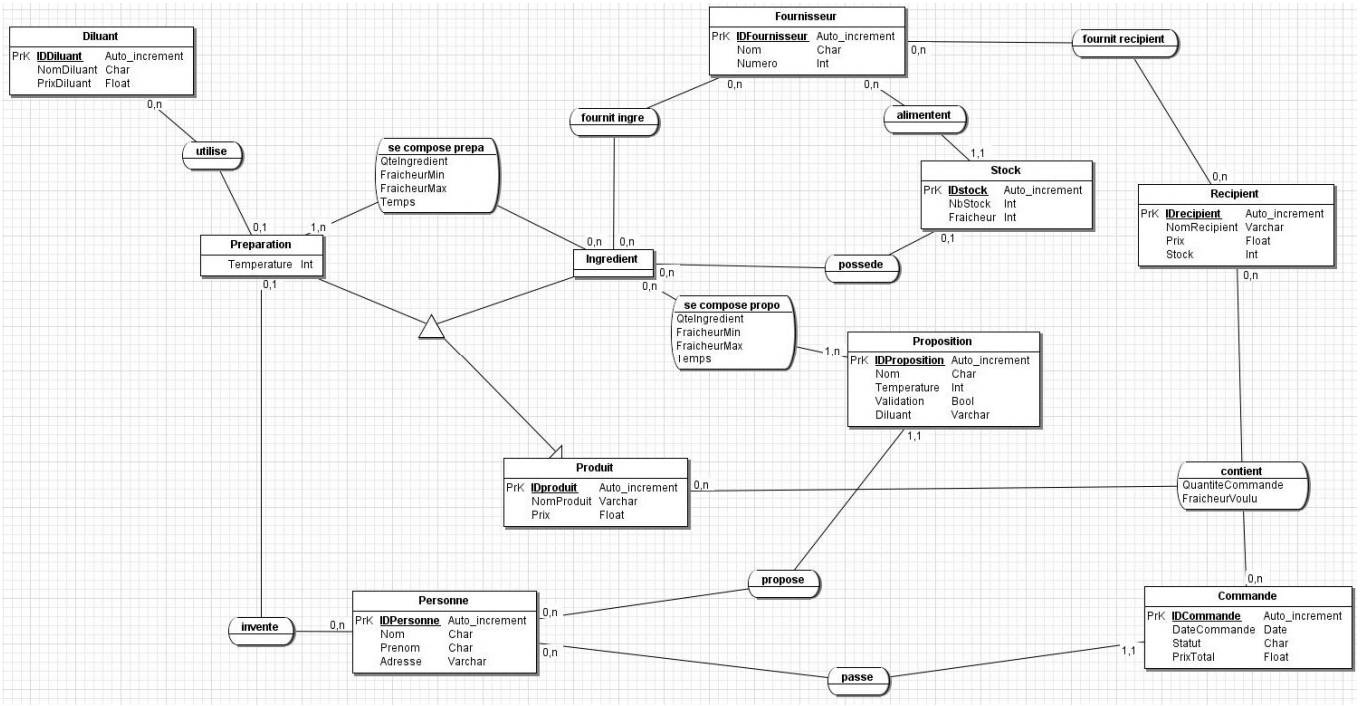

Figure 6 - Modèle Conceptuel de données

La figure 6 présente le modèle conceptuel de données de la future base de données du site GiselleMagicArts.com.

Tout d'abord, l'entité produit rassemble tous ce que la sorcière Giselle vend sur son site ; c'est-à-dire potions, onguents et des ingrédients seuls. Chacun de ces produits sont identifiés de manière unique par un numéro (un peu comme une référence). Cette entité produit est l'entité mère d'une relation d'héritage dans laquelle les deux entités files sont :

- Ingrédient : entité contenant les références des produits qui sont des ingrédients
- Préparation : entité contenant les références des produits qui sont des préparations (onguents et potions)

Ainsi, l'entité préparation contient aussi bien les onguents que les potions. Pour identifier les onguents des potions et vice versa il suffira de distinguer les préparations qui utilisent un diluant (potions) et celles qui n'en utilisent pas (onguents).

Une préparation utilise un ou aucun diluant (cas des onguents).

L'entité personne rassemble les clients mais aussi les inventeurs. En effet, un inventeur peut aussi être un client et inversement. C'est pour cette raison que la cardinalité du côté de l'entité personne de l'association invente est 0,n. Une personne peut inventer plusieurs ou aucune préparation. L'autre cardinalité de cette association montre que pour une préparation on ne connaît pas toujours l'inventeur.

Une personne passe des commandes qui contiennent des produits ainsi que des récipients. En effet, pour chaque produit commandé il faudra indiqué un récipient pour le transport et sa conservation.

Les personnes peuvent également proposer des idées de préparation qui sont composés de produits (Ingrédients, potions et onguents). De ce fait, il sera possible de créer une potion dans laquelle un des ingrédients est une autre potion ou un onguent.

L'entité fournisseur rassemble l'ensemble des fournisseurs qui Giselle sollicitera. Ainsi, les fournisseurs fournissent des ingrédients et des récipients. Tout en sachant qu'un même type de récipient ou un même ingrédient peut être fournit par plusieurs fournisseurs différents. Ils alimentent également un stock d'ingrédient.

Un identifiant de stock représente un ingrédient avec une fraîcheur précise. Par conséquent, un même ingrédient peut se retrouver plusieurs fois dans le stock mais avec des fraicheurs différentes.

b. Modèle logique de données (MLD)

Maintenant nous devons transformer le MCD en modèle logique de données (MLD). La quatrième étape de l'analyse des données est donc la traduction du MCD en MLD. Cette traduction s'effectue en suivant des règles précises.

Le MLD est toujours indépendant du SGBD utilisé. Il est composé uniquement de relations. Une relation est composée d'attributs et possède systématiquement une et une seule clé primaire qui permet d'identifier sans ambiguïté chaque occurrence de cette relation.

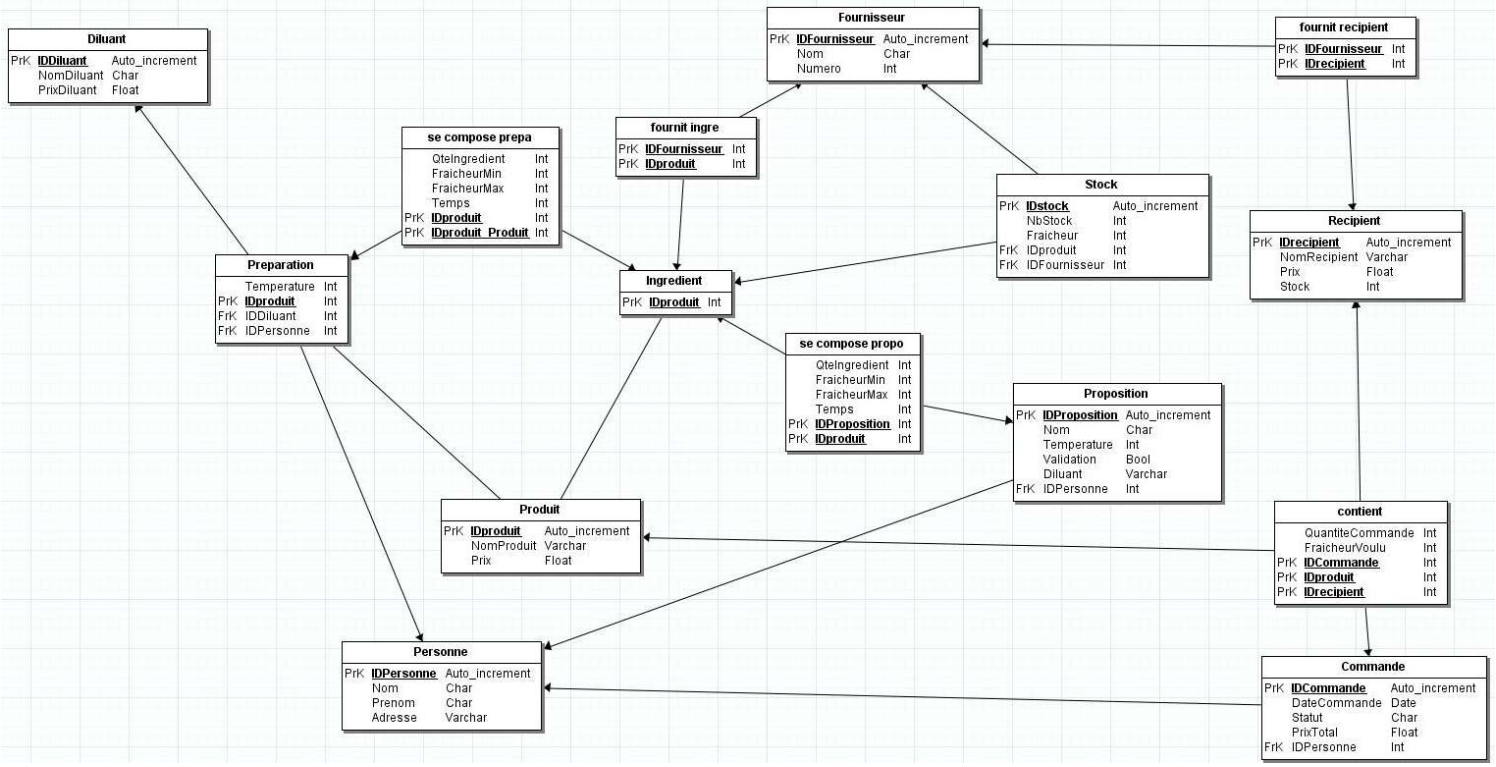

Figure 7 - Modèle logique de données

c. Choix du SGBDR

Nous avons choisi le Système de Gestion de Base de Données Relationnel (SGBDR) : MySQL. Comme la majeure partie des SGBDR, il utilise le langage SQL.

Figure 8 - Logo de MySQL

Dans un premier temps, nous avons été formés à l'école sur MySQL. Cela nous a permis d'avoir plus d'aisance dans la création de la base de données sur ce SGBDR.

De plus MySQL est gratuit et disponible sur plusieurs systèmes d'exploitation. Il est multi-utilisateur, cela nous permet donc de travailler tous ensemble sur la même base de données. Pour cela nous avons donc hébergé notre serveur MySQL sur un Raspberry Pi 3[2].

MySQL dispose également de fonctionnalité de sécurité. Ce SGBDR contient un système de privilèges qui nous permet de gérer les différents accès sur ce serveur et il dispose d'un système de sauvegardes.

d. Modèle physique de données (MPD)

Le modèle physique de données est l'implémentation du MLD par un logiciel. On va devoir traduire le MLD pour obtenir un MPD. On va notamment préciser le type et la taille (octets, bits) des données pour savoir comment les stocker.

La traduction d'un MLD relationnel en MPD passe par la création d'une base de données hébergée par un SGBDR (ici MySQL).

Le MPD est donc le script de création de l'ensemble des tables et des relations entre elles en langage SQL.

Vous trouverez le fichier script_creation_bdd dans l'annexe 2 ainsi que sur le dépôt GitHub du projet.

[2] Pour plus d'informations sur la création du serveur Mysql sur un Raspberry Pi 3 se repporter à l'annexe 1

ANNEXE 1 : Créer un serveur MySQL sur un Raspberry Pi

Nous voulions que l'ensemble des membres de l'équipe du projet puisse se connecter et travailler sur la base de données de manière simultanée. Nous avons donc décidé d'installer un serveur MySQL (version 5.7) sur un raspberry pi 3. Ce dernier est connecté au domicile d'un membre de l'équipe et accessible depuis l'extérieur.

A l'aide de l'adresse IP publique du routeur domestique et à la création de comptes utilisateurs sur la base de données MySQL, les membres de l'équipe peuvent travailler sur la base de données en même temps.

Détaillons les étapes de déploiement de ce serveur sur un raspberry pi 3.

Tout d'abord nous conseillons de mettre à jour l'ensemble du système d'exploitation de l'ordinateur.

Ensuite on utilise la commande suivante pour récupérer le paquet d'installation du serveur MySQL version 5.7 (dernière version stable actuellement) :

```
pi@raspberrypi:~ $ sudo apt-get install mysql-server-5.7
```

Après quelques minutes, le serveur MySQL est en place, pour le démarrer on utilise la commande :

```
pi@raspberrypi:~ $ sudo service mysql start
```

On peut maintenant se connecter en local au serveur mysql en utilisant la commande :

```
mysql -h localhost -u user -ppassword
```

Une fois connecté, on crée un compte utilisateur pour chaque membre de l'équipe du projet à l'aide de la clause GRANT.

```
GRANT ALL PRIVILEGES ON *.* TO 'florian'@'%' IDENTIFIED BY 'password';
```

Ici on attribue tous les privilèges à l'utilisateur florian qui peut se connecter depuis n'importe quelle adresse IP avec le mot de passe password.

Ensuite sur le routeur, on créer une règle de redirection du port 3306 (port MySQL par défaut) arrivant de l'extérieur vers l'adresse IP local du Raspberry comme le montre la capture d'écran suivante.

server-mysql	TCP	Port	3306	192.168.1.48	3306

Grâce à cette règle de redirection peut maintenant se connecter à distance au serveur MySQL et donc à la base de données du projet. Pour l'administrer on pourra utiliser un terminal ou alors phpmyadmin.

L'installation de phpmyadmin requiert la mise en place d'un serveur Apache et la redirection du port 80.

ANNEXE 2 : Script de création de la base de données

```
#------------------------------------------------------------
#        Script MySQL.
#------------------------------------------------------------

#------------------------------------------------------------
# Table: Personne
#------------------------------------------------------------

CREATE TABLE Personne(
        IDPersonne int (11) Auto_increment  NOT NULL ,
        Nom        Char (25) NOT NULL ,
        Prenom     Char (25) NOT NULL ,
        Adresse    Varchar (25) NOT NULL ,
        PRIMARY KEY (IDPersonne )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Ingredient
#------------------------------------------------------------

CREATE TABLE Ingredient(
        IDproduit Int NOT NULL ,
        PRIMARY KEY (IDproduit )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Diluant
#------------------------------------------------------------

CREATE TABLE Diluant(
        IDDiluant   int (11) Auto_increment  NOT NULL ,
        NomDiluant  Char (25) NOT NULL ,
        PrixDiluant Float NOT NULL ,
        PRIMARY KEY (IDDiluant )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Recipient
#------------------------------------------------------------

CREATE TABLE Recipient(
        IDrecipient  int (11) Auto_increment  NOT NULL ,
        NomRecipient Varchar (255) ,
        Prix         Float NOT NULL ,
        Stock        Int ,
        PRIMARY KEY (IDrecipient )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Commande
#------------------------------------------------------------

CREATE TABLE Commande(
        IDCommande   int (11) Auto_increment  NOT NULL ,
        DateCommande Date NOT NULL ,
        Statut       Char (25) ,
        PrixTotal    Float NOT NULL ,
        IDPersonne   Int ,
        PRIMARY KEY (IDCommande )
)ENGINE=InnoDB;
```

```
#------------------------------------------------------------
# Table: Fournisseur
#------------------------------------------------------------

CREATE TABLE Fournisseur(
        IDFournisseur int (11) Auto_increment  NOT NULL ,
        Nom           Char (25) NOT NULL ,
        Numero        Int NOT NULL ,
        PRIMARY KEY (IDFournisseur )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Proposition
#------------------------------------------------------------

CREATE TABLE Proposition(
        IDProposition int (11) Auto_increment  NOT NULL ,
        Nom           Char (25) NOT NULL ,
        Temperature   Int ,
        Validation    Bool ,
        Diluant       Varchar (25) ,
        IDPersonne    Int ,
        PRIMARY KEY (IDProposition )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Stock
#------------------------------------------------------------

CREATE TABLE Stock(
        IDstock       int (11) Auto_increment  NOT NULL ,
        NbStock       Int NOT NULL ,
        Fraicheur     Int ,
        IDproduit     Int ,
        IDFournisseur Int ,
        PRIMARY KEY (IDstock )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Produit
#------------------------------------------------------------

CREATE TABLE Produit(
        IDproduit  int (11) Auto_increment  NOT NULL ,
        NomProduit Varchar (255) ,
        Prix       Float ,
        PRIMARY KEY (IDproduit )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: Preparation
#------------------------------------------------------------

CREATE TABLE Preparation(
        Temperature Int ,
        IDproduit   Int NOT NULL ,
        IDDiluant   Int NOT NULL ,
        IDPersonne  Int ,
        PRIMARY KEY (IDproduit )
)ENGINE=InnoDB;
```

```
#------------------------------------------------------------
# Table: contient
#------------------------------------------------------------

CREATE TABLE contient(
        QuantiteCommande Int ,
        FraicheurVoulu   Int ,
        IDCommande       Int NOT NULL ,
        IDproduit        Int NOT NULL ,
        IDrecipient      Int NOT NULL ,
        PRIMARY KEY (IDCommande ,IDproduit ,IDrecipient )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: fournit ingre
#------------------------------------------------------------

CREATE TABLE fournit_ingre(
        IDFournisseur Int NOT NULL ,
        IDproduit     Int NOT NULL ,
        PRIMARY KEY (IDFournisseur ,IDproduit )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: se compose prepa
#------------------------------------------------------------

CREATE TABLE se_compose_prepa(
        QteIngredient     Int ,
        FraicheurMin      Int ,
        FraicheurMax      Int ,
        Temps             Int ,
        IDproduit         Int NOT NULL ,
        IDproduitFinal Int NOT NULL ,
        PRIMARY KEY (IDproduit ,IDproduitFinal )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: se compose propo
#------------------------------------------------------------

CREATE TABLE se_compose_propo(
        QteIngredient Int ,
        FraicheurMin  Int ,
        FraicheurMax  Int ,
        Temps         Int ,
        IDProposition Int NOT NULL ,
        IDproduit     Int NOT NULL ,
        PRIMARY KEY (IDProposition ,IDproduit )
)ENGINE=InnoDB;

#------------------------------------------------------------
# Table: fournit recipient
#------------------------------------------------------------

CREATE TABLE fournit_recipient(
        IDFournisseur Int NOT NULL ,
        IDrecipient   Int NOT NULL ,
        PRIMARY KEY (IDFournisseur ,IDrecipient )
)ENGINE=InnoDB;
```

```
ALTER TABLE Ingredient ADD CONSTRAINT FK_Ingredient_IDproduit FOREIGN KEY (IDproduit)
REFERENCES Produit(IDproduit);

ALTER TABLE Commande ADD CONSTRAINT FK_Commande_IDPersonne FOREIGN KEY (IDPersonne)
REFERENCES Personne(IDPersonne);

ALTER TABLE Proposition ADD CONSTRAINT FK_Proposition_IDPersonne FOREIGN KEY (IDPersonne)
REFERENCES Personne(IDPersonne);

ALTER TABLE Stock ADD CONSTRAINT FK_Stock_IDproduit FOREIGN KEY (IDproduit) REFERENCES
Produit(IDproduit);

ALTER TABLE Stock ADD CONSTRAINT FK_Stock_IDFournisseur FOREIGN KEY (IDFournisseur)
REFERENCES Fournisseur(IDFournisseur);

ALTER TABLE Preparation ADD CONSTRAINT FK_Preparation_IDproduit FOREIGN KEY (IDproduit)
REFERENCES Produit(IDproduit);

ALTER TABLE Preparation ADD CONSTRAINT FK_Preparation_IDDiluant FOREIGN KEY (IDDiluant)
REFERENCES Diluant(IDDiluant);

ALTER TABLE Preparation ADD CONSTRAINT FK_Preparation_IDPersonne FOREIGN KEY (IDPersonne)
REFERENCES Personne(IDPersonne);

ALTER TABLE contient ADD CONSTRAINT FK_contient_IDCommande FOREIGN KEY (IDCommande)
REFERENCES Commande(IDCommande);

ALTER TABLE contient ADD CONSTRAINT FK_contient_IDproduit FOREIGN KEY (IDproduit)
REFERENCES Produit(IDproduit);

ALTER TABLE contient ADD CONSTRAINT FK_contient_IDrecipient FOREIGN KEY (IDrecipient)
REFERENCES Recipient(IDrecipient);

ALTER TABLE fournit_ingre ADD CONSTRAINT FK_fournit_ingre_IDFournisseur FOREIGN KEY
(IDFournisseur) REFERENCES Fournisseur(IDFournisseur);

ALTER TABLE fournit_ingre ADD CONSTRAINT FK_fournit_ingre_IDproduit FOREIGN KEY (IDproduit)
REFERENCES Produit(IDproduit);

ALTER TABLE se_compose_prepa ADD CONSTRAINT FK_se_compose_prepa_IDproduit FOREIGN KEY
(IDproduit) REFERENCES Produit(IDproduit);

ALTER TABLE se_compose_prepa ADD CONSTRAINT FK_se_compose_prepa_IDproduitFinal FOREIGN KEY
(IDproduitFinal) REFERENCES Produit(IDproduit);

ALTER TABLE se_compose_propo ADD CONSTRAINT FK_se_compose_propo_IDProposition FOREIGN KEY
(IDProposition) REFERENCES Proposition(IDProposition);

ALTER TABLE se_compose_propo ADD CONSTRAINT FK_se_compose_propo_IDproduit FOREIGN KEY
(IDproduit) REFERENCES Produit(IDproduit);

ALTER TABLE fournit_recipient ADD CONSTRAINT FK_fournit_recipient_IDFournisseur FOREIGN KEY
(IDFournisseur) REFERENCES Fournisseur(IDFournisseur);

ALTER TABLE fournit_recipient ADD CONSTRAINT FK_fournit_recipient_IDrecipient FOREIGN KEY
(IDrecipient) REFERENCES Recipient(IDrecipient);
```