

UTBM - Département Informatique

**Projet – BD40**

**Sujet n.3 : Cave à vin**

*Automne 2018 – 10/10/2018*

**Groupe 3**

Anaïs JARNO

Haoyun LIAO

Weitao YAN

**TABLE DES MATIERES**

[1 Historique du document 1](#_Toc535018402)

[2 Description du sujet de projet 2](#_Toc535018403)

[2.1 Contexte du projet 2](#_Toc535018404)

[2.2 Domaine étudié 2](#_Toc535018405)

[2.3 Fonctionnement du système 3](#_Toc535018406)

[2.4 Acteurs et flux 4](#_Toc535018407)

[2.5 Périmètre du système 4](#_Toc535018408)

[2.6 Extensions possibles 4](#_Toc535018409)

[3 Dossier de conception du projet 5](#_Toc535018410)

[3.1 Modélisation conceptuelle 5](#_Toc535018411)

[3.1.1 Dictionnaire de données 5](#_Toc535018412)

[3.1.2 Modèle entité association 7](#_Toc535018413)

[3.1.3 Modèles de communication et de traitements 11](#_Toc535018414)

[4 La modélisation logique 12](#_Toc535018415)

[5 La modélisation de l’IHM 15](#_Toc535018416)

[6 Dossier de développement du projet 20](#_Toc535018417)

[6.1 Présentation de l’interface développée avec MS/ACCESS 20](#_Toc535018418)

[6.1.1 Page principale 20](#_Toc535018419)

[6.1.2 Page des vins connus 21](#_Toc535018420)

[6.2 Présentation des requêtes développées 22](#_Toc535018421)

[6.3 Présentation du code VBA développé dans l’application 25](#_Toc535018422)

[6.3.1 Modules VBA 25](#_Toc535018423)

[6.3.2 VBA pour formulaires 27](#_Toc535018424)

[6.3.3 Code via DAO 27](#_Toc535018425)

[6.4 Présentation des options de sécurité 28](#_Toc535018426)

[6.4.1 Mise en place ruban personnalisé 28](#_Toc535018427)

[6.4.2 Le verrouillage de l’accès aux données et la réactualisation des liens 31](#_Toc535018428)

[6.4.3 Contrôle des saisies 34](#_Toc535018429)

[6.4.4 Conclusion sur la sécurité 34](#_Toc535018430)

[7 Conclusion sur le Projet 35](#_Toc535018431)

[8 Sources 36](#_Toc535018432)

# Historique du document

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Auteur** | **Modifications du document** |
| Etape 1 | 09/10/2018 | Anaïs J. | Création du rapport, page de garde, table des matières et mise en page |
|  | 13/10/2018 | Anaïs J. | Création d’un schéma « résumé » pour la partie explicative du projet |
|  | 21/10/2018 | Weitao | Rédaction du contenu 2.3 et 2.4 |
|  | 21/10/2018 | Haoyun | Rédaction du contenu 2.1 et 2.2 |
|  | 21/10/2018 | Anaïs J. | Création du MCD (entité-relation) |
|  | 21/10/2018 | Weitao | Création du modèle de communication (Diagramme de flux) |
|  | 21/10/2018 | Haoyun et Anaïs | Modifications et correction des parties 2.1, 2.2 |
|  | 22/10/2018 | Weitao | Modification du contenu 2.3 et 2.4 et rédaction du 2.5 |
|  | 22/10/2018 | Anaïs J. | Ajout de la partie 2.6 et modification du MCD |
|  | 22/10/2018 | Weitao | Création du MCT et modification du MCC |
|  | 22/10/2018 | Haoyun | Création du dictionnaire |
|  | 26/10/2018 | Anaïs J. | Correction du MCD |
|  | 05/11/2018 | Weitao | Modification du MCT et MCC |
|  | 07/11/2018 | Anaïs J. | Dernières modifications du MCD, du dictionnaire et des parties avant rendu |
| Etape 2 | 23/11/2018 | Anaïs J. | Correction MCD, orientation des pages et tri du dictionnaire |
|  | 04/11/2018 | Weitao | Ajout de la partie Normalisation |
|  | 04/11/2018 | Anaïs J. | Ajout et rédaction de la partie IHM |
| Etape 3 | 28/12/2018 | Anaïs J. | Rédaction des options de sécurité |
|  | 10/01/2019 | Anaïs J. | Fin de la rédaction de toute la partie Dossier de développement |
|  | 11/01/19 | Anaïs J. | Correction + conclusion |
|  | 12/01/19 | Anaïs J. | Ajout de la partie traitant de la liaison des tables |

# Description du sujet de projet

## Contexte du projet

Nous sommes des étudiants en première année d’informatique à l’UTBM. Dans le cadre de l’UV de BD40 concernant la conception de systèmes d’informations, nous avons à réaliser un projet pour mettre en application les connaissances acquises durant les cours et les séances de travaux pratiques. Les membres de notre groupe étaient très intéressés et curieux de découvrir le sujet du vin. Le groupe étant composé de deux chinois, nous pensions que ce serait une bonne occasion pour approfondir un sujet très lié à la culture française.

C’est ainsi que nous avons donc choisi le sujet de la cave à vin.

## Domaine étudié

Nous avons choisi de restreindre le domaine à une utilisation personnelle.

Le principe est de créer une application de base de données permettant la gestion de cave à vin personnelle. Elle devra ainsi permettre à son utilisateur de visualiser le contenu de sa cave, les informations concernant ses vins et bouteilles, et d’ajouter ou de retirer des bouteilles de sa cave.

Nous étudions donc les diverses fonctions nécessaires à l’application, telles que la conservation du vin, l’enregistrement de la date d'achat, de la meilleure date de consommation (l’apogée) etc.

Nous avons alors réalisé un schéma servant à résumer les caractéristiques principales de notre gestion de cave :

Une image contenant texte, carte

Description générée avec un niveau de confiance très élevé

## Fonctionnement du système

Ce système d’information s’adresse à la gestion d’une cave à vin personnelle. On distingue le vin et la bouteille. Les termes ‘*fournisseur’* et ‘*producteur’* indiqueront le même acteur.

Des fournisseurs vont produire des vins de différentes régions et ayant d’autres caractéristiques comme la couleur, le cépage, l’appellation, etc.

Le client peut acheter plusieurs bouteilles à la fois, ou une seule. La cave à vin sert à stocker des bouteilles achetées par le consommateur. Les bouteilles sont stockées dans la cave jusqu’au jour où le consommateur veut la déguster.

Dès que la bouteille est sortie de la cave pour être dégustée, celle-ci est retirée de la liste des bouteilles, ou sa quantité est réduite. Une date d’entrée est également notée dès que la bouteille entre en stock.

On a également une fiche de dégustation qui permet d’enregistrer toutes les informations concernant la dégustation pour que le moment venu le consommateur puisse choisir le vin le plus adapté à son désir, à son repas...

## Acteurs et flux

|  |  |  |  |
| --- | --- | --- | --- |
| Les acteurs externes | Les acteurs internes | Le flux primaire | Les flux secondaires |
| Producteur de vin | * Le consommateur * La cave | Les entrées-sorties de bouteilles dans la cave | * Le stockage du vin (information enregistrée) * La dégustation du vin * L’achat du vin |

## Périmètre du système

Notre système est très simple. C’est une cave à vin personnelle. C’est-à-dire la cave sert à stocker des bouteilles pour un foyer ou pour une personne.

De temps en temps, l’utilisateur achètera une bouteille ou plusieurs bouteilles, qu’il fera alors entrer dans sa cave.

Puis, il invitera des amis, des parents ou des collègues chez lui et ils pourront alors déguster l’un des vins contenus dans la cave. Par la suite, il pourra alors remplir une fiche de dégustation. Cela permet de noter les avis après dégustation pour que l’utilisateur puisse choisir la prochaine fois son vin préféré, ou le plus adapté à la situation.

## Extensions possibles

Les extensions possibles du projet, c’est-à-dire les options que l’on ferait dans un contexte plus professionnel et non scolaire, pourrait être du type :

* Système de facture lié à l’enregistrement d’une arrivée de stock
* Revente de bouteilles
* Commande directe depuis l’application pour un vin spécifique vers le producteur concerné et gestion de la livraison

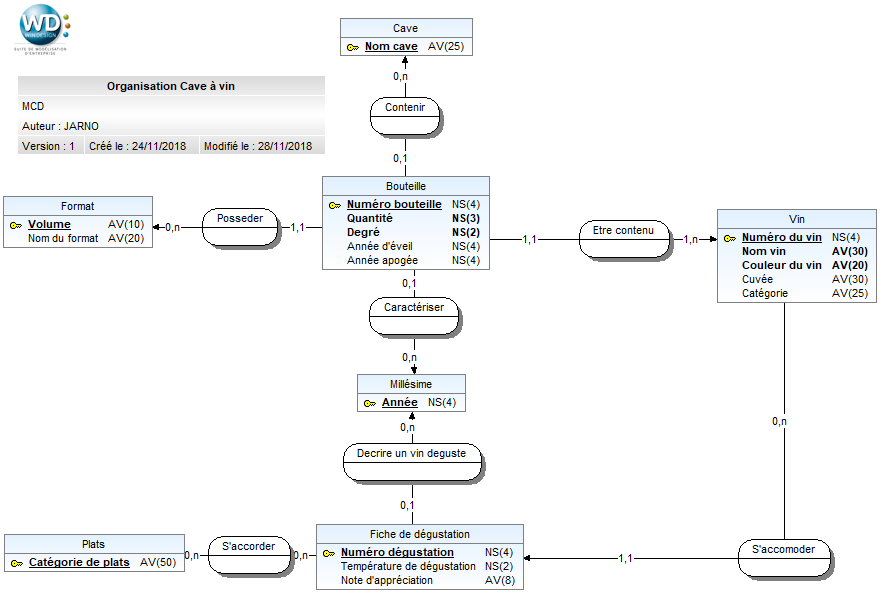
# Dossier de conception du projet

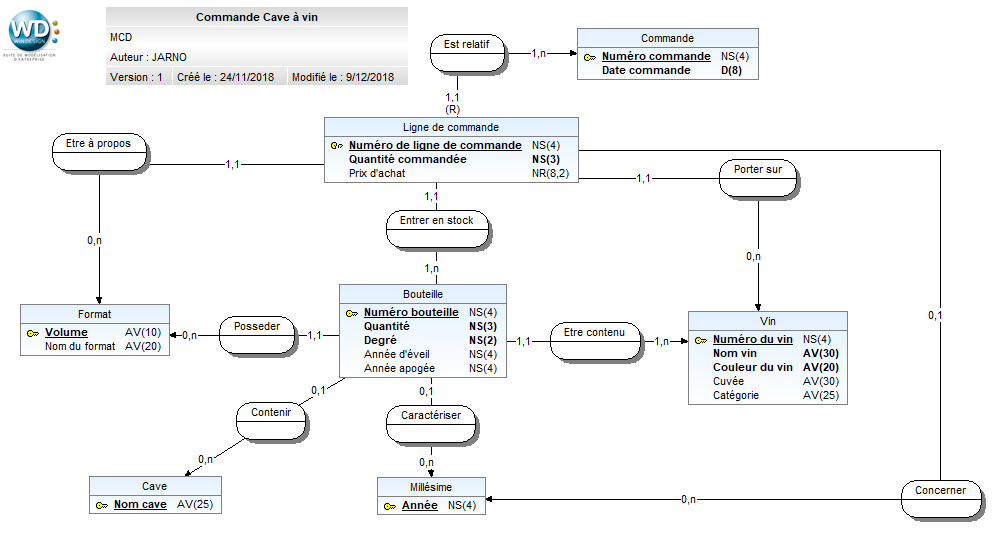
## Modélisation conceptuelle

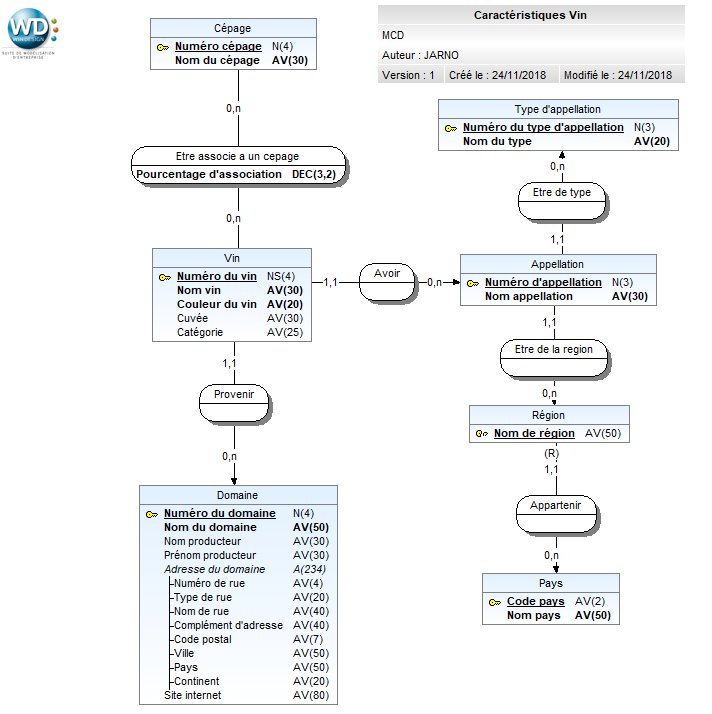
### Dictionnaire de données

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom Conceptuel | Nom Logique ou Alias | Type  (E, Ca, Co) | Nature + Longueur | Type  Win’Design | Identifiant |
| Adresse du domaine | ADR\_DOMAINE | Co | AN234 | A(234) |  |
| Numéro de rue | NUM\_RUE\_DOMAINE | E | N4 | AV(4) |  |
| Type de rue | TYPE\_RUE\_DOMAINE | E | A20 | AV(20) |  |
| Nom de rue | NOM\_RUE\_DOMAINE | E | A40 | AV(40) |  |
| Complément d’adresse | COMPL\_ADR\_DOMAINE | E | A40 | AV(40) |  |
| Ville | VILLE\_DOMAINE | E | A50 | AV(50) |  |
| Code postal | CP\_DOMAINE | E | A7 | AV(7) |  |
| Pays | PAYS\_DOMAINE | E | A50 | AV(50) |  |
| Continent | CONTINENT\_DOMAINE | E | A20 | AV(20) |  |
| Année | ANNEE\_MILLESIME | E | N4 | NS(4) | OUI |
| Année apogée | ANNEE\_APOGEE | E | N4 | NS(4) |  |
| Année d’éveil | ANNEE\_EVEIL | E | N4 | NS(4) |  |
| Catégorie | CATEG\_VIN | E | A25 | AV(25) |  |
| Catégorie de plats | CATEGORIE\_PLATS | E | A50 | AV(50) |  |
| Code pays | CODE\_PAYS | E | A2 | AV(2) | OUI |
| Couleur du vin | COULEUR\_VIN | E | A20 | AV(20) |  |
| Cuvée | CUVEE\_VIN | E | A30 | AV(30) |  |
| Date commande | DATE\_COMMANDE | E | AN8 | D(8) |  |
| Degré | DEGRE\_BOUTEILLE | E | N2 | NS(2) |  |
| Nom appellation | NOM\_APPELLATION | E | A30 | AV(30) |  |
| Nom cave | NOM\_CAVE | E | A25 | AV(25) | OUI |
| Nom de région | NOM\_REGION | E | A50 | AV(50) | OUI |
| Nom du cépage | NOM\_CEPAGE | E | A30 | AV(30) |  |
| Nom du domaine | NOM\_DOMAINE | E | A50 | AV(50) |  |
| Nom du format | NOM\_FORMAT | E | A20 | AV(20) |  |
| Nom du type | NOM\_TYPEAPP | E | A20 | AV(20) |  |
| Nom pays | NOM\_PAYS | E | A50 | AV(50) |  |
| Nom producteur | NOM\_PROD | E | A30 | AV(30) |  |
| Nom vin | NOM\_VIN | E | A30 | AV(30) |  |
| Note d’appréciation | NOTE\_DEGUST | E | A8 | AV(8) |  |
| Numéro bouteille | NUM\_BOUTEILLE | E | N4 | NS(4) | OUI |
| Numéro cépage | NUM\_CEPAGE | E | N4 | N(4) | OUI |
| Numéro commande | NUM\_COMMANDE | E | N4 | NS(4) | OUI |
| Numéro d’appellation | NUM\_APPELLATION | E | N3 | N(3) | OUI |
| Numéro dégustation | NUM\_DEGUST | E | N4 | NS(4) | OUI |
| Numéro de ligne de commande | NUMERO\_LIGNE\_DE\_COMMANDE | E | N4 | NS(4) | OUI |
| Numéro du domaine | NUM\_DOMAINE | E | N4 | N(4) | OUI |
| Numéro du type d’appellation | NUM\_TYPEAPP | E | N3 | N(3) | OUI |
| Numéro du vin | NUM\_VIN | E | N4 | NS(4) | OUI |
| Pourcentage d’association | POURCENTAGE\_CEPAGE | E | AN(3,2) | DEC(3,2) |  |
| Prénom producteur | PRENOM\_PROD | E | A30 | AV(30) |  |
| Prix d’achat | PRIX\_ACHAT | E | AN(8,2) | DEC(8,2) |  |
| Quantité | QUANTITE\_BOUTEILLE | E | N3 | NS(3) |  |
| Quantité commandée | QUANTITE\_COMMANDE | E | N3 | NS(3) |  |
| Site internet | SITE\_PROD | E | A80 | AV(80) |  |
| Température de dégustation | TEMP\_DEGUST | E | N2 | NS(2) |  |
| Volume | VOLUME\_FORMAT | E | A10 | AV(10) | OUI |

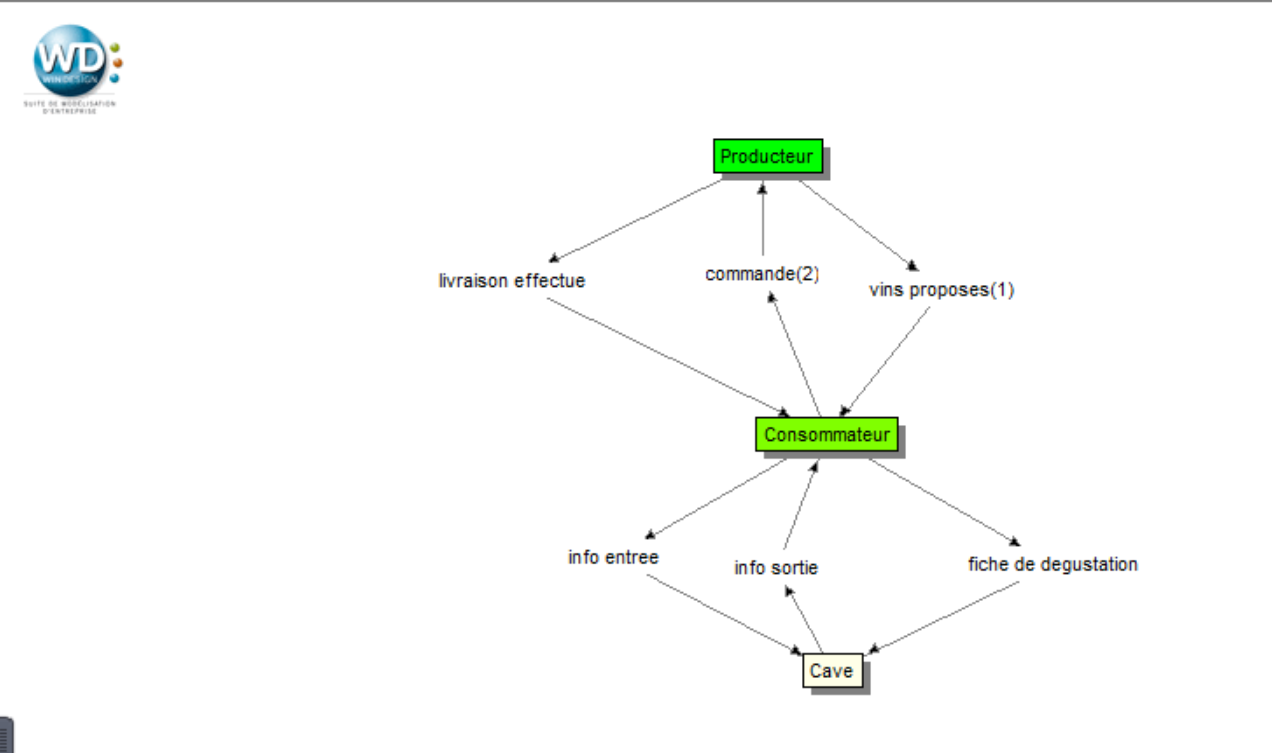
### Modèle entité association

**Décomposition du MCD global en sous-modèles :**





### Modèles de communication et de traitements



# La modélisation logique

**MLR-Sous modèle-Organisation**

**1FN : Oui**

*Justification* : Tous les attributs sont fonctionnellement dépendants de la clé.

**2FN : Oui**

*Justification* : Toutes les relations sauf ‘*S’ACCORDER*’ n’ont qu’une seule clé. Donc naturellement tout attribut non clé ne dépend pas d’une partie de cette clé. C’est-à-dire qu’ils sont en dépendance fonctionnelle élémentaire par rapport à la clé. Quant à ‘*S’ACCORDER*’, elle est en 2FN étant donné qu’elle n’a pas d’attribut non clé.

**3FN : Oui**

*Justification* :

*BOUTEILLE :* Tous les attributs non clé sont indépendants des autres.

*VIN :* Tous les attributs non clé sont indépendants des autres. Pour *NUM\_DOMAINE* et *NUM\_APPELLATION*, vu qu’ils se réfèrent à des entités différentes n’ayant pas d’associations entre elles. Ils sont ainsi indépendants.

*FICHE\_DEGUST* : Tous les attributs non clé sont indépendants des autres. Pour *NUM\_VIN* et *ANNEE\_MILLESIME*, vu que **MILLESIME** caractérise la bouteille, ils sont ainsi indépendants.

**Conclusion** : Ce sous-modèle est en 3FN, il est ainsi normalisé.

**MLR-Sous modèle-Commande**

**1FN : Oui**

*Justification* : Tous les attributs sont fonctionnellement dépendants de la clé.

**2FN : Oui**

*Justification* : Toutes les relations sauf ‘*LIGNE\_DE\_COMMANDE*’ et ‘*ENTRER EN STOCK*’ n’ont qu’une seule clé. Donc naturellement tout attribut non clé ne dépend pas d’une partie de cette clé. C’est-à-dire qu’ils sont en dépendance fonctionnelle élémentaire par rapport à la clé. Quant à ‘*ENTRER EN STOCK*’, elle est en 2FN étant donné qu’elle n’a pas d’attribut non clé. Quant à ‘*LIGNE\_DE\_COMMANDE*’, vu que l’identifiant de l’entité *LIGNE\_DE\_COMMANDE* est relatif, autrement dit, tout attribut non clé de *LIGNE\_DE\_COMMANDE* sont en dépendance fonctionnelle élémentaire par rapport à ces deux clés. Elle est ainsi en 2FN.

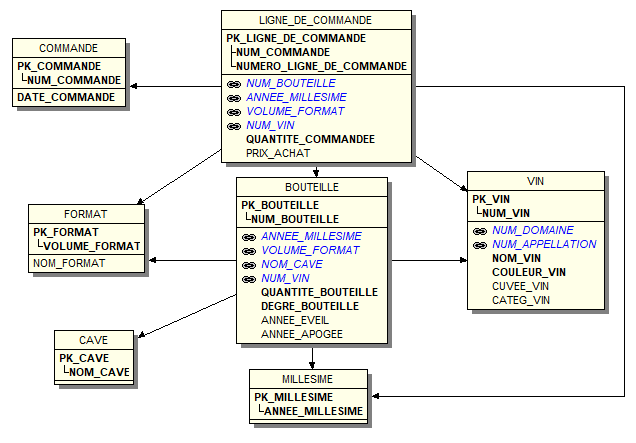
**3FN : Oui**

*Justification* :

*BOUTEILLE***:** Tous les attributs non clé sont indépendants des autres.

*VIN* : Tous les attributs non clé sont indépendants des autres. Pour **NUM\_DOMAINE** et **NUM\_APPELLATION**, vu qu’ils se réfèrent à des entités différentes n’ayant pas d’associations entre elles. Ils sont ainsi indépendants.

*LIGNE\_DE\_COMMANDE :*Tous les attributs non clé sont indépendants des autres.

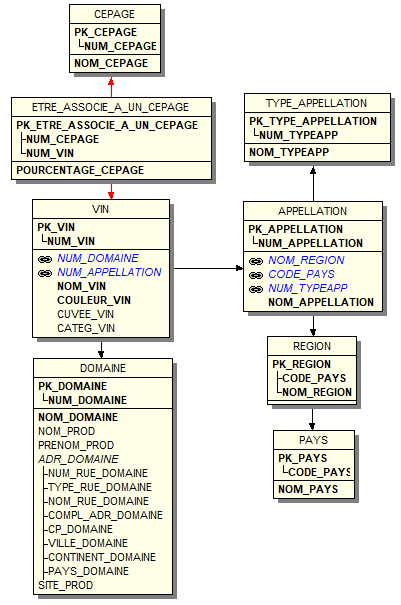
**Conclusion** : Ce sous-modèle est en 3FN, il est ainsi normalisé.

**MLR-Sous modèle-Commande**

**1FN : Oui**

*Justification* : Tous les attributs sont fonctionnellement dépendants de la clé. Dans *DOMAINE*, même s’il y a des propriétés composées, **ADR\_DOMAINE** ne fait pas partie des attributs non clé. De plus, chaque valeur de la clé (**NUM\_DOMAINE**) ne correspond qu’à une seule valeur des attributs composés. Elle est ainsi en 1FN.

**2FN : Oui**

*Justification* : Toutes les relations sauf ‘*ETRE\_ASSOCIE\_A\_UN\_CEPAGE*’ et ‘*REGION*’ n’ont qu’une seule clé. Donc naturellement tout attribut non clé ne dépend pas d’une partie de cette clé. C’est-à-dire qu’ils sont en dépendance fonctionnelle élémentaire par rapport à la clé. Quant à ‘*REGION*’, elle est en 2FN étant donné qu’elle n’a pas d’attribut non clé. Quant à ‘*ETRE\_ASSOCIE\_A\_UN\_CEPAGE*’, vu que **POURCENTAGE\_CEPAGE** pour un vin donné est déterminé en même temps par **NUM\_CEPAGE** et **NUM\_VIN**, il est en dépendance fonctionnelle élémentaire par rapport à ces deux clés. La relation est ainsi en 2FN.

**3FN : Oui**

*Justification* :

*VIN :* Tous les attributs non clé sont indépendants des autres. Pour **NUM\_DOMAINE** et **NUM\_APPELLATION**, vu qu’ils se réfèrent à des entités différentes n’ayant pas d’associations entre elles. Ils sont ainsi indépendants.

*APPELLATION :* Tous les attributs non clé sont indépendants des autres.

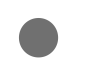
*DOMAINE* *:* Tous les attributs non clé sont indépendants des autres. Pour l’adresse, il existe des pays qui se situe dans deux continents. Donc **PAYS\_DOMAINE** et **CONTINENT\_DOMAINE** sont indépendants. Même dans un pays, il se peut qu’une ville a un homonyme, donc **VILLE\_DOMAINE** et **CP\_DOMAINE** sont aussi indépendants.

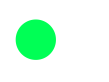
**Conclusion** : Toutes les relations dans MLR sont en 3FN, donc le MLR global est normalisé.

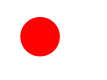
# La modélisation de l’IHM

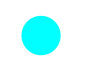
En réfléchissant à l’organisation orientée métier de notre application, dans le but d’être utilisée par une personne lambda, nous avons alors découpé notre IHM en quelques formulaires qui nous semblaient être les plus pertinents.

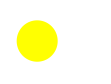
Les informations ont été triées en différentes catégories :

Informations non modifiables

Informations pouvant être modifiées

Informations obligatoires

Informations optionnelles

Informations calculées

En ouvrant son application, l'utilisateur peut voir les informations de base de tous les vins qu’il connaît déjà. Il peut sélectionner le vin dans le formulaire pour afficher des informations spécifiques à celui-ci. Il peut également trouver des informations sur certains vins en effectuant une recherche selon le nom du vin et en filtrant selon sa couleur.

De même, il peut modifier certaines informations spécifiques au vin en question en accédant au bouton « *Modifier* ». Ce qui nous donne la maquette suivante :



Figure 5‑1 Affichage principal - Liste des vins

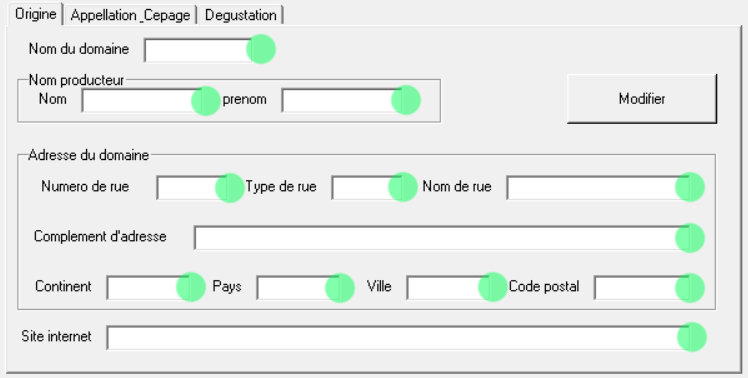


Figure 5‑2 Affichage principal - L’onglet de l’origine du vin

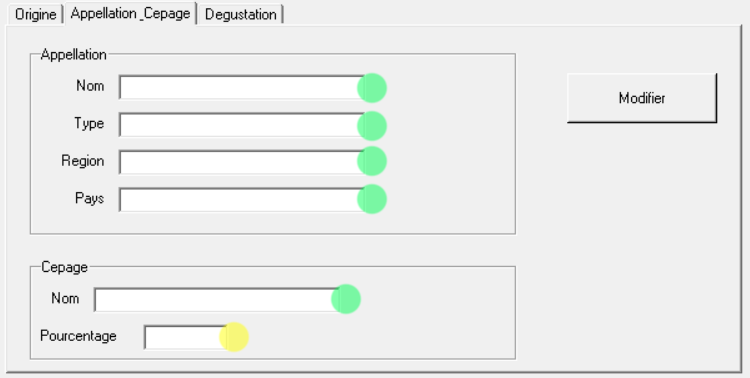


Figure 5‑3 Affichage principal - L’onglet de l’appellation et le cépage du vin

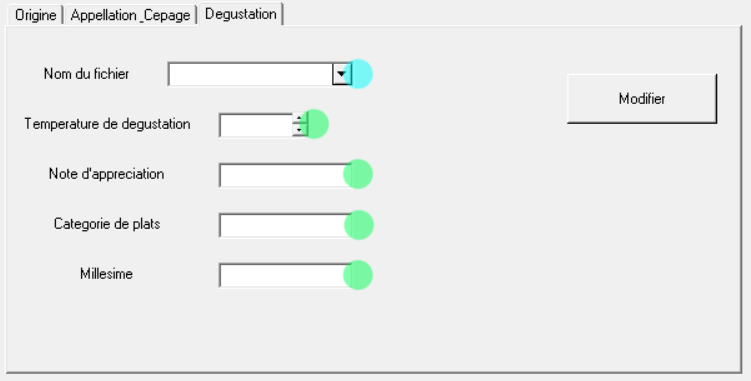


Figure 5‑4 Affichage principal - L’onglet de la dégustation du vin

Considérons la sortie d’une bouteille. L'utilisateur peut renseigner les noms des vins dont il souhaite retirer des bouteilles, toutes les bouteilles du même nom de vin seront affichées au bas de la page. L'utilisateur retire une unité en validant la sélection de la bouteille.

Ce même utilisateur peut également ajouter une nouvelle bouteille en choisissant le nom du vin puis en renseignant les informations de base.

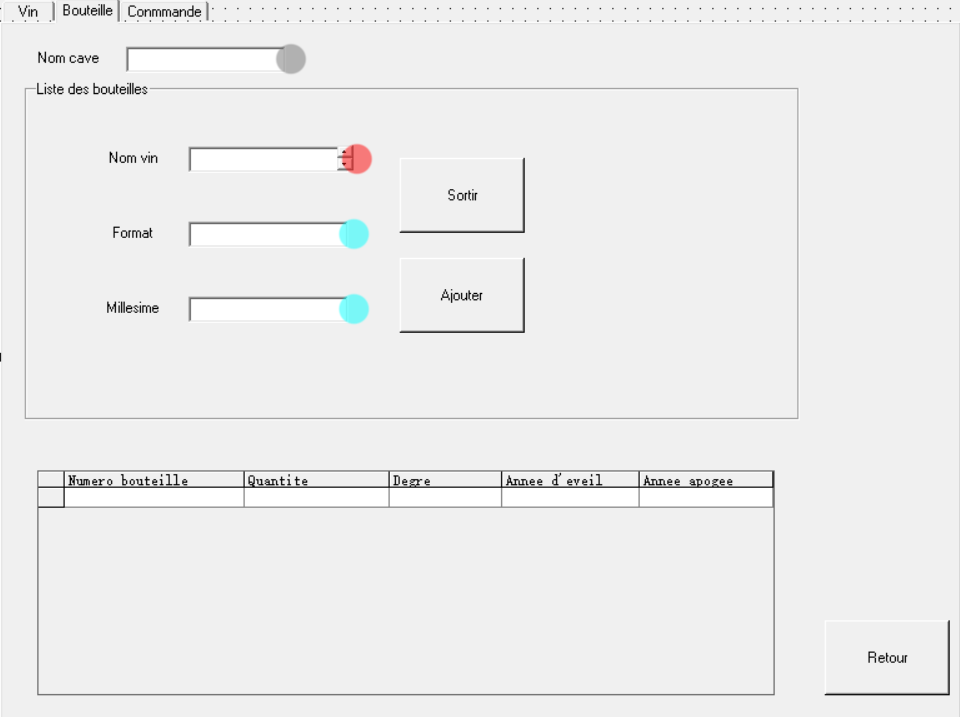


Figure 5‑5 Sortir et ajouter bouteille

De même, nous avons une page pour voir toutes les informations concernant les commandes.

Selon la date entrée, le tableau ci-dessous affiche les informations de commande. En sélectionnant une commande, toutes les informations concernant les bouteilles comprises dans la commande sélectionnée seront affichées en bas de la page.

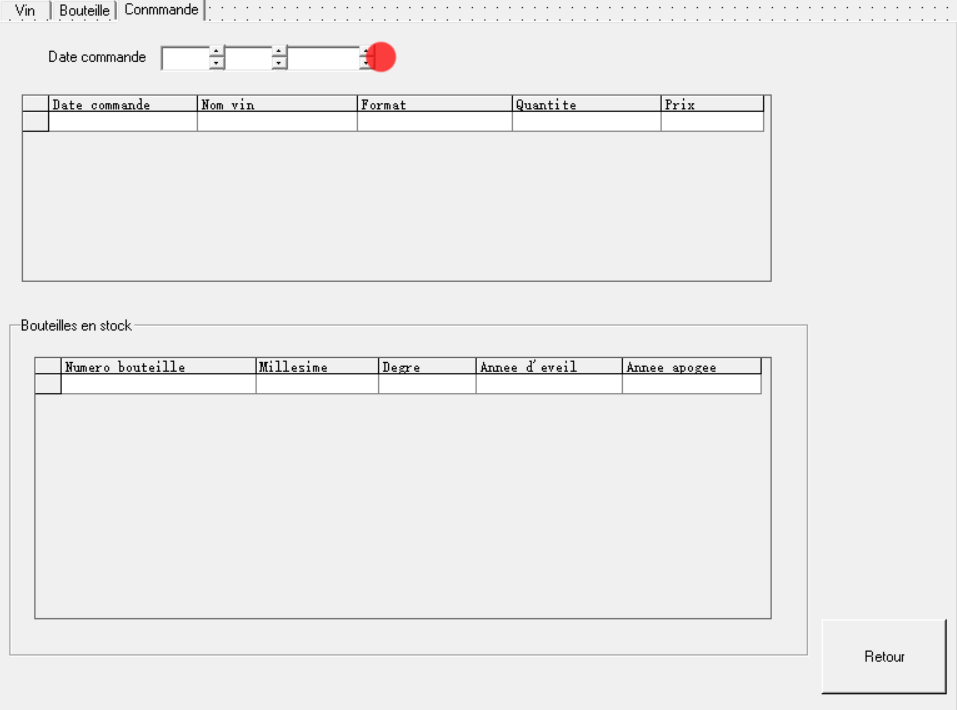


Figure 5‑6 Voir les informations de commande

L’utilisateur peut entrer une nouvelle commande de vin via notre application. Les informations sur le vin pouvant être commandé sont affichées en bas de la page. L'utilisateur peut choisir le vin ou entrer le nom du vin pour déterminer le vin qu’il souhaite acheter. L'utilisateur entre ensuite les informations de base. L'utilisateur peut alors valider l’entrée de sa commande en appuyant sur le bouton « Commander ».

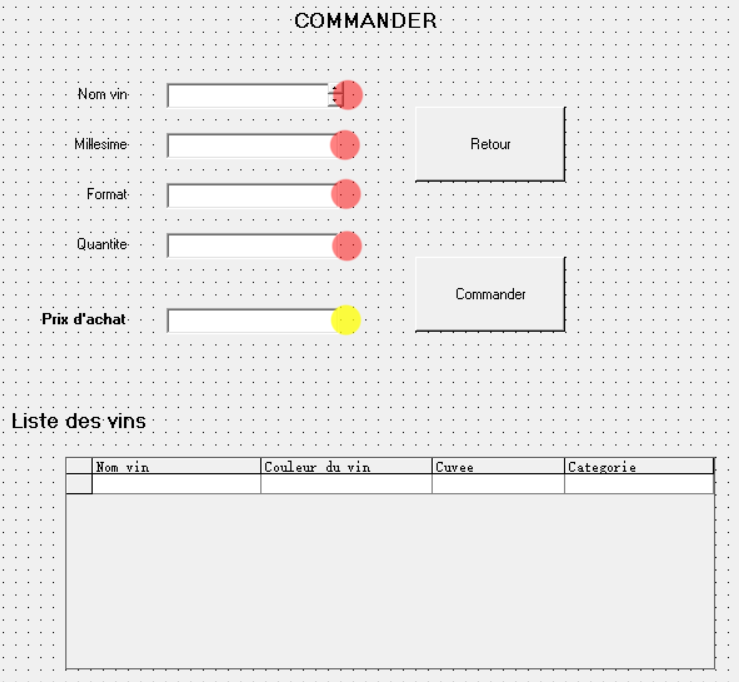
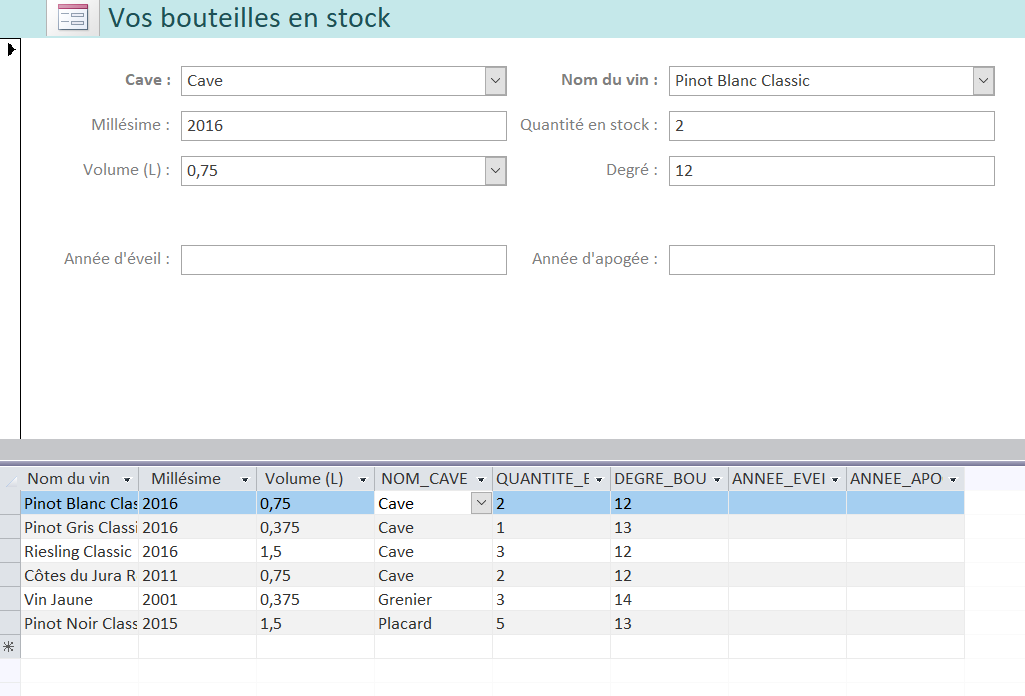


Figure 5‑2 Visuel de commande de bouteilles

# Dossier de développement du projet

## Présentation de l’interface développée avec MS/ACCESS

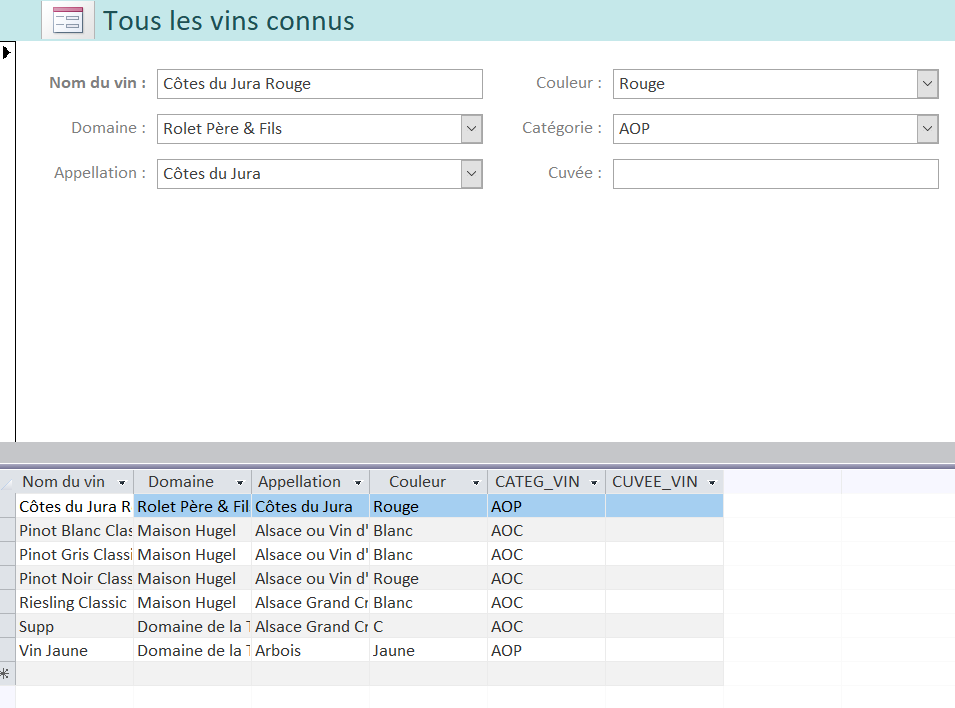
### Page principale

Tout d’abord, il nous a semblé important que le programme s’ouvre directement sur la page permettant l’accès aux bouteilles, car c’est la fonctionnalité qui devrait intéresser le plus l’utilisateur. Ainsi notre page d’accueil et page principale correspond à la suivante :

### Page des vins connus

Ensuite, une autre fonctionalité importante pour l’utilisateur était d’avoir accès aux vins qu’il connaît déjà, et bien sûr de pouvoir ajouter de nouveaux vins.

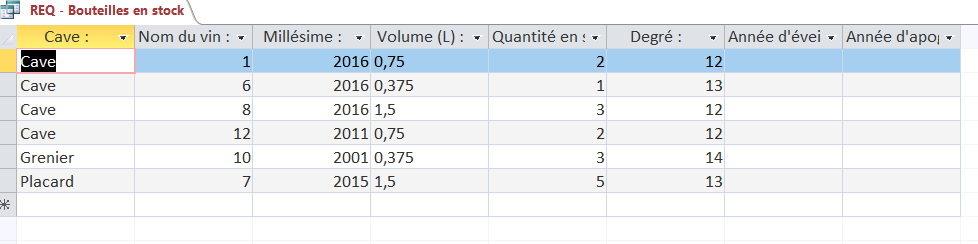
C’est ainsi que sur cette page, nous pouvons voir tous les vins connus de l’utilisateur.



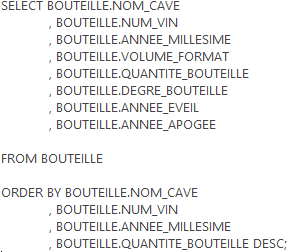
## Présentation des requêtes développées

Nous avons deux requêtes principales pour les formulaires et les autres requêtes sont utilisées pour remplir les Combobox.

La requête suivante est celle utilisée pour notre formulaire principal, *Bouteilles en stock*

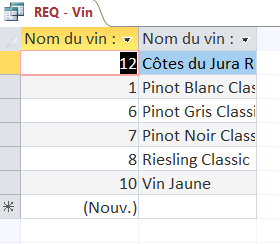


Le code SQL associé est le suivant :



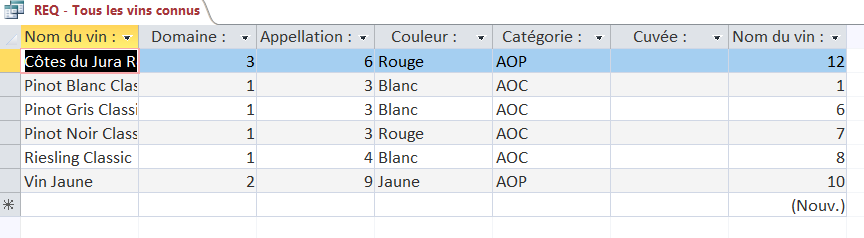
Il nous permet de sélectionner les champs voulus tout en exerçant un tri sur certaines valeurs.

Ensuite, nous avions besoin du nom du vin plutôt que de son numéro pour l’affichage. Car en effet, on affiche le numéro du vin car c’est le champ present dans bouteille. Cependant celui-ci ne signifie rien pour l’utilisateur, il lui faut donc plutôt son nom, présent dans la Combobox remplie grâce à la requête suivante :



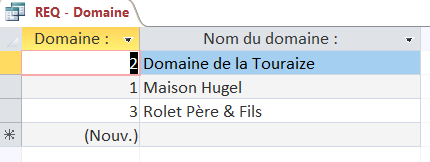
De la même façon nous avions besoin des noms de cave, et nous avons donc rempli la Combobox par les valeurs de la table Cave étant donné que son Identifiant est aussi son nom. Nous avions également besoin du Volume, de la capacité de contenance de la bouteille. Nous avons traité cette Combobox de la même manière que celle pour les caves, à la difference près que nous avons ajouter l’affichage du nom de format avec le volume associé.

Puis, pour ce qui est du formulaire des vins nous avons la requête suivante :

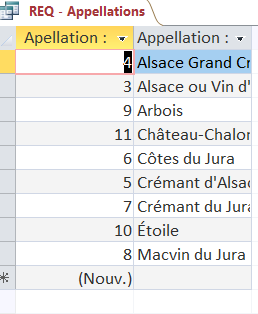


Nous avons fonctionné de la même manière que pour le formulaire sur les bouteilles.

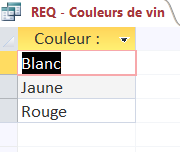
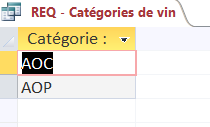
C’est ainsi que nous avons eu besoin d’une requête pour le domaine:



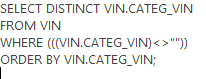
De la même façon pour les appellations:



Par la suite, nous avons voulu proposer à l’utilisateur de choisir la couleur de son vin et sa catégorie via une autre Combobox. Cependant ces valeurs ne sont que des attributs de la table VIN et non des tables à part entière.

C’est donc ainsi que nous avons formulés les requêtes suivantes :

Dont les codes *SQL* associés sont les suivants :

Ceci nous permet de sélectionner toutes les categories/couleurs de vin et de les trier. De plus le mot de clef DISTINCT au niveau du SELECT permet de sélectionner de manière unique les éléments. C’est-à-dire que s’il y plusieurs vins avec attribut *jaune* pour la couleur, *jaune* n’apparaîtra qu’une seule fois dans la requête.

## Présentation du code VBA développé dans l’application

### Modules VBA

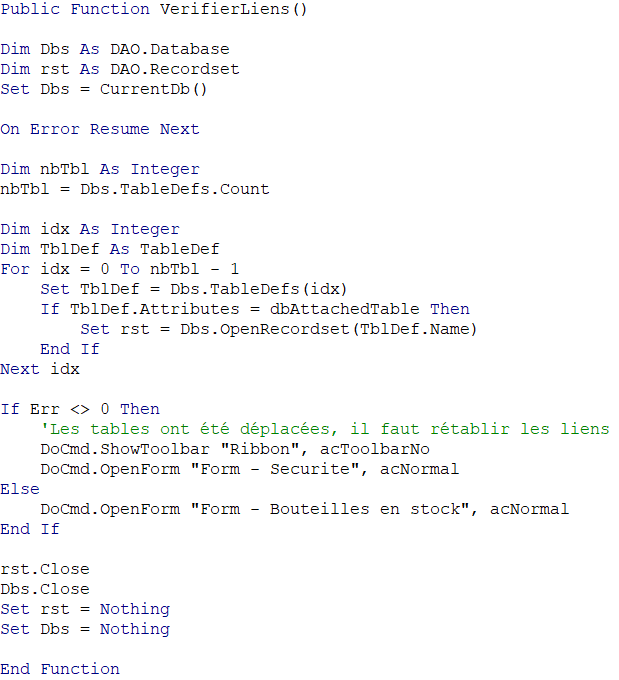
Nous avons 4 principaux modules :

* Désactivation de la Touche *MAJ*
* Vérification des liens
* Chargement du ruban
* Ruban contrôle

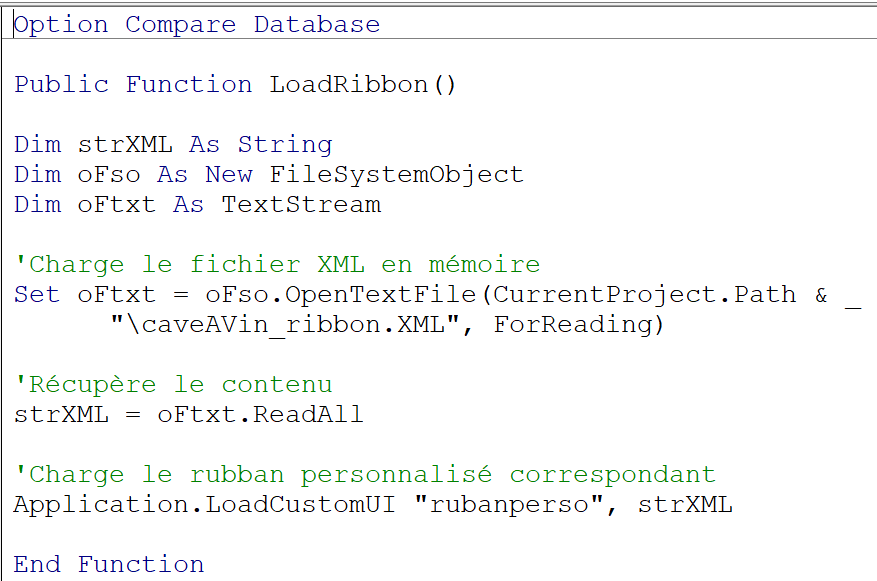
La désactivation de la touche *MAJ* est décrite un peu plus tard dans la partie sur les options de sécurité.

Le module de vérification des liens permet d’appeler le formulaire de sécurité exécutant la protection de la base de données et la réactualisation des liens si ceux-ci ne sont plus valables. Autrement, il ouvre juste le formulaire d’accueil.

Il est ainsi codé :



Le module permettant le chargement du ruban personnalisé en *XML* a été codé de la façon suivante :

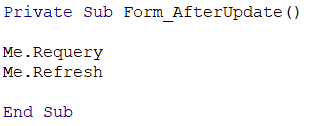


Le module Ruban contrôle configure les boutons ***Mes bouteilles en stock*** et ***Mes vins.*** En effet, on définit la réaction de ces deux boutons. Lorsque *Mes vins* est pressé, on ouvre le formulaire concernant les vins. Tandis que lorsque l’on clique sur *Mes bouteilles*, en plus d’ouvrir le formulaire correspondant, cela permet de réactualiser les enregistrements (et donc les Combobox présentes dans le formulaire).



### VBA pour formulaires

Une méthode est appelée pour les différents formulaires, celle-ci est la suivante :



Elle permet de réactualiser le formulaire courant et ses enregistrements, ce qui met à jour les Combobox présentes dans le formulaire.

### Code via DAO

Les seuls codes comportant du DAO sont ceux s’occupant des options de sécuritédétaillés dans la partie suivante.

## Présentation des options de sécurité

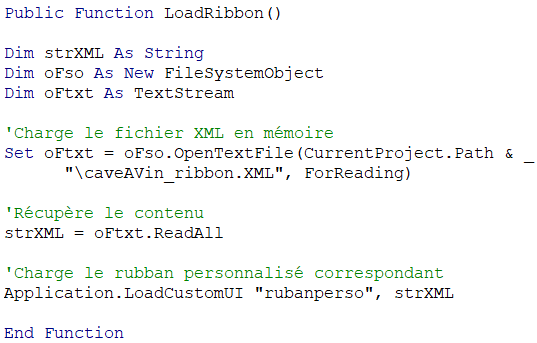
Nous avons sécurisé l’application de diverses manières.

Tout d’abord, il nous a semblé important que l’utilisateur n’ait pas accès à toutes les options proposées par Access. Autrement, il aurait pu créer de nouvelles tables, requêtes… Pour cela, nous avons mis en place un ruban personnalisé.

### Mise en place ruban personnalisé

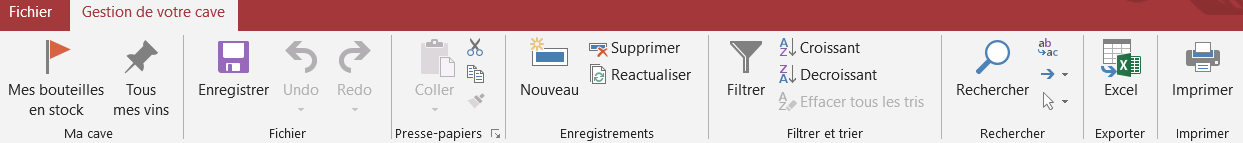
**Celui-ci est implémenté grâce au fichier suivant : ***caveAVin\_ribbon.XML***

Il est ensuite chargé grâce à la macro/fonction VBA suivante : ***LoadRibbon***



Celle-ci est à son tour appelée par la macro ***AutoExec***. Elle charge alors le ruban et comme nous avons défini dans les *options de la base de données actuelle* que le ruban principal était le ruban personnalisé, alors celui-ci est tout de suite lancé et remplace celui de la base Access. En effet, toute macro nommée ***AutoExec*** est exécutée à chaque fois au démarrage de l’application.

Dans notre application, nous avons alors le ruban suivant :



**Les différents onglets choisis et mis en place :**

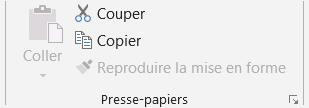
* Ma cave

Cet onglet comporteune galerie *Mes outils* avec les trois boutons (*Mes bouteilles en stock* – *Tous mes vins* – *Commander*) qui renvoient les formulaires principaux de gestion de ceux-ci. Le code XML de ces boutons a une fonction *onAction* faisant appel à une macro VBA implémentée par nos soins, servant à ouvrir les formulaires correspondants.

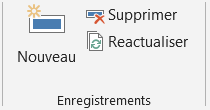
* Fichier

Onglet personnalisé permettant *la sauvegarde du fichier*, *le retour arrière* et *le retour avant*, boutons XML pré-codés par la bibliothèque office.

* Presse-papiers

Onglet pré-codé par la bibliothèque office.

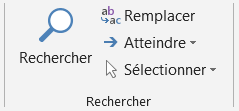
* Enregistrements

Onglet personnalisé utilisant les boutons XML pré-codés par la bibliothèque office permettant *la création d’un nouvel enregistrement*, *la suppression d’un enregistrement* et *la réactualiser de tout le document*.

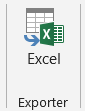
* Filtrer et trier

Onglet personnalisé limitant les filtres, et les tris aux *filtres généraux* et au *tri croissant/décroissant*, contrôles XML pré-codés par la bibliothèque office.

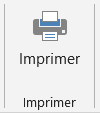
* Rechercher

Onglet pré-codé par la bibliothèque office, permettant la *recherche* et le *déplacement parmi les enregistrements*.

* Exporter

Onglet personnalisé permettant *l’exportation de la page courantes vers un fichier Excel*, bouton XML pré-codé par la bibliothèque office.

* Imprimer

Onglet pré-codé par la bibliothèque office, permettant *l’aperçu et l’impression*.

Aussi, la macro ***AutoExec*** ouvre le formulaire principal en tant que page d’accueil pour l’utilisateur.

### Le verrouillage de l’accès aux données et la réactualisation des liens

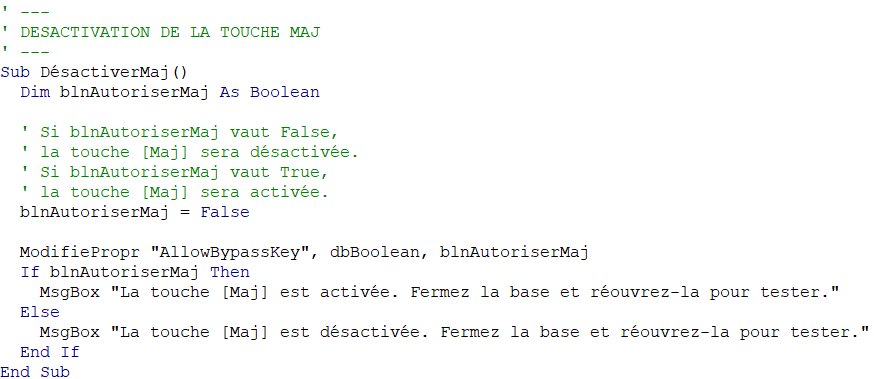
#### Pour l’application

Nous avons également mis en place une application pour verrouiller l’accès aux données.

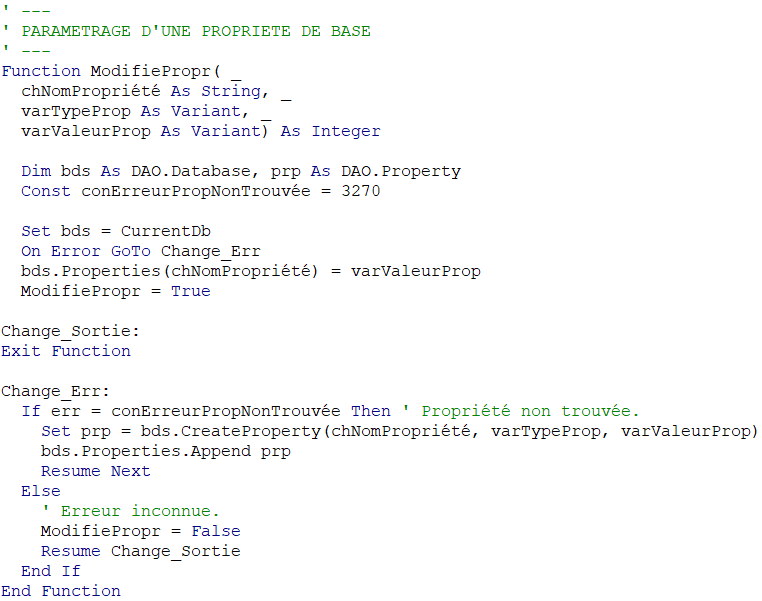
Nous avons choisi de restreindre l’accès de l’utilisateur aux Tables, Requêtes, Formulaires, Macros et Modules habituellement accessibles au travers du volet de navigation.

1. Ainsi, via les ***options de la base de données***, nous avons choisi de **masquer le Volet de Navigation**.

Cependant, cette sécurité n’était pas suffisante. Il suffisait d’appuyer sur la touche *MAJ* au démarrage de l’application pour accéder aux données, formulaires, etc.

1. Grâce à un ***code VBA***, nous avons donc la possibilité de **désactiver la touche *MAJ***, et ainsi de verrouiller l’accès aux tables.

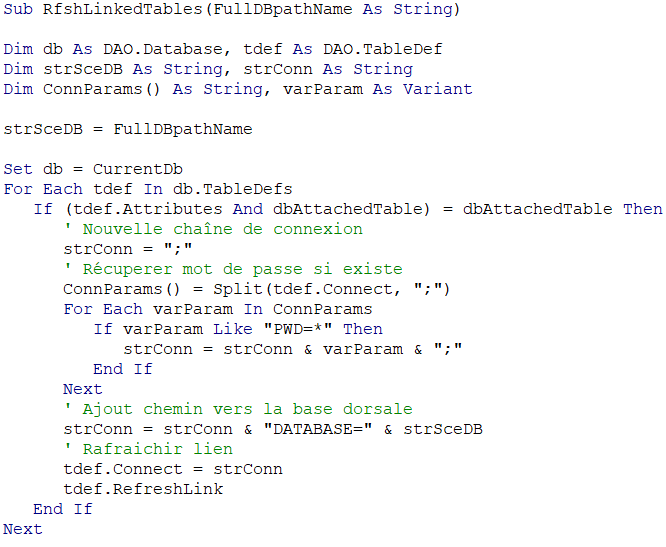
Cette macro appelle une ***fonction VBA DAO*** pour modifier directement la propriété *AllowBypassKey* qui définit si la touche *MAJ* peut être utilisée ou non pour l’ouverture du centre de navigation.



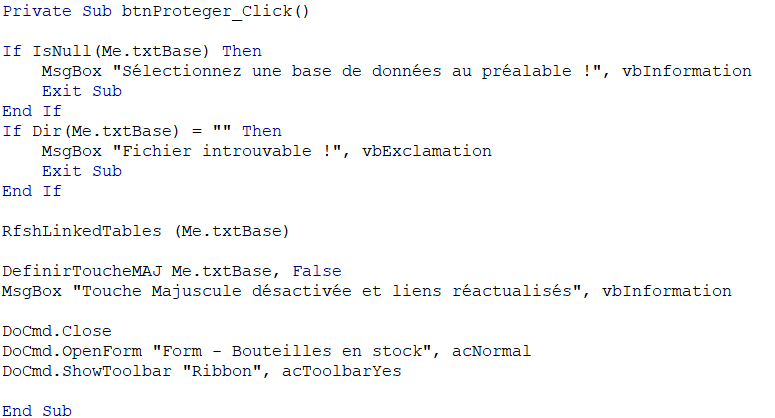
Une fois la macro lancée, il a alors suffi de fermer, puis de rouvrir la base de données pour que la désactivation soit effective.

#### Pour la base de données

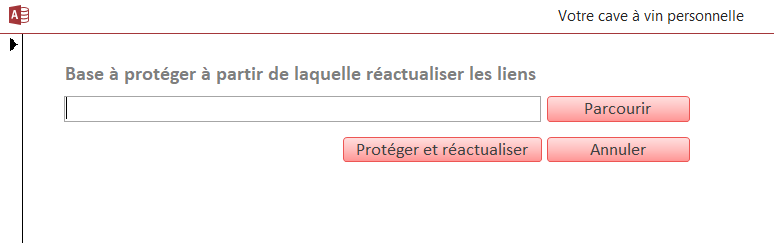
Nous avons séparé l’application de la base de données en deux fichiers. Cependant lorsque la base de données est déplacée ou renommée, celle-ci n’est plus trouvée par l’application, il fallait donc une manière de réactualiser les liens entre les tables et l’application. Pour cela nous avons utilisé le code VBA suivant :



Ce code fait partie des options appelées par le clic sur le bouton *Protéger et Réactualiser* du formulaire de sécurité. En effet, voici le code de ce bouton :



Comme précisé dans la fin du code, on utilise *ShowToolbar* qui permet d’afficher ou non le ruban. Pour des mesures de sécurité, nous avons désactivé l’affichage du ruban lorsque le formulaire de sécurité est appelé. Voici donc l’affichage produit :



Si les liens ont bien été réactualisés, on ouvre alors le formulaire principal et on affiche le ruban.

### Contrôle des saisies

Aussi des contrôles concernant les saisies ont été réalisés.

Par exemple, le millésime doit être supérieur à 1800 lors de la saisie, autrement elle est refusée et un message d’erreur averti l’utilisateur.

Aussi, certains champs parmi les *Combobox* refusent toute nouvelle saisie car il est nécessaire de faire un nouvel enregistrement dans la base de données pour le champ donné. Ainsi, l’option refusant la saisie d’un nouvel élément dans la *Combobox* permet à l’utilisateur d’être encadré car il ne lui sera alors pas possible d’essayer de rentrer cet élément, et donc il ne recevra pas d’erreur lorsqu’il voudra valider la saisie de sa nouvelle bouteille, ou de son nouveau vin.

### Conclusion sur la sécurité

Ainsi lorsque l’application est lancée, même en maintenant la touche MAJ lors du démarrage de l’application, l’utilisateur ne peut accéder au volet de navigation. De même il visualise le formulaire de base de l’application et n’a accès qu’au ruban personnalisé et non à celui de base. L’utilisateur est donc contraint d’utiliser les fonctionnalités à sa portée, fonctionnalités n’ayant pas d’influence sur la sécurité de l’application.

# Conclusion sur le Projet

Nous avions choisi ce thème pour permettre aux chinois présents dans le groupe de découvrir un thème fortement lié à la culture française. Cela s’est résulté en une grande difficulté ; freinant considérablement notre avancée, faute d’entente entre les membres du groupe ne comprenant pas le projet de la même façon.

Toutefois, nous remarquerons que nous avons tout de même pu mener à bien la partie conceptuelle du projet grâce un gros investissement...

Malgré tout, la partie concernant le développement de l’applicatif fut difficile à achever. En effet, il manque nombre de fonctionnalités en comparaison du modèle fourni. On notera néanmoins que les parties fournies sont fonctionnelles et peuvent être parfaitement utilisées. Une réelle utilisation de l’application se révélerait cependant complexe du fait du manque de fonctionnalités présentes.

Les buts concernant le développement de l’applicatif ne sont donc que partiellement atteints.

Ainsi, l’application telle qu’elle est, est fonctionnelle mais ne se prête pas à l’utilisation.

# Sources

***Sites internet***

<https://www.linternaute.com/homme/art-de-vivre/1011553-appellation-cepage-domaine-quelles-differences/>

<https://www.le-vin-pas-a-pas.com/4-niveaux-a-connaitre-pour-bien-classer-les-vins/>

<http://www.vinsvignesvignerons.com/Regions/Alsace/LA-REGION-ALSACE/Les-appellations>

***Consultation de diverses personnes ayant des connaissances en vitiviniculture et même en œnologie***

***Toutes les images présentes dans notre rapport, sont citées dans nos sources, libres de droits ou alors réalisées par nos propres soins.***