

ADDE Mathis
AMON Pauline
AUBRY Gatien
ARNAUDO Alexandre
BABOK Julien
ENGEL Victoria

Guide de fonctionnement VBA

Projet Base de données VBA/SQL

Sommaire:

1/ Gestion du catalogue produits et des fournisseurs	3
1. Ajout Fournisseur:.....	3
2. Suppression Fournisseur:.....	4
3. Affichage Fournisseur :	6
4. Ajout Produit:.....	7
5. Suppression/Edition Produit:	8
6. Affichage Produit:	10
7. Bilan Stocks :	12
8. Prix Moyen :.....	13
2/ Gestion des commandes clients	14
9. Afficher Consommateur:.....	14
10. Liste Commande selon Statut:	17
11. Liste Commande avec Livraison Offerte :	18
12. Qualité des livraisons :	20
13. Qualité des commandes :	21
3/ Gestion commerciale	22
14. Bilan par catégorie de produits :.....	22
15. Bilan des ventes annuelles :	23
16. Produits vendus par producteur et par département :	25
17. Alerte des livraisons en retard :	26
18. Statistiques sur les retards :.....	27

1/ Gestion du catalogue produits et des fournisseurs

1. Ajout Fournisseur:



```
Sub Ajout_Fournisseur()  
Nouveau_Fournisseur.Show  
End Sub
```

Lorsque la macro Ajout_Fournisseur est activée en appuyant sur le l'icône d'ouvrière agricole, le sub Ajout_Fournisseur est appelé dans le module nommé : « GestionCatalogue ». La Userform nommée : « Nouveau_Fournisseur » est alors affichée, comme ci-dessous :

Créer un fournisseur X

Nom de la ferme	<input type="text"/>
Code postal	<input type="text"/>
<input type="button" value="Valider"/>	

Lorsque l'on rentre des données pour le code Postal, la fonction suivante est appelée :

```
Private Sub CODEPOST_FOURNISSEUR_IN_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)  
Select Case KeyAscii  
    Case Asc("0") To Asc("9")  
    Case Else  
        KeyAscii = 0  
    End Select  
End Sub
```

On vérifie ainsi grâce au code Ascii que les caractères rentrés sont des chiffres. Les autres types de caractères sont bloqués.

Lorsque le bouton « Valider » est activé, le code suivant est activé :

```

Private Sub CommandButton1_Click()
Dim nFournisseur As Integer
nFournisseur = Sheets("Fournisseur").Range("FOURNISSEUR").Rows.Count - 2
IdTest = Trim("S") + Trim(Str(Sheets("Fournisseur").Range("I2") + 1))
Sheets("Fournisseur").Range("I2") = Sheets("Fournisseur").Range("I2") + 1
Sheets("Fournisseur").Range("A" & nFournisseur + 3) = IdTest
Sheets("Fournisseur").Range("B" & nFournisseur + 3) = FERME_FOURNISSEUR_IN.Text
Sheets("Fournisseur").Range("C" & nFournisseur + 3) = CODEPOST_FOURNISSEUR_IN.Text
Sheets("Fournisseur").Range("A1:C" & nFournisseur + 3).Name = "FOURNISSEUR"
Nouveau_Fournisseur.Hide
End Sub

```

On récupère ici le nombre de lignes de la table « FOURNISSEUR », puis on écrit les informations du nouveau fournisseur en dessous du dernier fournisseur. L'ID_Fournisseur est créé automatiquement en prenant en compte les identifiants précédents. On utilise la fonction Trim pour enlever les espaces à la fin des chaînes de caractères afin de les concaténer sous la forme : « S1 ». Le tableau « FOURNISSEUR » est ensuite agrandi avec le fournisseur nouvellement créé.

2. Suppression Fournisseur:



```

Sub Suppression_Fournisseur()
Supprimer_Fournisseur.Liste_Fournisseur.RowSource = "Fournisseur! A3:A" & Range("FOURNISSEUR").Rows.Count
Supprimer_Fournisseur.Show
End Sub

```

Lorsque la macro Ajout_Fournisseur est activée en appuyant sur le l'icône de fermier, le sub Suppression_Fournisseur est appelé dans le module nommé : « GestionCatalogue ». La liste modifiable est remplie en parcourant la colonne A du tableau « FOURNISSEUR ». La Userform nommée : « Supprimer_Fournisseur » est alors affichée, comme ci-dessous :

Supprimer un fournisseur

Choisissez le fournisseur à supprimer

Modifier Supprimer

Lorsque le bouton « Supprimer » est activé, le code suivant est appelé :

```
Private Sub CommandButton1_Click()  
    Sheets("Fournisseur").Activate  
    nFournisseur = Range("FOURNISSEUR").Rows.Count  
    For i = 1 To nFournisseur  
        If Cells(i + 2, 1) = Liste_Fournisseur.Value Then  
            Cells(i + 2, 1).EntireRow.Delete  
            Exit For  
        End If  
    Next i  
    Supprimer_Fournisseur.Hide  
End Sub
```

On parcourt le tableau « FOURNISSEUR » et on compare l'identifiant du fournisseur avec celui sélectionné dans la liste. Lorsqu'on trouve un nom en commun, la liste entière du fournisseur est supprimée.

Lorsqu'on clique sur « Modifier », la UserForm « Edit_Fourni » est appelée.

ID du produit :	Données précédentes	Nouvelles données
S2		
Nom de la ferme	Catacombes	Catacombes
Code postal	75014	75014

Valider

Le bouton « Valider » renferme le paragraphe suivant :

```
Private Sub CommandButton1_Click()  
    If (Edit_Fourni.FERME_FOURNISSEUR_IN.Text <> "") Then  
        Sheets("Fournisseur").Activate  
        nfourni = Range("FOURNISSEUR").Rows.Count - 2  
        For i = 1 To nfourni  
            If Cells(i + 2, 1) = Edit_Fourni.Label20.Caption Then  
                Sheets("Fournisseur").Range("B" & i + 2) = FERME_FOURNISSEUR_IN.Text  
                Sheets("Fournisseur").Range("C" & i + 2) = CODEPOST_FOURNISSEUR_IN.Text  
            End If  
        Next i  
    Else: MsgBox ("Veuillez renseigner tous les champs!")  
    End If  
End Sub
```

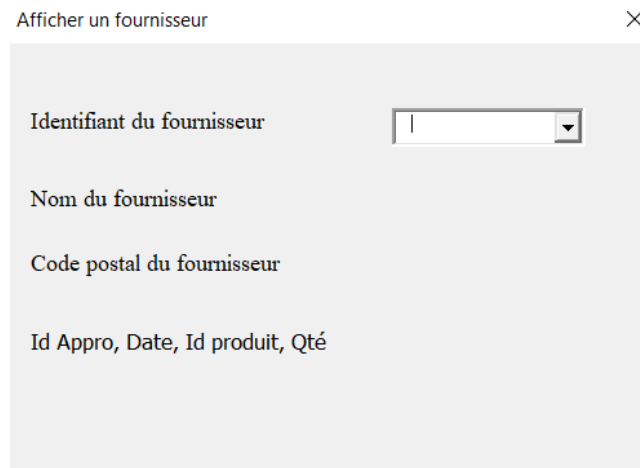
Ainsi, on vérifie que les cases soient remplies par l'utilisateur, puis on va remplacer les anciennes informations du fournisseur par les informations nouvellement rentrées.

3. Affichage Fournisseur :



```
Sub Affichage_Fournisseur()  
Afficher_Fournisseur.Liste_Fournisseur.RowSource = "Fournisseur! A3:A" & Range("FOURNISSEUR").Rows.Count  
Afficher_Fournisseur.Show  
End Sub
```

Lorsque la macro Ajout_Fournisseur est activée en appuyant sur le l'icône de fermier, le sub Suppression_Fournisseur est appelé dans le module nommé : « GestionCatalogue ». La liste modifiable est remplie en parcourant la colonne A du tableau « FOURNISSEUR ». La Userform nommée : « Afficher_Fournisseur » est alors affichée, comme ci-dessous :

The screenshot shows a UserForm titled "Afficher un fournisseur" with a close button (X) in the top right corner. The form has a light gray background and contains the following labels and controls:

- "Identifiant du fournisseur" followed by a text box with a dropdown arrow.
- "Nom du fournisseur"
- "Code postal du fournisseur"
- "Id Appro, Date, Id produit, Qté"

Dès que le contenu de la liste modifiable est changé, le programme suivant est parcouru :

```
Private Sub Liste_Fournisseur_Change()  
Sheets("Fournisseur").Activate  
nFournisseur = Range("FOURNISSEUR").Rows.Count  
For i = 1 To nFournisseur  
    If Cells(i + 2, 1) = Liste_Fournisseur.Value Then  
        TextBox_Nom_Fournisseur.Caption = Cells(i + 2, 2).Value  
        TextBox_CodePost_Fournisseur.Caption = Cells(i + 2, 3).Value  
        Affichage_Historique.Caption = ("Id Appro, Date, Id produit, Qté")  
        GetApprovisionnement (Liste_Fournisseur.Value)  
    End If  
Next i  
Affichage_Historique.Visible = True  
End Sub
```

On parcourt la fiche fournisseur et on compare l'identifiant demandé par l'utilisateur à ceux du tableau « FOURNISSEUR ». Lorsqu'on trouve l'identifiant demandé par l'utilisateur, on récupère toutes les autres

informations liées à ce fournisseur, notamment grâce à la fonction GetApprovisionnement qui récupère les données des approvisionnements associés au fournisseur.

4. Ajout Produit:



```
Sub Ajout_Produit()  
Range(Worksheets("Produit").Range("U3"), Worksheets("Produit").Range("U3").End(xlDown)).Clear  
Range("Produit!F3:F" & Range("PRODUIT").Rows.Count).AdvancedFilter Action:=xlFilterCopy, _  
CopyToRange:=Worksheets("Produit").Range("U3"), Unique:=True  
Nouveau_Produit.CATEGORIE_PRODUIT_INN.RowSource = "Produit! U3:U" & Sheets("Produit"). _  
[H65000].End(xlUp).Row  
Range(Worksheets("Produit").Range("V3"), Worksheets("Produit").Range("V3").End(xlDown)).Clear  
Range("Produit!G3:G" & Range("PRODUIT").Rows.Count).AdvancedFilter Action:=xlFilterCopy, _  
CopyToRange:=Worksheets("Produit").Range("V3"), Unique:=True  
Nouveau_Produit.SOUS_CATEGORIE_PRODUIT_INN.RowSource = "Produit! V3:V" & Sheets("Produit"). _  
[H65000].End(xlUp).Row  
Nouveau_Produit.ComboBox1.RowSource = "Fournisseur! A3:B" & Range("FOURNISSEUR").Rows.Count  
Nouveau_Produit.Show  
End Sub
```

Lorsque la macro est activée, le code ci-dessus est parcouru. On parcourt toutes les catégories et sous-catégories de produit existantes, puis on les trie, avant de les afficher dans des listes déroulantes.

On affiche ensuite la Userform suivante :

Créer un produit

Nom du produit

Description du produit

Prix de conditionnement

Prix au kilo

Catégorie du produit

Sous-Catégorie du produit

Stock du produit

Fournisseur

Marge

Date de péremption:

jjmmaaaa

Valider

Quand le bouton « Valider » est sélectionné, les lignes suivantes sont appliquées :

```

Private Sub CommandButton1_Click()
|
Dim Compteur As Integer
Dim nProduit As Integer
Dim nCatégorie As Integer
Dim nSousCatégorie As Integer

nProduit = Sheets("Produit").Range("PRODUIT").Rows.Count - 2
IdTest = Trim("P") + Trim(Str(Sheets("Produit").Range("X2") + 1))
Sheets("Produit").Range("X2") = Sheets("Produit").Range("X2") + 1
Sheets("Produit").Range("A" & nProduit + 3) = IdTest
Sheets("Produit").Range("B" & nProduit + 3) = NOM_PRODUIT_IN.Text
Sheets("Produit").Range("C" & nProduit + 3) = DESCRIPTION_PRODUIT_IN.Text
Sheets("Produit").Range("D" & nProduit + 3) = PRIXCONDI_PRODUIT_IN.Text
Sheets("Produit").Range("E" & nProduit + 3) = PRIXKILO_PRODUIT_IN.Text
Sheets("Produit").Range("F" & nProduit + 3) = CATEGORIE_PRODUIT_INN.Value
Sheets("Produit").Range("G" & nProduit + 3) = SOUS_CATEGORIE_PRODUIT_INN.Value
Sheets("Produit").Range("I" & nProduit + 3) = STOCK_PRODUIT_IN.Text
Sheets("Produit").Range("J" & nProduit + 3) = 10
Sheets("Produit").Range("H" & nProduit + 3) = ComboBox1.Column(0)
Sheets("Produit").Range("K" & nProduit + 3) = Marge_IN.Text
Sheets("Produit").Range("L" & nProduit + 3) = CDate(Date_IN.Value)
Sheets("Produit").Range("A1:L" & nProduit + 3).Name = "PRODUIT"

Nouveau_Produit.Hide
End Sub

```

On récupère ainsi toutes les informations entrées par l'utilisateur pour les inscrire dans les colonnes correspondantes dans la table « PRODUIT ». Certaines informations devant être rentrées sous format de nombres ou de dates, sont contrôlées par d'autres fonctions, comme celle-ci :

```

Private Sub PRIXCONDI_PRODUIT_IN_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
Select Case KeyAscii
    Case Asc("0") To Asc("9")
    Case Asc(".")
        If InStr(1, Me.PRIXCONDI_PRODUIT_IN.Text, ".") > 0 Then
            KeyAscii = 0
        End If
    Case Else
        KeyAscii = 0
End Select
End Sub

```

Lorsque une information est rentrée dans une textbox par le biais d'une touche de clavier, la fonction précédente est appelée et vérifie que les seuls caractères rentrables pour le prix de conditionnement sont des chiffres et un point (faisant office de virgule).

5. Suppression/Édition Produit:



Après activation de la macro, comme pour les macros précédentes, la liste déroulante est remplie avec les identifiants de produits existant. L'image suivante apparaît à l'écran :

Supprimer/ éditer un produit X

Choisissez le produit à éditer

Modifier Supprimer

On a alors deux choix qui s'offrent à nous :

- On peut cliquer sur « Supprimer » et le sub ci-dessous est alors parcouru :

```
Private Sub CommandButton1_Click()
    Sheets("Produit").Activate
    nProduit = Range("PRODUIT").Rows.Count
    For i = 1 To nProduit
        If Cells(i + 2, 1) = Liste_Produit.Value Then
            For k = 0 To Range("PRODUIT").Rows.Count - 1
                Cells(i + 2, Range("PRODUIT").Rows.Count - k).Delete
            Next k
        End If
    Next i
    Sheets("Produit").Range("A1:L" & nProduit - 1).Name = "PRODUIT"
    Supprimer_Produit.Hide
End Sub
```

Ici, après avoir retrouvé le produit sélectionné par l'utilisateur grâce à une boucle for, on supprime entièrement la ligne correspondant au produit avec un Delete. Puis, on enlève une ligne de la table « PRODUIT » en la redimensionnant.

- On peut aussi cliquer sur « Modifier ». Dans ce cas, la fiche d'édition de produit suivante est affichée (avec un exemple très concret, le fameux SPAM):

Modifier Produit X

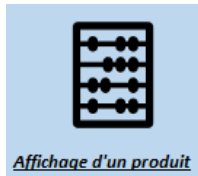
ID du produit :	P13	Données précédentes	Nouvelles données
Nom	SPAM tomato	<input type="text"/>	
Description	Vikings	<input type="text"/>	
Prix de conditionnement	5	<input type="text"/>	
Prix au kilo	3,66	<input type="text"/>	
Catégorie	SPAM	<input type="text" value="SPAM"/>	
Sous-Catégorie	NoComment	<input type="text" value="NoComment"/>	
Stock	0	<input type="text"/>	
Fournisseur	L'Elysée	<input type="text"/>	
Marge	3	<input type="text"/>	
<input type="button" value="Valider"/>			

Lorsque le bouton « Valider » est activé, le sub ci-dessous s'applique :

```
Private Sub CommandButton1_Click()
|   If Not Edit_Prod.ComboBox1.ListIndex = -1 Then
      Sheets("Produit").Activate
      nProduit = Range("PRODUIT").Rows.Count - 2
      For i = 1 To nProduit
          If Cells(i + 2, 1) = Edit_Prod.Label20.Caption Then
              Sheets("Produit").Range("B" & i + 2) = NOM_PRODUIT_IN.Text
              Sheets("Produit").Range("C" & i + 2) = DESCRIPTION_PRODUIT_IN.Text
              Sheets("Produit").Range("D" & i + 2) = PRIXCONDI_PRODUIT_IN.Text
              Sheets("Produit").Range("E" & i + 2) = PRIXKILO_PRODUIT_IN.Text
              Sheets("Produit").Range("F" & i + 2) = CATEGORIE_PRODUIT_INN.Value
              Sheets("Produit").Range("G" & i + 2) = SOUS_CATEGORIE_PRODUIT_INN.Value
              Sheets("Produit").Range("I" & i + 2) = STOCK_PRODUIT_IN.Text
              Sheets("Produit").Range("H" & i + 2) = ComboBox1.Column(0)
              Sheets("Produit").Range("K" & i + 2) = TextBox1.Text
          End If
      Next i
      Else: MsgBox ("Veuillez renseigner le fournisseur du produit!")
      End If
End Sub
```

On récupère l'identifiant du produit demandé par l'utilisateur et on le compare avec les produits de la table « PRODUIT ». Quand on trouve un produit qui correspond, on affecte les données rentrées dans les autres textbox aux colonnes de la table « PRODUIT ». Certaines données doivent être obligatoirement rentrées sous forme de chiffres, comme dans la macro « Ajout d'un produit ».

6. Affichage Produit:



Quand la macro est utilisée, la fenêtre suivante apparaît :

Afficher un produit

Catégorie de produit: Fruits et légumes

Sous catégorie de produit: Fruits de saison

Nom du produit:

Comme dans « Ajout d'un produit », les catégories et sous-catégories sont triées avant d'être positionnées dans les combobox. Lorsqu'on sélectionne une sous-catégorie dans la combobox de droite, le code suivant s'exécute :

```

Private Sub Liste_Nom_Produit_Change()
    Sheets("Produit").Activate
    nProduit = Range("PRODUIT").Rows.Count
    For i = 1 To nProduit
        If Cells(i + 2, 2) = Liste_Nom_Produit.Value Then
            ID = Cells(i + 2, 1).Value
            Label4.Visible = True
            Label_Description_Produit.Visible = True
            Label5.Visible = True
            TextBox_Prix_Conditionnement.Visible = True
            TextBox_Prix_Kilo.Visible = True
            Label6.Visible = True
            Label7.Visible = True
            TextBox_Nom_Fournisseur.Visible = True
            Label8.Visible = True
            TextBox_Stock.Visible = True
            TextBox1.Visible = True
            TextBox2.Visible = True
            Exit For
        End If
    Next i
    FonctionAfficheProduit (ID)
End Sub

Private Sub Liste_Sous_Categorie_Change()
    Sheets("Produit").Activate
    nProduit = Range("PRODUIT").Rows.Count
    Afficher_Produit.Liste_Nom_Produit.Clear
    For i = 1 To nProduit
        If Cells(i + 2, 6) = Liste_Categorie.Value Then
            If Cells(i + 2, 7) = Liste_Sous_Categorie.Value Then
                Afficher_Produit.Liste_Nom_Produit.AddItem Cells(i + 2, 2)
                Label3.Visible = True
                Liste_Nom_Produit.Visible = True
            End If
        End If
    Next i
End Sub

```

On ajoute ainsi dans la liste déroulante du bas, tous les produits appartenant à cette sous-catégorie. La combobox suivante est aussi affichée. Puis, lorsqu'on sélectionne un produit dans la nouvelle liste déroulante, les textbox et labels sont tous affichés, puis remplis avec la fonction « FonctionAfficheProduit », qui prend l'identifiant du produit en entrée en tant que String.

```

Private ID As String
Private Sub FonctionAfficheProduit(ID As String)
    Sheets("Produit").Activate
    nProduit = Range("PRODUIT").Rows.Count - 2
    For i = 1 To nProduit
        If ID = Cells(i + 2, 1).Value Then
            Label_Description_Produit.Caption = Cells(i + 2, 3).Value
            TextBox_Prix_Conditionnement.Value = Cells(i + 2, 4).Value
            TextBox_Prix_Kilo.Value = Cells(i + 2, 5).Value
            TextBox_Stock.Value = Cells(i + 2, 9).Value
            IdFourni = Cells(i + 2, 8).Value
            TextBox1.Value = Cells(i + 2, 11).Value
            TextBox2.Value = Cells(i + 2, 12).Value
            GetFourniName (IdFourni)
        End If
    Next i
End Sub

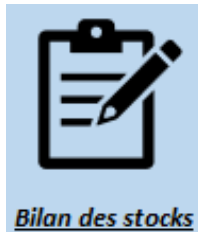
Private Sub GetFourniName(IdFournisseur As String)
    Sheets("Fournisseur").Activate
    nFournisseur = Range("FOURNISSEUR").Rows.Count
    For k = 1 To nFournisseur
        If Cells(k + 2, 1).Value = IdFournisseur Then
            TextBox_Nom_Fournisseur.Value = Cells(k + 2, 2).Value
            Exit For
        End If
    Next k
End Sub

```

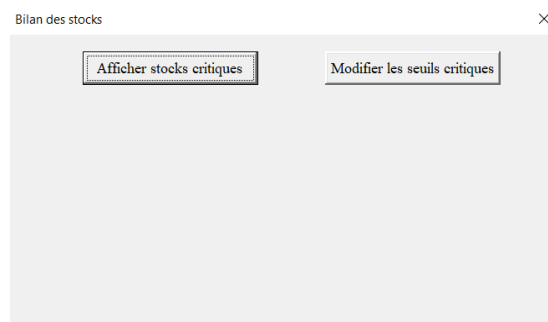
La fonction « FonctionAfficheProduit » assigne à chaque textbox l'information correspondant à l'identifiant produit donné en argument de la fonction. Puis la fonction « GetFourniName » est utilisée avec l'identifiant du fournisseur en argument.

En parcourant la table « FOURNISSEUR », cette fonction permet d'assigner à une textbox le nom du fournisseur du produit demandé.

7. Bilan Stocks :



Après activation de la macro, la Userform suivante se matérialise :



Le bouton « Afficher les stocks critiques » est relié au code suivant :

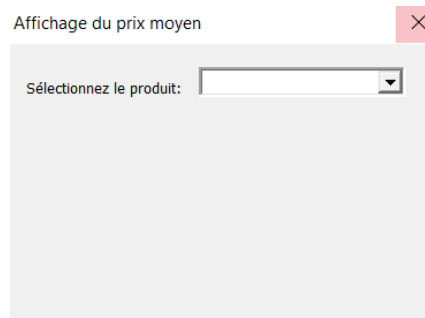
```
Sheets("Produit").Activate
nProduit = Range("PRODUIT").Rows.Count
Label2.Visible = True
Affichage_Critique.Visible = True
Bilan_Stocks.Affichage_Critique.Caption = ("Produit / Stock")
For i = 1 To nProduit
    If Cells(i + 2, 9) < Cells(i + 2, 10) Then
        Bilan_Stocks.Affichage_Critique.Caption = (Bilan_Stocks.Affichage_Critique.Caption _
            & Chr(13) & Chr(10) & Cells(i + 2, 2).Value & " / " & Cells(i + 2, 9).Value)
    End If
Next i
End Sub
```

Dans ce programme, on parcourt toute la liste des produits et on compare à chaque fois le stock avec le stock critique. Quand on trouve un stock trop faible, on rentre dans le if et on affiche à la ligne le nom du produit et son stock, grâce aux caractères Chr(13) & Chr(10).

8. Prix Moyen :



Après avoir cliqué sur la macro, la fiche suivante apparaît :



Quand l'utilisateur sélectionne un produit en cliquant dans la liste modifiable, le sub ci-dessous est parcouru :

```
Private Sub Liste_Produits_PM_Change()  
    Dim S As Double  
    Dim N As Double  
    N = 0  
    S = 0  
    Sheets("Produit").Activate  
    nProduit = Range("PRODUIT").Rows.Count  
    For i = 1 To nProduit  
        If Cells(i + 2, 2) = Liste_Produits_PM.Value Then  
            S = S + Cells(i + 2, 4).Value  
            N = N + 1  
        End If  
    Next i  
    Label2.Caption = ("Prix moyen du produit: " & S / N)  
End Sub
```

On parcourt les produits dans la table « PRODUIT ». On fait alors la somme des prix du produit demandé en comptant le nombre de fois où le produit apparaît. Puis on fait la moyenne en divisant le prix total par le nombre de produits. On affiche ensuite le prix moyen dans un label.

2/ Gestion des commandes clients

9. Afficher Consommateur:



Après activation de la macro, la fenêtre suivante est montrée à l'utilisateur.



```
Sub Affichage_Consoommateur()  
Afficher_Consoommateur.ComboBox1.AddItem "Tous"  
For i = 65 To 90  
    Afficher_Consoommateur.ComboBox1.AddItem Chr(i)  
Next i  
Afficher_Consoommateur.Show  
End Sub
```

Lorsque l'utilisateur utilise cette macro, le code précédent est appelé. Il permet d'afficher les lettres de l'alphabet dans une combobox, leur charcode allant de 65 à 90.

La première combobox permet à l'utilisateur de définir la première lettre du nom du client. Ainsi, la liste est reliée au code ci-dessous :

```

Private Sub ComboBox1_Change()
Liste_Nom_Consoommateur_IN.Clear
If ComboBox1.Value = "Tous" Then
    For Each Cellule In Range("CONSOMMATEUR").Columns(2).Cells
        If Cellule.Value2 <> "NOM_CONSOMMATEUR" Then
            Liste_Nom_Consoommateur_IN.AddItem Cellule.Value
        End If
    Next Cellule
ElseIf ComboBox1.ListIndex <> 0 Then
    For Each Cellule In Range("CONSOMMATEUR").Columns(2).Cells
        If Left(Cellule, 1) = ComboBox1.Value Then
            Liste_Nom_Consoommateur_IN.AddItem Cellule.Value
        End If
    Next Cellule
End If
End Sub

```

Si la valeur « Tous » est sélectionnée, alors on en allant dans la table « CONSOMMATEUR », on affichera dans la deuxième liste déroulante le nom de tous les consommateurs.

Sinon, lorsqu'une lettre est sélectionnée, on parcourt aussi la liste des consommateurs et on vérifie que la première lettre du nom correspond à la lettre choisie par l'utilisateur, grâce à la formule : Left(Cellule,1).

L'utilisateur peut alors choisir le consommateur sur lequel il veut se renseigner dans la liste déroulante. Une fois que c'est fait, le code suivant est appelé :

```

Private Sub Liste_Nom_Consoommateur_IN_Change()
Affichage_Commande.Caption = ("Statut Commande/ Horodatage Facture / Nombre de produits achetés")
Liste_Commande.Clear

Sheets("Consoommateur").Activate
nConsoommateur = Range("CONSOMMATEUR").Rows.Count - 2

Dim ID_Consoommateur As String

For i = 1 To nConsoommateur
    If Cells(i + 2, 2) = Liste_Nom_Consoommateur_IN Then
        Prenom_Consoommateur.Value = Cells(i + 2, 3)
        Points_Consoommateur.Value = Cells(i + 2, 4)
        ID_Consoommateur = Cells(i + 2, 1)
    End If
Next i

Sheets("Commande").Activate
nCommande = Range("COMMANDE").Rows.Count
For i = 1 To nCommande
    If Cells(i + 2, 4) = ID_Consoommateur Then
        Liste_Commande.AddItem (Cells(i + 2, 3))
    End If
Next i

Label2.Visible = True
Label3.Visible = True
Label4.Visible = True
Prenom_Consoommateur.Visible = True
Points_Consoommateur.Visible = True
Liste_Commande.Visible = True

End Sub

```

On parcourt la table « CONSOMMATEUR » pour obtenir les informations correspondant au consommateur recherché. On enregistre alors son identifiant dans la variable « ID_Consoommateur », car c'est la clef étrangère reliant cette table à la table « COMMANDE ». Une fois dans la table commande, on recherche les commandes appartenant à ce client et on les affiche dans une liste déroulante.



Quand l'utilisateur modifie le contenu de la combobox ci-dessus, le code suivant s'applique :

```
Private Sub Liste_Commande_Change()
Affichage_Commande.Visible = True
Sheets("Commande").Activate
nCommande = Range("COMMANDE").Rows.Count - 2

Dim Id_Facture As String
Dim Statut_Commande As String
Dim Horodatage As String
Dim Nom_Produit As String
Dim Id_Produit As String

For i = 1 To nCommande
    If Range("C" & i + 2).Text = Liste_Commande.Value Then
        Id_Facture = Cells(i + 2, 5)
        Statut_Commande = Cells(i + 2, 2)
        Affichage_Commande.Caption = ("Statut Commande/ Horodatage Facture / Nombre de produits achetés")

        Sheets("Facture").Activate
        nFacture = Range("FACTURE").Rows.Count - 2

        For j = 1 To nFacture
            If Cells(j + 2, 1).Value = Id_Facture Then
                Horodatage = Cells(j + 2, 2)

                Sheets("Ligne Facture").Activate
                nLigneFacture = Range("LIGNEFACTURE").Rows.Count - 2

                For k = 1 To nLigneFacture
                    If Sheets("Ligne Facture").Range("B" & k + 2).Text = Id_Facture Then
                        Sheets("Ligne Facture").Activate
                        nLigneFacture = Range("LIGNEFACTURE").Rows.Count - 2

                        Id_Produit = Cells(k + 2, 3).Value
                        Sheets("Produit").Activate
                        nProduit = Range("PRODUIT").Rows.Count - 2

                        For l = 1 To nProduit
                            If Range("A" & l + 2).Text = Id_Produit Then
                                Nom_Produit = Cells(l + 2, 2)
                            End If
                        Next l

                        Affichage_Commande.Caption = (Affichage_Commande.Caption & Chr(13) & Chr(10) & Chr(13) & Chr(10) & Statut_Commande & " / " & Horodatage & " / " & Sheets("Ligne Facture").Range("D" & k + 2).Text & " * " & Nom_Produit)
                    End If
                Next k
            End If
        Next j
    End If
Next i

End Sub
```

On parcourt ici la fiche commande pour récupérer toutes les informations de la commande nécessaires à afficher, ainsi que l'ID_Facture associé à la commande. En effet, l'ID_Facture est une clef étrangère de la table « FACTURE ». On parcourt donc ensuite la ligne de facture correspondante, grâce à l'ID_Facture , et

on y récupère le nombre de produits vendus ainsi que leur identifiant. Enfin, on récupère les noms des produits dans la table « PRODUIT ». On affiche ensuite toutes ces informations dans un label à chaque itération de boucle, en faisant bien attention d'ajouter les dernières informations aux précédentes.

10. Liste Commande selon Statut:



Après avoir appuyé sur le bouton de la macro, la fiche suivante apparaît :

Commandes par statut X

Statut :

☐ En préparation ☐ En livraison ☐ Livraison effectuée

En fonction de la checkbox sélectionnée, on appelle la fonction « commande_selon_statut » avec un argument différent.

```
Private Sub CheckBox1_Click()
If CheckBox1.Value = True Then
    commande_selon_statut ("En préparation")
    CheckBox1.Enabled = False
End If
End Sub

Private Sub CheckBox2_Click()
If CheckBox2.Value = True Then
    commande_selon_statut ("En livraison")
    CheckBox2.Enabled = False
End If
End Sub

Private Sub CheckBox3_Click()

If CheckBox3.Value = True Then
    commande_selon_statut ("Livraison effectuée")
    CheckBox3.Enabled = False
End If
End Sub
```

Ensuite, la fonction ci-après permet d'afficher toutes les informations d'une commande dans les 4 textbox de la fenêtre. On récupère ainsi les informations d'une ligne de la table « COMMANDE », dès qu'on trouve une commande avec le même statut que celui voulu par l'utilisateur. On parcourt aussi la table « CONSOMMATEUR » afin d'obtenir le nom du consommateur ayant effectué la commande.

```

Private Sub commande_selon_statut(Status)
    Sheets("Commande").Activate
    For i = 1 To Range("COMMANDE").Rows.Count - 2
        Sheets("Commande").Activate
        If Cells(i + 2, 2).Text = Status Then
            TextBox1.Text = TextBox1.Text & Cells(i + 2, 1) & Chr(13) & Chr(10)
            TextBox3.Text = TextBox3.Text & Cells(i + 2, 3) & Chr(13) & Chr(10)
            TextBox4.Text = TextBox4.Text & Status & Chr(13) & Chr(10)
            IdClient = Cells(i + 2, 4)
            Sheets("Consommateur").Activate
            For j = 1 To Range("CONSOMMATEUR").Rows.Count - 2
                If Cells(j + 2, 1).Text = IdClient Then
                    TextBox2.Text = TextBox2.Text & Cells(j + 2, 2) & " " & Cells(j + 2, 3) & Chr(13) & Chr(10)
                Exit For
            End If
        Next j
    End If
Next i
End Sub

```

Enfin, le bouton « Vider » permet de réinitialiser l'afficher des labels, décocher les checkbox et les remettre à disposition pour l'utilisateur.

11. Liste Commande avec Livraison Offerte :



Lorsque l'utilisateur utilise la macro, le code suivant s'exécute :

```

Sub Livraison_Offerte()
    Afficher_Livraison_Offerte.Label_Livraison_Offerte.Caption = _
    |("N° Commande/ Client / Date de la commande / Prix de la commande")

    Dim Id_Client As String
    Dim Id_Commande As String
    Dim Date_Commande As String
    Dim Id_Facture As String
    Dim nb_Produit As Integer
    Dim Id_Produit As String
    Dim Prix_Condi As String
    Dim Prix_Conditionnement As Long
    Dim Prix_Ligne As Integer
    Dim Prix_Commande As Integer
    Dim Nom_Client As String
    Dim var As Integer

    Sheets("Commande").Activate
    nCommande = Range("COMMANDE").Rows.Count - 2
    For i = 1 To nCommande
        Prix_Commande = 0
        Id_Client = Sheets("Commande").Range("D" & i + 2).Text
        Id_Commande = Sheets("Commande").Range("A" & i + 2).Text
        Date_Commande = Sheets("Commande").Range("C" & i + 2).Text
        Id_Facture = Sheets("Commande").Range("E" & i + 2).Text

        Sheets("Ligne Facture").Activate
        nLigne_Facture = Range("LIGNEFACTURE").Rows.Count - 2
        For j = 1 To nLigne_Facture
            If Sheets("Ligne Facture").Range("B" & j + 2).Text = Id_Facture Then
                nb_Produit = CInt(Sheets("Ligne Facture").Range("D" & j + 2).Text)
                Id_Produit = Sheets("Ligne Facture").Range("C" & j + 2).Text

                Sheets("Produit").Activate
                nProduit = Range("PRODUIT").Rows.Count - 2
                For k = 1 To nProduit
                    If Not IsEmpty(Id_Produit) And Sheets("Produit").Range("A" & k + 2).Text = Id_Produit Then
                        Prix_Condi = Sheets("Produit").Range("D" & k + 2).Text
                        Prix_Conditionnement = CInt(Prix_Condi)
                        Prix_Ligne = Prix_Conditionnement * nb_Produit
                    End If
                Next k
            End If
        Next j
    Next i
End Sub

```

```

        Prix_Commande = Prix_Commande + Prix_Ligne
    End If
Next j

Sheets("Consommateur").Activate
nClient = Range("CONSOMMATEUR").Rows.Count - 2
var = 0
For m = 1 To nClient
    If Id_Client = Sheets("Consommateur").Range("A" & m + 2).Text Then
        Nom_Client = Sheets("Consommateur").Range("B" & m + 2).Text
        If Sheets("Consommateur").Range("E" & m + 2).Text = 0 Then
            If Prix_Commande >= 50 Then
                var = 1
            End If
        ElseIf Prix_Commande >= 30 Then
            var = 2
        End If
    End If
Next m

If var = 1 Or var = 2 Then
    Afficher_Livraison_Offerte.Label_Livraison_Offerte.Caption = _
    (Afficher_Livraison_Offerte.Label_Livraison_Offerte.Caption & Chr(13) & Chr(10) & Chr(13) & Chr(10) & _
    & Id_Commande & " / " & Nom_Client & " / " & Date_Commande & " / " & Prix_Commande & "€")
End If

Next i

Afficher_Livraison_Offerte.Show
End Sub

```

Dans le code précédent, on récupère toutes les informations devant être affichées à partir de plusieurs tables. On calcule aussi le prix total de la commande en passant tous les chiffres en Int pour le calcul, grâce à « CInt() ». Puis, on vérifie si l'utilisateur a un compte fidélité ou non. Ainsi, dans la colonne E de la table « CONSOMMATEUR », si la valeur est de 1, alors l'utilisateur a un compte fidèle, si elle est de 0, alors il n'en a pas.

Puis, pour un client fidélisé, on vérifie que son total de commande dépasse 30€ pour afficher sa livraison. De même pour un client non fidélisé mais pour un montant de 50€.

Puis, on affiche le résultat final comme-suit :

N° Commande/ Client / Date de la commande / Prix de la commande
O1 / EPF / 01/01/1970 / 152€
O3 / EPF / 03/01/1971 / 170€
O5 / Starfire / 05/01/1970 / 41€
O6 / EPF / 06/01/1970 / 99€
O8 / Arnaudal / 08/01/1970 / 144€

12. Qualité des livraisons :



En activant la macro, les lignes de code suivantes s'exécutent :

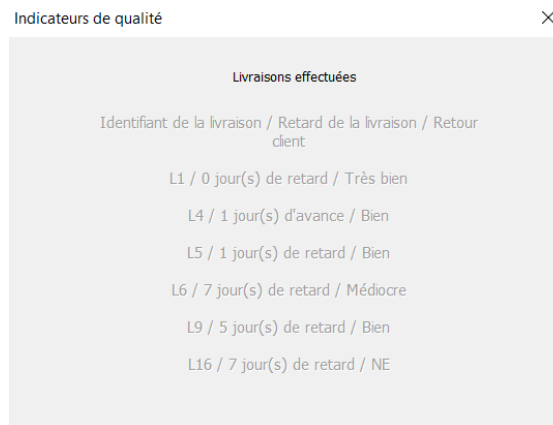
```
Sub Indicateur_Qualite_Livraison()
    Sheets("Commande").Activate
    nCommande = Range("COMMANDE").Rows.Count - 2
    Indicateur_Qualité.TextBox1.Text = ("Identifiant de la livraison / Retard de la livraison / Retour client")

    Dim Id_Commande As String
    Dim Retard As Long
    Dim Qualité As String
    Dim Id_Livraison As String

    For i = 1 To nCommande
        If Sheets("Commande").Range("B" & i + 2).Text = "Livraison effectuée" Then
            Id_Commande = Sheets("Commande").Range("A" & i + 2).Text

            Sheets("Livraison").Activate
            nLivraison = Range("LIVRAISON").Rows.Count - 2
            For j = 1 To nLivraison
                If Sheets("Livraison").Range("G" & j + 2).Text = Id_Commande Then
                    Retard = CLng(Cells(j + 2, 3)) - CLng(Cells(j + 2, 2))
                    Qualité = Sheets("Livraison").Range("E" & j + 2).Text
                    Id_Livraison = Sheets("Livraison").Range("A" & j + 2).Text
                    If Retard >= 0 Then
                        Indicateur_Qualité.TextBox1.Text = (Indicateur_Qualité.TextBox1.Text & Chr(13) & Chr(10) & Chr(13) & Chr(10) & Id_Livraison & " / " & Retard & " jour(s) de retard / " & Qualité)
                    ElseIf Retard < 0 Then
                        Indicateur_Qualité.TextBox1.Text = (Indicateur_Qualité.TextBox1.Text & Chr(13) & Chr(10) & Chr(13) & Chr(10) & Id_Livraison & " / " & -Retard & " jour(s) d'avance / " & Qualité)
                    End If
                End If
            Next j
        End If
    Next i
    Indicateur_Qualité.Show
End Sub
```

On affiche ici les retours des clients pour les livraisons effectuées. On vérifie alors tout d'abord le statut des livraisons. Puis on calcule le retard de la livraison en faisant la différence entre la date d'arrivée et la date d'arrivée prévue. On utilise donc CLng pour faire la différence et avoir un résultat sous le format Long. Puis on affiche les informations de la livraison avec deux cas différents, afin de pouvoir préciser si la livraison était en retard ou en avance. Voici ci-dessous, l'affichage final.



13. Qualité des commandes :



Après activation de la macro, le sub suivant est appelé :

```
Sub Indicateur_Qualite_Commande()
Indicateur_Commande.Label2.Caption = ("Identifiant de la commande/ Statut Commande/ Produits périmés / Retour client")
Sheets("Commande").Activate
nCommande = Range("COMMANDE").Rows.Count - 2

Dim Id_Commande As String
Dim Statut_Commande As String
Dim Id_Facture As String
Dim Id_Produit As String
Dim Date_Peremption As Date
Dim Nb_Perimés As Integer
Dim Retour_Client As String

For i = 1 To nCommande
    Id_Commande = Sheets("Commande").Range("A" & i + 2).Text
    Statut_Commande = Sheets("Commande").Range("B" & i + 2).Text
    Id_Facture = Sheets("Commande").Range("E" & i + 2).Text

    Sheets("Ligne Facture").Activate
    nLigneFacture = Range("LIGNEFACTURE").Rows.Count - 2
    Nb_Perimés = 0
    For j = 1 To nLigneFacture
        If Sheets("Ligne Facture").Range("B" & j + 2).Text = Id_Facture Then
            Id_Produit = Sheets("Ligne Facture").Range("C" & j + 2).Text

            Sheets("Produit").Activate
            nProduit = Range("PRODUIT").Rows.Count - 2
            For k = 1 To nProduit
                If Sheets("Produit").Range("A" & k + 2).Text = Id_Produit Then
                    Date_Peremption = CDate(Sheets("Produit").Range("I" & k + 2).Text)

                    Sheets("Livraison").Activate
                    nLivraison = Range("LIVRAISON").Rows.Count - 2
                    For m = 1 To nLivraison
                        If Sheets("Livraison").Range("G" & m + 2).Text = Id_Commande Then
                            If CDate(Sheets("Livraison").Range("C" & m + 2).Text) > Date_Peremption Then
                                Nb_Perimés = Nb_Perimés + 1
                            End If
                            Retour_Client = Sheets("Livraison").Range("E" & m + 2).Text
                        End If
                    Next m
                End If
            Next k
        End If
    Next j
    Indicateur_Commande.Label2.Caption = (Indicateur_Commande.Label2.Caption & Chr(13) & Chr(10) & Chr(13) & Chr(10) & Id_Commande & " / " & Statut_Commande & " / " & Nb_Perimés & " produit(s) périmé(s) / " & Retour_Client)
Next i

Indicateur_Commande.Show
End Sub
```

Dans cette macro, comme dans les macros précédentes, on parcourt différentes tables pour récupérer différentes informations à afficher, comme le statut de la commande, l'identifiant de la commande, etc... On récupère aussi la date de péremption qu'on transforme en format date grâce à CDate(), puis on

compare la date de péremption à la date de livraison, afin de définir si le produit était périmé ou non à la livraison. Voici ci-après l’affichage de la fenêtre :

Indication de la qualité de la commande ×

Identifiant de la commande/ Statut Commande/ Produits périmés / Retour client

O1 / Livraison effectuée / 0 produit(s) périmé(s) / Très bien

O2 / En livraison / 0 produit(s) périmé(s) / NE

O3 / En livraison / 0 produit(s) périmé(s) / NE

O4 / Livraison effectuée / 0 produit(s) périmé(s) / Bien

O5 / Livraison effectuée / 2 produit(s) périmé(s) / Bien

O6 / Livraison effectuée / 1 produit(s) périmé(s) / Médiocre

O7 / En préparation / 1 produit(s) périmé(s) / NE

O8 / En préparation / 2 produit(s) périmé(s) / NE

O9 / Livraison effectuée / 1 produit(s) périmé(s) / Bien

O10 / En livraison / 1 produit(s) périmé(s) / NE

3/ Gestion commerciale

14. Bilan par catégorie de produits :



Lorsque la macro est activée, les différentes catégories de produit sont affichées dans la combobox de l’image ci-dessous. Les différentes catégories sont obtenues de la même manière que pour la macro « Ajout d’un Produit » avec un tri.

Affichage du bilan par catégorie de produit ×

Listes des catégories

Après avoir choisi une catégorie, le programme suivant est exécuté :

```

Private Sub Liste_Cat_Change()
Sheets("Produit").Activate
nProduit = Range("PRODUIT").Rows.Count - 2

Label_Bilan_Cat.Caption = ("Bilan des ventes pour la catégorie sélectionnée:")

Dim Prix_Condi As Integer
Dim Id_Produit As String
Dim Prix_Tot As Integer

Prix_Tot = 0
For i = 1 To nProduit
    If Cells(i + 2, 6) = Liste_Cat.Value Then
        Prix_Condi = Cells(i + 2, 4)
        Id_Produit = Cells(i + 2, 1)

        Sheets("Ligne Facture").Activate
        nLigneFacture = Range("LIGNEFACTURE").Rows.Count - 2
        For j = 1 To nLigneFacture
            If Sheets("Ligne Facture").Range("C" & j + 2).Text = Id_Produit Then
                Prix_Tot = Prix_Tot + CInt(Sheets("Ligne Facture").Range("D" & j + 2).Text) * Prix_Condi
            End If
        Next j
    End If
Next i
Label_Bilan_Cat.Visible = True
Label_Bilan_Cat.Caption = (Label_Bilan_Cat.Caption & " " & Prix_Tot & "€")
End Sub

```

Dans la table « PRODUIT », on parcourt les produits appartenant à la catégorie demandée. On récupère alors leur prix de conditionnement et on additionne ce prix au prix total à chaque itération. On utilise ici aussi CInt() pour pouvoir faire un calcul avec des integer. On affiche finalement le prix total dans un label.

15. Bilan des ventes annuelles :



```

Sub Affichage_Bilan()
Year (Now())
For i = 1970 To Year(Now())
    Bilan_annu.ComboBox1.AddItem i
Next i
Bilan_annu.Show
End Sub

```

Bilan Annuel
×

Année :

▼

Quand la macro est appelée, le sub ci-dessus est appelé dans le module « GestionCommerciale ». On ajoute grâce à une boucle for et à la fonction Year(Now()), les années dans la liste déroulante ci-dessus :

```

Private Sub ComboBox1_Change()
If ComboBox1.ListIndex <> -1 Then
    Dim ListeFacture As ArrayList
    Set ListeFacture = New ArrayList
    Sheets("Facture").Activate
    For i = 1 To Range("FACTURE").Rows.Count - 2
        Label2.Caption = Year(Cells(i + 2, 2))
        If Label2.Caption = ComboBox1.Value Then
            ListeFacture.Add Cells(i + 2, 1).Text
        End If
    Next i
    Label2.Caption = ""
    Total = 0
    Total2 = 0
    Sheets("Ligne Facture").Activate
    Dim ListeProd As ArrayList
    Set ListeProd = New ArrayList
    For j = 1 To Range("LIGNEFACTURE").Rows.Count - 2
        a = ListeFacture.Contains(Cells(j + 2, 2).Text)

        If a <> False Then
            For k = 1 To Cells(j + 2, 4)
                ListeProd.Add Cells(j + 2, 3)
            Next k
        End If
    Next j
    Sheets("Produit").Activate
    For l = 1 To Range("PRODUIT").Rows.Count - 2
        For Each Prod In ListeProd
            If Prod.Text = Cells(l + 2, 1) Then
                Total = Total + Cells(l + 2, 11)
                Total2 = Total2 + Cells(l + 2, 4)
            End If
        Next Prod
    Next l
    Label2.Caption = "Le bilan de l'année " & ComboBox1.Value & " est de :" & Chr(10) & Chr(13) & Total _
    & " € de marge commerciale" & Chr(10) & Chr(13) & Total2 & " € de chiffre d'affaire."

    Set ListeFacture = New ArrayList
    Sheets("Facture").Activate
    For i = 1 To Range("FACTURE").Rows.Count - 2
        Label3.Caption = Year(Cells(i + 2, 2))
        If Label3.Caption = ComboBox1.Value - 1 Then
            ListeFacture.Add Cells(i + 2, 1).Text
        End If
    Next i
    Label3.Caption = ""
    Total = 0
    Total2 = 0
    Sheets("Ligne Facture").Activate
    Set ListeProd = New ArrayList
    For j = 1 To Range("LIGNEFACTURE").Rows.Count - 2
        a = ListeFacture.Contains(Cells(j + 2, 2).Text)

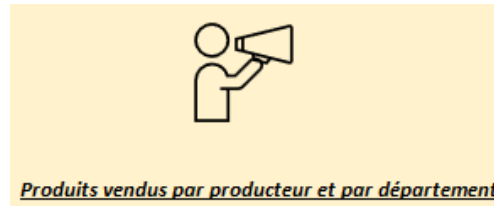
        If a <> False Then
            For k = 1 To Cells(j + 2, 4)
                ListeProd.Add Cells(j + 2, 3)
            Next k
        End If
    Next j
    Sheets("Produit").Activate
    For l = 1 To Range("PRODUIT").Rows.Count - 2
        For Each Prod In ListeProd
            If Prod.Text = Cells(l + 2, 1) Then
                Total = Total + Cells(l + 2, 11)
                Total2 = Total2 + Cells(l + 2, 4)
            End If
        Next Prod
    Next l
    Label3.Caption = "Le bilan de l'année " & ComboBox1.Value - 1 & " est de :" & Chr(10) & Chr(13) _
    & Total & " € de marge commerciale" & Chr(10) & Chr(13) & Total2 & " € de chiffre d'affaire."
End If
End Sub

```


Ce code est décomposé en deux parties similaires pour l'année choisie et l'année précédente.

On crée ainsi des variables sous format ArrayList pour stocker des données, notamment les identifiants de facture puis de produits. On parcourt ensuite la table « PRODUIT » pour additionner les valeurs de marge pour tous les produits. Idem pour les prix de conditionnement. On affiche ensuite les totaux dans les labels.

16. Produits vendus par producteur et par département :



Après activation de la macro, la fenêtre suivante apparaît :

Ventes de produits par département ×

Entrez le numéro de département:

```
Private Sub CommandButton1_Click()

Dim Chiffres_Dep As String
Dim Id_Fournisseur As String
Dim Id_Produit As String
Dim Nombre_Vendu As Integer

Sheets("Fournisseur").Activate
Label_Affichage_Ventes.Caption = ("Département / Identifiant du fournisseur / Nombre de produits vendus")

For i = 1 To Range("FOURNISSEUR").Rows.Count - 2
    Chiffres_Dep = Left(Sheets("Fournisseur").Range("C" & i + 2).Text, 2)
    If Département_In.Value = Chiffres_Dep Then
        Id_Fournisseur = Sheets("Fournisseur").Range("A" & i + 2).Text

        Sheets("Produit").Activate
        Nombre_Vendu = 0
        For j = 1 To Range("PRODUIT").Rows.Count - 2
            If Sheets("Produit").Range("H" & j + 2).Text = Id_Fournisseur Then
                Id_Produit = Sheets("Produit").Range("A" & j + 2).Text

                Sheets("Ligne Facture").Activate
                For k = 1 To Range("LIGNEFACTURE").Rows.Count - 2
                    If Sheets("Ligne Facture").Range("C" & k + 2).Text = Id_Produit Then
                        Nombre_Vendu = Nombre_Vendu + CInt(Sheets("Ligne Facture").Range("D" & k + 2).Text)
                    End If
                Next k
            End If
        Next j
        Label_Affichage_Ventes.Visible = True
        Label_Affichage_Ventes.Caption = (Label_Affichage_Ventes.Caption & Chr(13) & Chr(10) & Chr(13) & Chr(10) & _
        & Chiffres_Dep & " / " & Id_Fournisseur & " / " & Nombre_Vendu)
    End If
End If
Next i
End Sub
```

Lorsqu'on clique sur le bouton Valider, le code ci-dessus d'exécute. On parcourt tout d'abord la table « FOURNISSEUR » pour récupérer les deux premiers chiffres du code postal, grâce à :

Chiffres_Dep = Left(Sheets("Fournisseur").Range("C" & i + 2).Text, 2)

Puis, on compare le département demandé par l'utilisateur avec Chiffres_Dep. S'ils sont égaux, on additionne le nombre de produits vendus par ce producteur en utilisant la encore la fonction CInt() pour le calcul. On affiche enfin toutes les informations pour chaque producteur dans le département recherché.

17. Alerte des livraisons en retard :



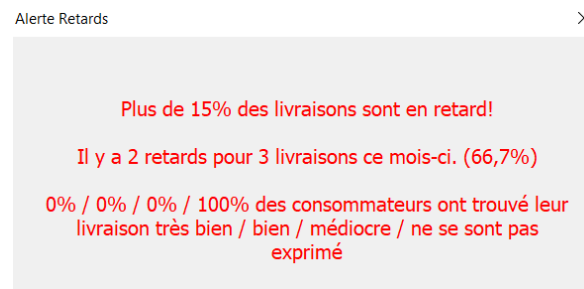
Après activation de la macro, la fonction suivante est appelée :

```
Sub Alert_Delay()
    nLivrMois = 0
    nRetard = 0
    tb = 0
    b = 0
    m = 0
    ne = 0
    Sheets("Livraison").Activate
    nLivraison = Range("LIVRAISON").Rows.Count - 2
    For j = 1 To nLivraison
        If Year(Cells(j + 2, 2)) = Year(Date) Then
            nLivrMois = nLivrMois + 1
            Retard = CLng(Cells(j + 2, 3)) - CLng(Cells(j + 2, 2))
            If Retard > 0 Then
                If Cells(j + 2, 5) = "Très Bien" Then
                    tb = tb + 1
                ElseIf Cells(j + 2, 5) = "Bien" Then
                    b = b + 1
                ElseIf Cells(j + 2, 5) = "Médiocre" Then
                    m = m + 1
                ElseIf Cells(j + 2, 5) = "NE" Then
                    ne = ne + 1
                End If
                nRetard = nRetard + 1
            End If
        End If
    Next j
    If nRetard <> 0 & nLivrMois <> 0 Then
        If nRetard / nLivrMois * 100 > 15 Then
            Alerte_Retard.Label1.ForeColor = vbRed
            Alerte_Retard.Label1 = "Plus de 15% des livraisons sont en retard!" & Chr(10) & Chr(13) _
            & "Il y a " & nRetard & " retards pour " & nLivrMois & _
            " livraisons ce mois-ci. (" & Round(nRetard / nLivrMois * 100, 1) & "%)" & Chr(13) & Chr(13) _
            & Round(tb / nRetard * 100, 1) & "% / " & Round(b / nRetard * 100, 1) & "% / " & Round(m / nRetard * 100, 1) _
            & "% / " & Round(ne / nRetard * 100, 1) _
            & "% des consommateurs ont trouvé leur livraison très bien / bien / médiocre / ne se sont pas exprimé "
        End If
    Else:
        Alerte_Retard.Label1 = "Moins de 15% des livraisons sont en retard." & Chr(10) & Chr(13) & "Il y a " _
        & nRetard & " retards pour " & nLivrMois & " livraisons ce mois-ci. (" & Round(nRetard / nLivrMois * 100, 1) & "%)" _
        & Chr(13) & Chr(13) & Round(tb / nRetard * 100, 1) & "% / " & Round(b / nRetard * 100, 1) _
        & "% / " & Round(m / nRetard * 100, 1) & "% / " & Round(ne / nRetard * 100, 1) _
        & "% des consommateurs ont trouvé leur livraison très bien / bien / médiocre / ne se sont pas exprimé "
    End If
End Sub
```

Dans le code précédent, on parcourt la table des livraisons pour trouver les livraisons de l'année. On calcule alors leur retard sous format Long en faisant la différence entre la date prévue initialement et la date réelle. On récupère ensuite le retour client pour chaque livraison et on compte le nombre de retours. Puis on affiche l'alerte en fonction du pourcentage de livraisons en retard, qu'on calcule comme cela :

$\text{Round}(n\text{Retard} / n\text{LivrMois} * 100, 1)$

L'affichage final devient alors :



18. Statistiques sur les retards :



En activant la macro, le code suivant s'exécute :

```
Sub Affiche_Stat_Delay()
    Sheets("Cause Retard").Activate
    nRetard = Range("CAUSE_RETARD").Rows.Count - 2
    a = 0
    b = 0
    c = 0
    For j = 1 To nRetard
        If Cells(j + 2, 2) = "Rupture de stock" Then
            a = a + 1
        ElseIf Cells(j + 2, 2) = "Grève des transports" Then
            b = b + 1
        ElseIf Cells(j + 2, 2) = "Perte de commande" Then
            c = c + 1
        End If
    Next j
    Stat_Delay.Label1.Caption = Round(a / nRetard * 100) & "% des retards sont dûs à une rupture de stock." _
    & Chr(10) & Chr(13) & Chr(10) & Chr(13) _
    & Round(b / nRetard * 100) & "% des retards sont dûs à une grève des transporteurs." _
    & Chr(10) & Chr(13) & Chr(10) & Chr(13) _
    & Round(c / nRetard * 100) & "% des retards sont dûs à une perte de la commande." _
    & Chr(10) & Chr(13) & Chr(10) & Chr(13)
    Stat_Delay.Show
End Sub
```

exOn parcourt la table retard et on compte les causes de retard par type. Puis on affiche les causes des retards dans le label comme ci-dessous :

