

Vérification de la qualité de donnée

projet VBA

Ahmedou Mohamed Ethmane

Senes quentin

Mohamed EL Hafed Ismail

Sehli mouad



Table de matières:

Introduction

Mode de fonctionnement du code

L'outlier

Outlier valeur continue

Outlier texte

Outlier valeur catégorielle

Outlier date

Imputation

Imputation manuelle

Imputation automatique

Explication du code

Userform outlier

Vérification de choix de l'utilisateur

variable continue

Suppression d'autre type de variable

Regle $1,5 * \text{ecart interquartile}$

variable catégorielle

date

texte

Iteration du processus

Userform imputation

Vérification de choix de l'utilisateur

Imputation manuelle

Imputation automatique

Conclusion

Introduction

Le contrôle de qualité des données est un enjeu majeur pour toute entreprise qui traite des données. Les erreurs dans les données peuvent conduire à des analyses erronées, à des décisions inappropriées et à des coûts considérables. C'est pourquoi, il est essentiel de garantir la qualité des données avant de les utiliser pour prendre des décisions importantes. Dans ce contexte, notre projet consiste en la création d'un outil de vérification de la qualité de données dans un fichier Excel, en utilisant le langage de programmation VBA (Visual Basic for Applications). L'objectif de cet outil est d'aider les utilisateurs à contrôler rapidement et efficacement la qualité de leurs données, et de corriger les éventuelles erreurs détectées.

Notre outil de vérification de la qualité de données utilise des algorithmes sophistiqués pour analyser les données du fichier Excel donné, et pour identifier les valeurs aberrantes, manquantes ou incohérentes. Une fois ces erreurs identifiées, l'outil propose différentes options pour les corriger, telles que la substitution des valeurs aberrantes par des valeurs plus réalistes, la suppression des valeurs incohérentes ou la demande à l'utilisateur de saisir les données manquantes.

L'outil de vérification de la qualité de données que nous avons développé est facile à utiliser, même pour des utilisateurs qui ne sont pas familiers avec le langage de programmation VBA. Il est également hautement personnalisable, ce qui permet aux utilisateurs de l'adapter à leurs besoins spécifiques. De plus, l'outil est capable de traiter des fichiers Excel de grande taille, ce qui est essentiel pour les entreprises qui traitent de grandes quantités de données.

Il a été aussi préparé pour traiter différents types d'entrées (variable catégorielle, variable continue, texte et date) et il les traite aussi avec beaucoup de précision, il tiendra compte des détails mineurs comme les années bisextiles par exemple. On a aussi pris en considération le fait que certaines entrées ne sont peut imputable c'est à dire qu'on ne peut pas les remplacer par d'autres valeurs car les rentrée erronée ne nous donne pas d'information sur les valeurs exactes q'on doit mettre (c'est le cas du texte par exemple), c'est pour cela qu'on a choisis de demander l'utilisateur de saisir les informations dans ce type descas.

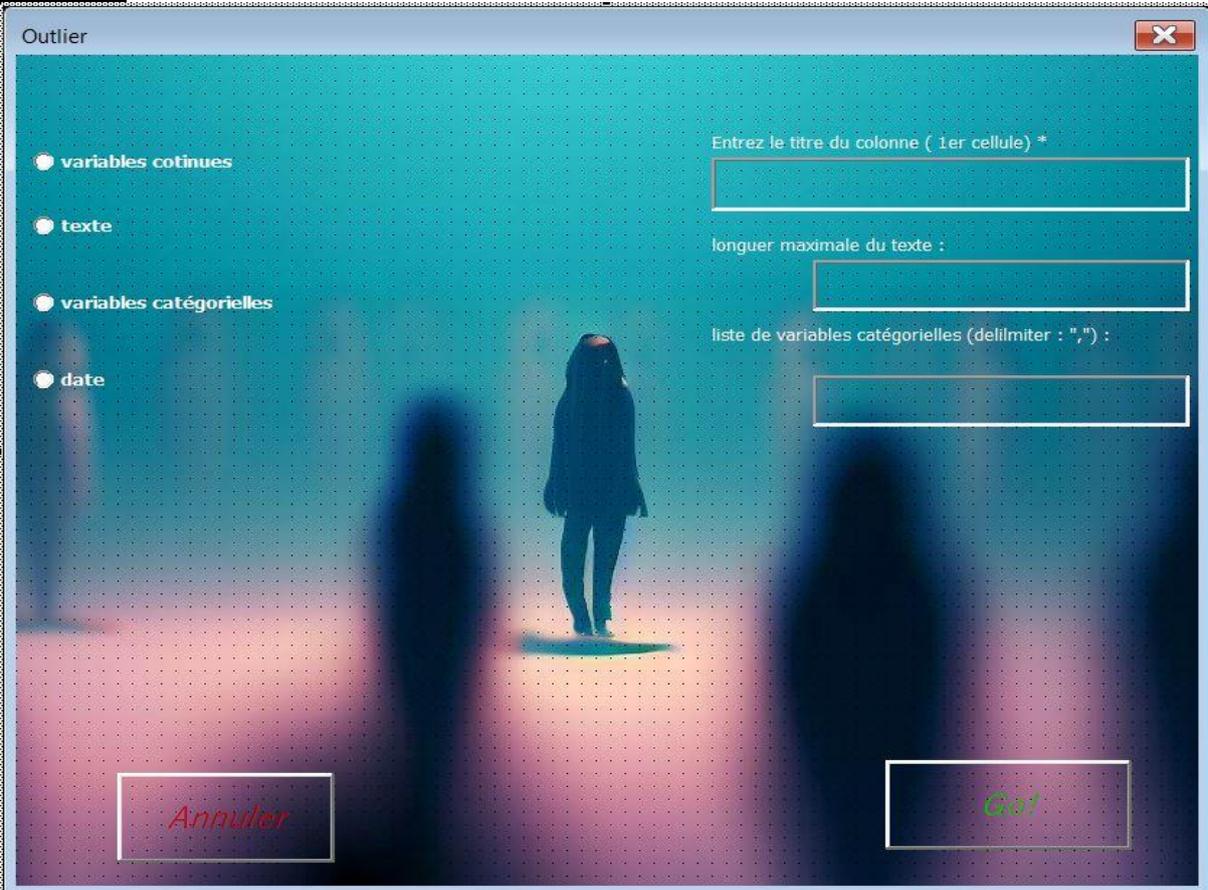
Dans ce rapport, nous allons décrire en détail les différentes parties de notre outil de vérification de la qualité de données. Nous espérons que notre outil contribuera à améliorer la qualité des données utilisées par les entreprises, et qu'il leur permettra de prendre des décisions plus éclairées et plus fiables.

Mode de fonctionnement du code

Notre code VBA est conçu pour vérifier, nettoyer et améliorer les données dans un fichier Excel. Tout d'abord, le code parcourt chaque fois les cellules d'une colonne du fichier Excel pour identifier les valeurs fausses ou aberrantes. Cela peut inclure des données incorrectes, des valeurs manquantes ou des entrées qui ne répondent pas aux critères définis. Une fois que ces valeurs problématiques sont repérées, le code les remplace par des valeurs plus réalistes ou demande à l'utilisateur de les remplacer manuellement en lui fournissant des informations pour l'aider.

Il se compose de 2 grands userforms : Outliers et imputations

L'outlier:



UserForm "Outlier" fournit une interface pratique pour l'identification et le traitement des valeurs aberrantes dans différents types de colonnes. Il offre une flexibilité en permettant à l'utilisateur de choisir le type de la colonne à traiter et propose des fonctionnalités spécifiques à chaque type.

Il a pour but d'identifier les valeurs aberrantes, manquantes ou incorrectes, il comprend différents outliors selon le type de la colonne à traiter.

Il détermine la colonne par le titre saisi par l'utilisateur, puis selon le choix de l'utilisateur, il applique l'une des quatre méthodes d'Outliers ci-dessous.

-Outlier valeur continue:

Dans cet outier, chaque cellule identifiée comme aberrante est remplacée par un zéro et colorée en rouge (pour qu'elle soit identifiable après si l'on veut faire l'imputation).

L'identification de valeurs dans la colonne choisie par l'utilisateur comporte deux étapes :

- °° On identifie les cellules de valeurs non numériques.

- °° Puis pour les autres cellules, on utilise la méthode d'identification de valeurs aberrantes par $1.5 * \text{l'écart interquartile}$. Cette approche repose sur le calcul de l'écart interquartile (EI), souvent utilisé pour identifier les données aberrantes dans un ensemble. L'EI est déterminé par la soustraction du premier quartile (Q1) du troisième quartile (Q3). Ces limites supérieure et inférieure sont obtenues en multipliant l'EI par 1,5, puis en ajoutant (pour la limite supérieure) ou en soustrayant (pour la limite inférieure) ce résultat de Q3 et Q1 respectivement. En d'autres termes, toute valeur au-delà de ces limites est considérée comme aberrante.

-Outlier Texte :

Dans cet outier, chaque cellule identifiée comme aberrante est remplacée par le texte "Erreur" et colorée en rouge.

L'identification de valeurs dans la colonne choisie par l'utilisateur comporte deux étapes :

- °° On identifie les cellules de valeurs de type différent de texte.

- °° Puis, si l'utilisateur décide de restreindre la longueur du texte, on vérifie si le texte a une longueur inférieure à une longueur limite saisie par l'utilisateur pour éviter d'avoir des textes qui sont trop longs.

-Outlier Variable catégorielle :

Dans cet outier, chaque cellule identifiée comme aberrante est remplacée par le texte "catégorielle" et colorée en rouge.

L'identification de valeurs dans la colonne choisie par l'utilisateur est basée sur la liste saisie par l'utilisateur (H, F par exemple). Si une cellule ne contient pas l'une des composantes de la liste, cette cellule est considérée comme aberrante.

-Outlier Date :

Dans cet outier, chaque cellule identifiée comme aberrante est remplacée par le texte "mm/jj/aaaa" et colorée en rouge.

L'identification de valeurs dans la colonne choisie par l'utilisateur comporte deux étapes :

On identifie les cellules de valeurs de type différent de date.

Puis pour les autres cellules, on utilise pour les années de ces cellules la méthode d'identification de valeurs aberrantes par $1.5 * \text{l'écart interquartile}$:

- Si une année est hors de l'intervalle, sa cellule est aberrante.

- Sinon, on vérifie si le mois de cette cellule est entre 1 et 12 :

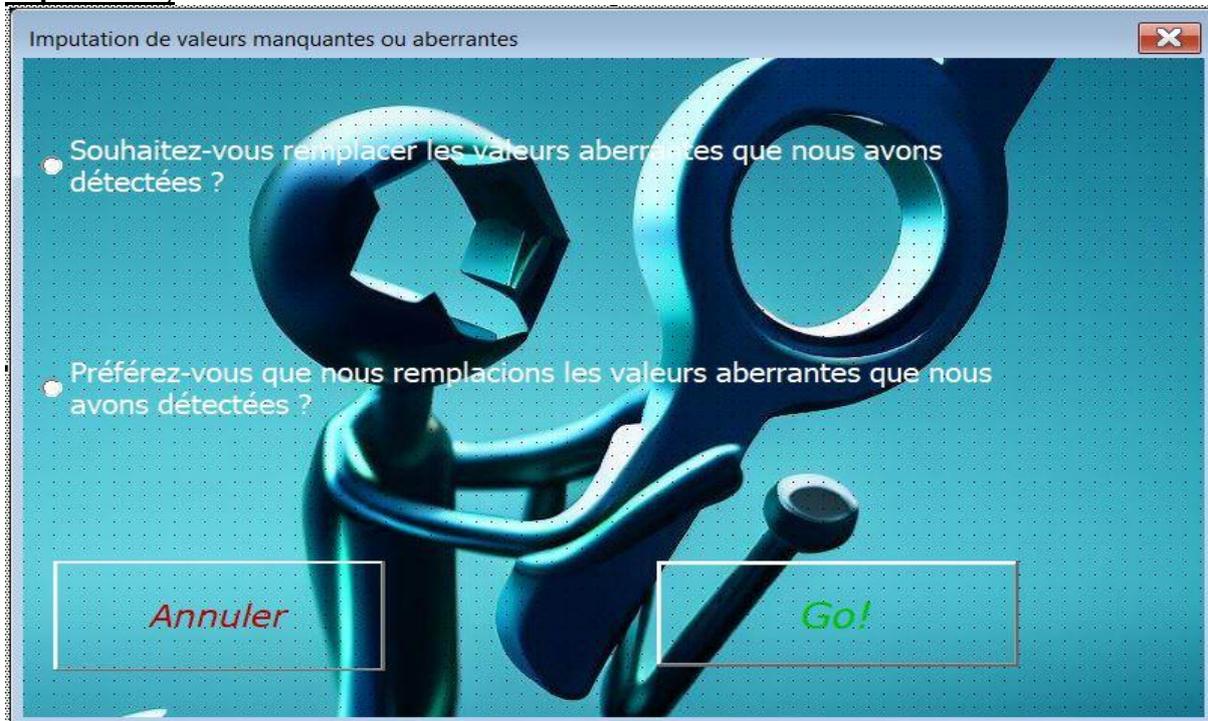
 - Si non, cette cellule est aberrante.

 - Si oui, on vérifie si le jour est entre 1 et le nombre de jours du mois (en étudiant la bissextilité de l'année pour le mois de février). Si non, la cellule est aberrante.

Après le traitement d'une colonne, l'utilisateur est invité à vérifier une autre colonne ou à

passer à l'UserForm "Imputation". En fonction du choix de l'utilisateur, l'UserForm "Outlier" est masqué, et l'UserForm approprié est affiché.

Imputation :



L'UserForm "Imputation" offre des options pour remplacer les valeurs aberrantes, manquantes ou incorrectes identifiées dans l'UserForm "Outlier". Il propose une approche manuelle, où l'utilisateur peut saisir les nouvelles valeurs, ainsi qu'une approche automatique qui utilise des méthodes spécifiques à chaque type de valeur aberrante. Cela permet d'améliorer la qualité des données en effectuant des remplacements appropriés.

Option manuelle :

Lorsque l'option "manuelle" est sélectionnée, l'utilisateur peut effectuer des remplacements de valeurs de manière interactive.

L'UserForm parcourt toutes les cellules non vides de la feuille active.

Si une cellule a une couleur de fond rouge et une valeur spécifique (0, "jj/mm/aaaa", "Erreur", "catégorielle"), cela indique qu'elle a été identifiée comme une valeur aberrante, manquante ou incorrecte.

Pour chaque cellule aberrante, un message est affiché à l'utilisateur avec des informations contextuelles sur la cellule et les autres valeurs de la ligne.

L'utilisateur est invité à saisir une nouvelle valeur pour la cellule aberrante.

La valeur de la cellule est mise à jour avec la nouvelle valeur saisie par l'utilisateur, et la couleur de fond rouge est supprimée.

Option automatique :

Lorsque l'option "automatique" est sélectionnée, l>UserForm effectue des remplacements automatiques des valeurs aberrantes en utilisant des méthodes spécifiques à chaque type de valeur aberrante identifiée :

L>UserForm parcourt toutes les cellules non vides de la feuille active.

Si une cellule a une couleur de fond rouge et une valeur spécifique (0, "jj/mm/aaaa", "Erreur", "catégorielle"), cela indique qu'elle a été identifiée comme une valeur aberrante, manquante ou incorrecte.

En fonction de la valeur de cette cellule, l>UserForm utilise des sous-procédures spécifiques pour effectuer les remplacements appropriés :

Pour les valeurs numériques aberrantes (0), on remplace la valeur par la médiane des autres valeurs numériques non aberrantes et on supprime la couleur rouge.

Pour les valeurs de date aberrantes ("jj/mm/aaaa") , on remplace la valeur par la date médiane des autres dates non aberrantes et on supprime la couleur rouge.

Pour les valeurs textuelles aberrantes ("Erreur"), on remplace la valeur par une indication spécifique, indiquant qu'elle n'a pas été trouvée et on supprime la couleur rouge.

Pour les valeurs catégorielles aberrantes("catégorielle"), on remplace la valeur par une valeur catégorielle aléatoire, en se basant sur les fréquences des autres valeurs catégorielles non aberrantes et on supprime la couleur rouge.

Explication du code

Userform Outlier :

Interface utilisateur :

L>UserForm "Outlier" présente une interface utilisateur conviviale avec les éléments suivants:

Trois zones de texte :

- a. "titre_de_la_colonne" : Cette zone de texte est destinée à l'utilisateur pour saisir le titre de la colonne qu'il souhaite traiter. Il est obligatoire de renseigner cette information.
- b. "maxLen" : Cette zone de texte est recommandée si les cellules de la colonne sont de type texte. L'utilisateur peut spécifier une longueur maximale pour les valeurs de cette colonne.
- c. "liste de variables catégorielles" : Cette zone de texte permet à l'utilisateur de spécifier une liste de variables catégorielles, séparées par des virgules. Cette option est applicable aux colonnes de type catégoriel.

Quatre boutons d'options (option boutons) :

- a. "variable continue" : L'utilisateur peut sélectionner cette option si la colonne à traiter est de type variable continue.
- b. "texte" : Cette option est à choisir si la colonne à traiter est de type texte.
- c. "variable catégorielle" : L'utilisateur peut sélectionner cette option si la colonne à traiter est de type variable catégorielle.
- d. "type_date" : Cette option est à choisir si la colonne à traiter est de type date.

Deux boutons de commande :

- a. "CommandButton1" : Ce bouton permet de fermer l'UserForm "Outlier".
- b. "CommandButton2" : Ce bouton déclenche le traitement de la colonne en fonction des options sélectionnées par l'utilisateur.

Traitement des colonnes :

Localisation de colonne via la fonction 'FindSelectedRange :

Function FindSelectedRange(titre As String) As range

```
Dim premiereLigne As range
Set premiereLigne = ActiveSheet.Rows(1)
Dim trouver_la_colonne As range
Set trouver_la_colonne = premiereLigne.Find(what:=titre, LookIn:=xlValues, lookat:=xlWhole)
```

```
'Vérifier si la colonne a été trouvée
If trouver_la_colonne Is Nothing Then
    MsgBox "Le titre de la colonne n'a pas été trouvé."
    Set FindSelectedRange = Nothing
    Exit Function
End If
```

```
'Définir la plage sélectionnée pour les cellules non vides de la colonne (à l'exception de la première cellule)
Set FindSelectedRange = range(trouver_la_colonne.Offset(1, 0), Cells(Rows.Count,
trouver_la_colonne.Column).End(xlUp))
```

```
'Vérifier si la plage sélectionnée contient des données
If FindSelectedRange Is Nothing Then
    MsgBox "La plage sélectionnée ne contient pas de données."
    Set FindSelectedRange = Nothing
    Exit Function
End If
```

End Function

La fonction "FindSelectedRange" recherche une colonne spécifique dans une feuille de calcul en utilisant un titre fourni en argument.

On utilise la méthode "Find" pour rechercher le titre spécifié (stocké dans la variable "titre") dans la première ligne de la feuille de calcul.

Les paramètres de la méthode "Find" sont les suivants :

"what" : spécifie la valeur recherchée (le titre de la colonne).

"LookIn" : spécifie où chercher la valeur, dans ce cas "xlValues" indique que la recherche doit être effectuée dans les valeurs des cellules.

"lookat" : spécifie si la recherche doit correspondre exactement à la valeur ou si elle peut être partielle. Dans ce cas, "xlWhole" indique une correspondance exacte.

Si aucune colonne correspondant au titre n'est trouvée, la variable "trouver_la_colonne" est définie sur "Nothing". Un message d'erreur est affiché à l'utilisateur et la fonction se termine en retournant "Nothing".

Si une colonne correspondant au titre est trouvée, la plage sélectionnée est définie à l'aide de la fonction "range".

La plage sélectionnée commence à partir de la cellule située en dessous de la cellule trouvée ("trouver_la_colonne.Offset(1, 0)").

La fin de la plage sélectionnée est déterminée en utilisant la méthode "End(xlUp)" à partir de la dernière ligne de la feuille de calcul dans la colonne correspondante ("Cells(Rows.Count, trouver_la_colonne.Column).End(xlUp)").

La plage sélectionnée comprend donc toutes les cellules non vides de la colonne, à l'exception de la première cellule (qui contient le titre).

Si la plage sélectionnée ne contient aucune donnée (c'est-à-dire toutes les cellules sont vides), un message d'erreur est affiché à l'utilisateur.

La variable "FindSelectedRange" est définie sur "Nothing" et la fonction se termine en retournant "Nothing".

Si la plage sélectionnée contient des données, la variable "FindSelectedRange" est définie sur la plage sélectionnée.

La fonction se termine en retournant la plage sélectionnée.

Traitement d'une colonne contenant des variables continues :

```
Sub TraiteVariableContinue(selectedRange As range)
    Dim Cell1 As range
    Dim q1 As Double, q3 As Double, iqr As Double
    Dim cellValues() As Double
    Dim cellCount As Integer
    Dim inf As Double, sup As Double

    On Error GoTo ErrorHandler

    For Each Cell1 In selectedRange
        If Not IsNumeric(Cell1.Value) Then
            Cell1.Value = 0
            Cell1.Interior.Color = vbRed
        End If
    Next Cell1
    Set Cell1 = Nothing

    ' on applique la méthode d'identification de valeurs aberrantes par 1.5 * l'écart interquartile

    cellCount = 0
    ReDim cellValues(1 To selectedRange.Cells.Count)
    For Each Cell1 In selectedRange
        If Not Cell1.Interior.ColorIndex = 3 And Cell1.Value <> 0 Then
            cellCount = cellCount + 1
            cellValues(cellCount) = Cell1.Value
        End If
    Next Cell1
    Set Cell1 = Nothing

    ReDim Preserve cellValues(1 To cellCount)

    q1 = Application.WorksheetFunction.Quartile.Exc(cellValues, 1)
    q3 = Application.WorksheetFunction.Quartile.Exc(cellValues, 3)
    iqr = q3 - q1
    inf = q1 - 1.5 * iqr
    sup = q3 + 1.5 * iqr

    For Each Cell1 In selectedRange
        If Not Cell1.Interior.ColorIndex = 3 And Cell1.Value <> 0 Then
```

```

    If Cell1.Value < inf Or Cell1.Value > sup Then
        Cell1.Value = 0
        Cell1.Interior.Color = vbRed
    End If
End If
Next Cell1

'Aucune erreur rencontrée, donc quittez la sous-procédure avant le gestionnaire d'erreurs
Exit Sub

ErrorHandler:
    MsgBox "Une erreur a été rencontrée : " & Err.Description, vbCritical
End Sub

```

Cette sub-procédure identifie les valeurs aberrantes dans une plage de cellules spécifiée en utilisant la méthode de l'écart interquartile. Elle remplace les valeurs aberrantes par 0 et les marque en rouge pour les mettre en évidence.

effectue les actions suivantes :

Parcours de chaque cellule dans la plage "selected Range".

Si une cellule a une valeur non numérique, elle est remplacée par 0 et sa couleur de fond est définie en rouge.

Les valeurs numériques non nulles de la plage sont stockées dans un tableau appelé "cell Values".

L'écart interquartile (IQR) est calculé à partir des quartiles (Q1 et Q3) des valeurs numériques non nulles.

Des limites inférieure (inf) et supérieure (sup) sont définies en utilisant l'écart interquartile. Chaque cellule dans la plage est vérifiée à nouveau.

Si une cellule a une valeur numérique non nulle qui est en dehors des limites inférieure et supérieure, sa valeur est remplacée par 0 et sa couleur de fond est définie en rouge.

En résumé, cette sub-procédure identifie les valeurs aberrantes dans une plage de cellules spécifiée en utilisant la méthode de l'écart interquartile. Elle remplace les valeurs aberrantes par 0 et les marque en rouge pour les mettre en évidence.

Si une cellule a une valeur non numérique, elle est remplacée par 0 et sa couleur de fond est définie en rouge.

Les valeurs numériques non nulles de la plage sont stockées dans un tableau appelé "cellValues".

L'écart interquartile (IQR) est calculé à partir des quartiles (Q1 et Q3) des valeurs numériques non nulles.

Des limites inférieure (inf) et supérieure (sup) sont définies en utilisant l'écart interquartile. Chaque cellule dans la plage est vérifiée à nouveau.

Si une cellule a une valeur numérique non nulle qui est en dehors des limites inférieure et supérieure, sa valeur est remplacée par 0 et sa couleur de fond est définie en rouge.

En résumé, cette sub-procédure identifie les valeurs aberrantes dans une plage de cellules spécifiée en utilisant la méthode de l'écart interquartile. Elle remplace les valeurs aberrantes par 0 et les marque en rouge pour les mettre en évidence.

Traitement d'une colonne contenant des textes :

```

Sub TraiteVariableTexte(selectedRange As range)
    Dim cell2 As range
    Dim LongueurMax As Long

```

```

On Error GoTo ErrorHandler

For Each cell2 In selectedRange
    If VarType(cell2.Value) <> vbString Then
        cell2.Value = "Erreur"
        cell2.Interior.Color = vbRed ' Modifiez la couleur de fond en rouge pour
indiquer une valeur aberrante
    End If
Next cell2
Set cell2 = Nothing

' Identifiez les textes aberrants
If IsNumeric(maxLen.Value) Then
    LongueurMax = CLng(maxLen.Value)
    For Each cell2 In selectedRange
        If Not cell2.Interior.ColorIndex = 3 And cell2.Value <> "Erreur" Then 'vérifie si
la cellule n'a pas une couleur de fond rouge et si elle est différente de "Erreur"
            If Len(cell2.Value) > LongueurMax Then
                cell2.Value = "Erreur"
                cell2.Interior.Color = vbRed ' Modifiez la couleur de fond en rouge pour
indiquer une valeur aberrante
            End If
        End If
    Next cell2
Set cell2 = Nothing

' Aucune erreur rencontrée, donc quittez la sous-procédure avant le gestionnaire
d'erreurs
Exit Sub

ErrorHandler:
    MsgBox "Une erreur a été rencontrée: " & Err.Description, vbCritical

End Sub

```

Cette sub-procédure identifie les valeurs aberrantes dans une plage de cellules contenant des valeurs textuelles. Elle marque les valeurs aberrantes en les remplaçant par le texte "Erreur" et en définissant leur couleur de fond en rouge pour les mettre en évidence. La sub-procédure vérifie également la longueur des valeurs textuelles et marque celles qui dépassent une limite spécifiée.

Traitement d'une colonne contenant des variables catégorielles :

```

Sub TraiteVariableCategorielle(selectedRange As range)
    On Error GoTo ErrorHandler

    ' Création outlayer variable catégorielle
    Dim liste As Variant
    Dim cell3 As range
    Dim r As Integer

    ' Obtention de la liste des variables catégorielles à partir du TextBox
    liste = Split(liste_de_variables_catégorielles.Text, ",")

```

```

For Each cell3 In selectedRange
    r = 0
    For j = LBound(liste) To UBound(liste)
        If cell3.Value = liste(j) Then
            r = 1
            Exit For
        End If
    Next j

    If r = 0 Then
        cell3.Value = "catégorielle" ' Remplacer la donnée non valide par "catégorielle"
        cell3.Interior.Color = vbRed ' Modifier la couleur de fond en rouge pour
indiquer une valeur aberrante
    End If
Next cell3
Exit Sub

ErrorHandler:
    MsgBox "Une erreur est survenue lors du traitement de la variable catégorielle.
Veuillez vérifier vos données et réessayer." & Err.Description, vbCritical
End Sub

```

Cette sub-procédure traite les variables catégorielles dans une plage de cellules. Elle compare la valeur de chaque cellule avec une liste de variables catégorielles spécifiée dans un TextBox. Si la valeur de la cellule ne correspond à aucune variable catégorielle de la liste, elle est remplacée par le texte "catégorielle" et sa couleur de fond est définie en rouge pour indiquer une valeur aberrante.

Traitement d'une colonne contenant des dates :

```

Sub TraiteVariableDate(selectedRange As range)
    Dim cell4 As range
    Dim yearStart As Long, yearEnd As Long
    Dim mois As Long, jour As Long, année As Long
    Dim daysInMonth As Long
    Dim listeDesAnnées() As Long
    Dim q1A As Long, q3A As Long, iqrA As Long
    Dim i As Integer

    On Error GoTo ErrorHandler

    For Each cell4 In selectedRange
        If Not IsDate(cell4.Value) Then
            ' Mettre la cellule en rouge et inscrire "jj/mm/aaaa"
            cell4.Interior.Color = vbRed
            cell4.Value = "jj/mm/aaaa"
        End If
    Next cell4
    Set cell4 = Nothing

    i = 0
    ReDim listeDesAnnées(1 To selectedRange.Cells.Count)

    ' Parcourir les cellules de la plage sélectionnée
    For Each cell4 In selectedRange
        If cell4.Interior.ColorIndex <> 3 And cell4.Value <> "jj/mm/aaaa" Then
            i = i + 1
        End If
    Next cell4
End Sub

```

```

        année = year(cell4.Value)
        listeDesAnnées(i) = année
    End If
Next cell4
Set cell4 = Nothing

' Redimensionner le tableau à la taille réelle
ReDim Preserve listeDesAnnées(1 To i)

' Calculer les quartiles et l'intervalle interquartile
q1A = Application.WorksheetFunction.Quartile_Exc(listeDesAnnées, 1)
q3A = Application.WorksheetFunction.Quartile_Exc(listeDesAnnées, 3)
iqrA = q3A - q1A

' Calculer l'intervalle d'années autorisées
yearStart = q1A - 1.5 * iqrA
yearEnd = q3A + 1.5 * iqrA

' Vérifier chaque cellule de la plage sélectionnée
For Each cell4 In selectedRange
    If cell4.Interior.ColorIndex <> 3 And cell4.Value <> "jj/mm/aaaa" Then
        ' Extraire le jour, le mois et l'année de la date
        jour = day(cell4.Value)
        mois = month(cell4.Value)
        année = year(cell4.Value)

        ' Vérifier si l'année est dans l'intervalle autorisé
        If année < yearStart Or année > yearEnd Then
            ' Mettre la cellule en rouge et inscrire "jj/mm/aaaa"
            cell4.Interior.Color = vbRed
            cell4.Value = "jj/mm/aaaa"
        Else
            ' Vérifier si le mois est entre 1 et 12
            If mois < 1 Or mois > 12 Then
                ' Mettre la cellule en rouge et inscrire "jj/mm/aaaa"
                cell4.Interior.Color = vbRed
                cell4.Value = "jj/mm/aaaa"
            Else
                ' Calculer le nombre de jours dans le mois
                Select Case mois
                    Case 2 ' février
                        If année Mod 4 = 0 And (année Mod 100 <> 0 Or année Mod 400 = 0) Then
                            daysInMonth = 29 ' année bissextile
                        Else
                            daysInMonth = 28 ' année non bissextile
                        End If
                    Case 4, 6, 9, 11 ' mois avec 30 jours
                        daysInMonth = 30
                    Case Else ' mois avec 31 jours
                        daysInMonth = 31
                End Select

                ' Vérifier si le jour est entre 1 et le nombre de jours dans le mois
                If jour < 1 Or jour > daysInMonth Then
                    ' Mettre la cellule en rouge et inscrire "jj/mm/aaaa"
                    cell4.Interior.Color = vbRed
                    cell4.Value = "jj/mm/aaaa"
                End If
            End If
        End If
    End If
Next cell4

' Pas d'erreur, donc sortie avant le gestionnaire d'erreur
Exit Sub

```

ErrorHandler:

```
MsgBox "Une erreur est survenue lors du traitement des dates. Veuillez vérifier les données  
et réessayer." & Err.Description, vbCritical  
End Sub
```

Le code de la sub-procédure "TraiteVariableDate" effectue les actions suivantes:

Gestion des erreurs : Si une erreur se produit dans le code, le programme va directement à la partie "ErrorHandler" pour afficher un message d'erreur.
Parcours de chaque cellule dans la plage "selectedRange".

Une boucle "For" parcourt à nouveau chaque cellule dans la plage sélectionnée.
Si la cellule n'a pas de couleur de fond rouge (vérifiée par "cell4.Interior.ColorIndex <> 3") et si la valeur de la cellule n'est pas "jj/mm/aaaa", alors l'année de la date de la cellule est extraite et ajoutée au tableau "listeDesAnnées".

Le tableau "listeDesAnnées" est redimensionné pour correspondre à la taille réelle des années valides.

Les quartiles et l'intervalle interquartile sont calculés à partir du tableau "listeDesAnnées".
L'intervalle autorisé des années est calculé en ajoutant ou soustrayant 1.5 fois l'écart interquartile aux quartiles.

Une dernière boucle "For" parcourt à nouveau chaque cellule dans la plage sélectionnée.
Si la cellule n'a pas de couleur de fond rouge (vérifiée par "cell4.Interior.ColorIndex <> 3") et si la valeur de la cellule n'est pas "jj/mm/aaaa", alors les éléments jour, mois et année de la date de la cellule sont extraits.

L'année est vérifiée pour s'assurer qu'elle est dans l'intervalle autorisé. Si ce n'est pas le cas, la cellule est mise en rouge et la valeur est remplacée par "jj/mm/aaaa".

Si l'année est dans l'intervalle autorisé, le mois est vérifié pour s'assurer qu'il est entre 1 et 12. Sinon, la cellule est mise en rouge et la valeur est remplacée par "jj/mm/aaaa".

Le nombre de jours dans le mois est calculé en fonction du mois et de l'année.

Le jour est vérifié pour s'assurer qu'il est entre 1 et le nombre de jours dans le mois. Si ce n'est pas le cas, la cellule est mise en rouge et la valeur est remplacée par "jj/mm/aaaa".

Le traitement se poursuit pour chaque cellule de la plage.

La sub-procédure se termine normalement si aucune erreur n'est rencontrée.

Transition entre Outlier et Imputation :

```
Dim choix As VbMsgBoxResult  
choix = MsgBox("Voulez-vous vérifier une autre colonne ?", vbYesNo)  
If choix = vbYes Then  
    ' Exécuter l'userform Outlier pour vérifier une autre colonne  
    Me.Hide ' Masquer l'UserForm Outlier actuel  
    Outlier.Show ' Afficher l'userform Outlier à nouveau  
Else  
    ' Exécuter l'userForm Imputation  
    Me.Hide ' Masquer l'UserForm outlier actuel  
    Imputation.Show ' Afficher l'userform imputation  
End If
```

ce code gère la transition entre les différentes étapes du processus en fonction du choix de l'utilisateur : soit vérifier une autre colonne avec l'UserForm Outlier, soit passer à l'UserForm Imputation.

Userform Imputation :

Imputation manuelle:

```
If manuelle.Value Then
' Boucler à travers toutes les cellules non vides de la feuille active
For Each cell In ActiveSheet.UsedRange.Cells
    If cell.Interior.ColorIndex = 3 Then
        If cell.Value = 0 Or cell.Value = "jj/mm/aaaa" Or cell.Value = "Erreur" Or cell.Value =
"catégorielle" Then
            'Obtenir le titre de la colonne correspondante
            col = cell.Column
            titre = Cells(1, col).Value

            msg = "Remplacez la cellule aberrante :" & vbCrLf & vbCrLf

            msg = msg & "Nous avons rassemblé ces informations pour vous aider :" & vbCrLf

            For i = 1 To ActiveSheet.UsedRange.Columns.Count
                If i <> col Then
                    msg = msg & Cells(1, i).Value & ": " & Cells(cell.Row, i).Value & vbCrLf
                End If
            Next i

            MsgBox msg
            'Sélectionner la cellule et supprimer la couleur rouge
            cell.Select
            cell.Interior.ColorIndex = xlNone
            'Demander à l'utilisateur de saisir une nouvelle valeur
            NewVal = InputBox("Entrez une nouvelle valeur pour " & titre & " :")
            'Mettre à jour la valeur de la cellule avec la nouvelle valeur saisie
            cell.Value = NewVal
        End If
    End If
Next cell
Set cell = Nothing
```

Ce code vérifie si l'option "manuelle" est sélectionnée. Ensuite, il parcourt toutes les cellules non vides de la feuille active. Pour chaque cellule ayant un indice de couleur de fond égal à 3 (rouge), il vérifie si la valeur de la cellule est égale à 0, "jj/mm/aaaa", "Erreur" ou "catégorielle". Si la condition est remplie, il effectue les actions suivantes:

Il récupère le titre de la colonne correspondante à l'aide de la variable "col" qui représente le numéro de colonne de la cellule.

Il construit un message contenant des informations utiles pour aider à identifier la cellule aberrante.

Il affiche le message dans une boîte de dialogue.

Il sélectionne la cellule et supprime la couleur rouge de fond.

Il demande à l'utilisateur de saisir une nouvelle valeur pour la cellule en utilisant une boîte de dialogue d'entrée.

Il met à jour la valeur de la cellule avec la nouvelle valeur saisie.

Ce processus permet à l'utilisateur de remplacer manuellement les valeurs aberrantes par des valeurs appropriées.

Imputation automatique:

Imputation pour les variables continues :

```
Sub HandleNumericCase(ByRef cell As range)
    Dim c As range

    ' Obtenir le numéro de colonne de la cellule
    col = cell.Column

    ' Parcourir chaque cellule dans la colonne de la cellule en question
    For Each c In ActiveSheet.Columns(col).Cells
        If c.Interior.ColorIndex <> 3 And c.Value <> 0 Then
            cellCount = cellCount + 1
            ReDim Preserve cellValues(1 To cellCount)
            cellValues(cellCount) = c.Value
        End If
    Next c

    ' Remplacer 0 par la médiane des cellules numériques non zéro et non rouge
    cell.Value = WorksheetFunction.Median(cellValues)
    cell.Interior.ColorIndex = xlNone

End Sub
```

Cette procédure parcourt les cellules d'une colonne spécifique, identifie les cellules numériques non nulles et non rouge, calcule la médiane de ces valeurs et remplace les zéros par la valeur médiane dans la cellule d'origine.

Imputation pour les dates :

La fonction MedianDate :

```
Function MedianDate(cell As range) As Date
    ' Cette fonction prend une cellule comme argument et retourne la date médiane de
    la colonne à laquelle appartient cette cellule
    Dim rng As range
    Dim ws As Worksheet
    Dim dataArr() As Date
    Dim i As Long, j As Long
    Dim temp As Date

    ' Définir la feuille de calcul et la plage de cellules
    Set ws = cell.Worksheet
    Set rng = ws.range(cell, ws.Cells(ws.Rows.Count, cell.Column).End(xlUp))

    ' Redimensionner le tableau en fonction du nombre de cellules
    ReDim dataArr(1 To rng.Cells.Count)

    ' Remplir le tableau avec des valeurs de date
    For Each c In rng
        ' On ignore la première ligne et les cellules qui ne sont pas des dates
        If c.Row > 1 And IsDate(c.Value) Then
            i = i + 1
            dataArr(i) = CDate(c.Value)
        End If
    Next c

    ' Redimensionner le tableau pour qu'il contienne uniquement les dates valides
    ReDim Preserve dataArr(1 To i)
End Function
```

```

' Trier le tableau
For i = LBound(dataArr) To UBound(dataArr) - 1
  For j = i + 1 To UBound(dataArr)
    ' Si la date actuelle est plus grande que la suivante, les échanger
    If dataArr(i) > dataArr(j) Then
      temp = dataArr(i)
      dataArr(i) = dataArr(j)
      dataArr(j) = temp
    End If
  Next j
Next i

' Obtenir la valeur médiane
i = UBound(dataArr)
' Si le nombre de dates est pair, la médiane est la moyenne des deux dates
centrales
If i Mod 2 = 0 Then
  MedianDate = (dataArr(i / 2) + dataArr(i / 2 + 1)) / 2
' Si le nombre de dates est impair, la médiane est la date centrale
Else
  MedianDate = dataArr((i + 1) / 2)
End If

End Function

```

Cette fonction extrait les dates de la colonne correspondante à partir de la cellule donnée, les trie en ordre croissant, puis calcule la médiane en fonction du nombre de dates.

Le code :

```

Sub HandleDateCase(ByRef cell As range)

  cell.Value = MedianDate(cell)
  cell.Interior.ColorIndex = xlNone

End Sub

```

Imputation pour les variables catégorielle :

La fonction GetRandomValue :

```

Function GetRandomValue(values() As Variant) As Long
  Dim randNum As Double
  Dim i As Long
  Dim total As Double

  ' Calculer la somme des pondérations des valeurs
  total = 0
  For i = 0 To UBound(values)
    total = total + values(i)(1)
  Next i

  ' Générer un nombre aléatoire entre 0 et la somme des pondérations
  randNum = Rnd * total

  ' Parcourir les valeurs et ajouter les pondérations jusqu'à ce que la somme
  soit supérieure au nombre aléatoire
  total = 0
  For i = 0 To UBound(values)
    total = total + values(i)(1)
    ' Si la somme dépasse le nombre aléatoire, retourner l'index de cette
    valeur
  Next i
End Function

```

```

    If randNum <= total Then
        GetRandomValue = i
    Exit Function
    End If
Next i
End Function

```

```

Function MedianDate(cell As range) As Date
' Cette fonction prend une cellule comme argument et retourne la date
médiante de la colonne à laquelle appartient cette cellule
Dim rng As range
Dim ws As Worksheet
Dim dataArr() As Date
Dim i As Long, j As Long
Dim temp As Date

' Définir la feuille de calcul et la plage de cellules
Set ws = cell.Worksheet
Set rng = ws.range(cell, ws.Cells(ws.Rows.Count, cell.Column).End(xlUp))

' Redimensionner le tableau en fonction du nombre de cellules
ReDim dataArr(1 To rng.Cells.Count)

' Remplir le tableau avec des valeurs de date
For Each c In rng
' On ignore la première ligne et les cellules qui ne sont pas des dates
If c.Row > 1 And IsDate(c.Value) Then
    i = i + 1
    dataArr(i) = CDate(c.Value)
End If
Next c

' Redimensionner le tableau pour qu'il contienne uniquement les dates
valides
ReDim Preserve dataArr(1 To i)

' Trier le tableau
For i = LBound(dataArr) To UBound(dataArr) - 1
    For j = i + 1 To UBound(dataArr)
' Si la date actuelle est plus grande que la suivante, les échanger
If dataArr(i) > dataArr(j) Then
        temp = dataArr(i)
        dataArr(i) = dataArr(j)
        dataArr(j) = temp
    End If
    Next j
Next i

' Obtenir la valeur médiane
i = UBound(dataArr)
' Si le nombre de dates est pair, la médiane est la moyenne des deux
dates centrales
If i Mod 2 = 0 Then
    MedianDate = (dataArr(i / 2) + dataArr(i / 2 + 1)) / 2
' Si le nombre de dates est impair, la médiane est la date centrale
Else
    MedianDate = dataArr((i + 1) / 2)
End If

End Function

```

La fonction GetRandomValue permet d'obtenir une valeur aléatoire pondérée à partir d'un tableau de valeurs pondérées.

Elle calcule la somme des pondérations des valeurs en parcourant le tableau values avec une boucle For. Chaque élément du tableau values est un tableau lui-même, contenant la valeur à sélectionner et sa pondération.

Elle génère un nombre aléatoire entre 0 et la somme des pondérations en utilisant la fonction Rnd (qui renvoie un nombre aléatoire entre 0 et 1) et en le multipliant par total. Parcours des valeurs dans le tableau values avec une autre boucle For et ajout des pondérations jusqu'à ce que la somme dépasse le nombre aléatoire généré. Si la somme dépasse le nombre aléatoire, la fonction retourne l'index de cette valeur dans le tableau values.

Le code :

```
Sub HandleCategoricalCase(ByRef cell As range)
    ' Obtenir le numéro de colonne de la cellule
    col = cell.Column

    ' créer un objet dictionnaire pour stocker les valeurs catégorielles et leur
    fréquence
    Set categList = CreateObject("Scripting.Dictionary")

    Set categRange = ActiveSheet.Columns(col).Cells(2).End(xlDown)

    ' parcourir la plage de cellules et stocker les valeurs catégorielles et leur
    fréquence dans un dictionnaire
    If Not categRange Is Nothing Then
        For Each val In categRange
            If val.Value <> "catégorielle" And val.Interior.ColorIndex <> 3 Then
                ' si la valeur catégorielle n'existe pas dans le dictionnaire,
                l'ajouter avec une fréquence de 1
                If Not categList.Exists(val.Value) Then
                    categList.Add val.Value, 1
                ' sinon, augmenter la fréquence de la valeur catégorielle dans le
                dictionnaire de 1
                Else
                    categList(val.Value) = categList(val.Value) + 1
                End If
            End If
        Next val
    End If

    ' réinitialiser la couleur de fond de la cellule traitée
    cell.Interior.ColorIndex = xlNone

    ' obtenir la liste des valeurs catégorielles du dictionnaire
    valueList = categList.keys()

    ' calculer la somme des fréquences des valeurs catégorielles
    total = Application.WorksheetFunction.Sum(categList.items())

    ' normaliser les fréquences des valeurs catégorielles dans le dictionnaire
    For i = 0 To UBound(valueList)
        categList(valueList(i)) = categList(valueList(i)) / total
    Next i

    ' obtenir une valeur aléatoire en fonction des fréquences normalisées des
    valeurs catégorielles
    randValue = GetRandomValue(categList.items())

    ' affecter la valeur catégorielle correspondante à la cellule traitée
    cell.Value = valueList(randValue)
End Sub
```

End Sub

La fonction HandleCategoricalCase traite les cellules contenant des valeurs catégorielles en sélectionnant aléatoirement une valeur catégorielle pondérée à partir des fréquences observées dans la colonne.

Imputation pour les textes :

```
Sub HandleTextCase(ByRef cell As range)
    ' Obtenir le numéro de colonne de la cellule
    col = cell.Column

    cell.Value = cell(1, col).Value & "pas trouvé"
    cell.Interior.ColorIndex = xlNone
End Sub
```

Conclusion

En somme, Notre application développée est à la fois une réussite et un défi. Elle représente une réussite parce qu'elle donne à l'utilisateur un large éventail de choix, permettant une personnalisation poussée et une grande exactitude dans les résultats. Cependant, cette multiplicité d'options présente également un défi : elle peut générer une complexité involontaire, ce qui peut rendre l'application difficile à utiliser pour certains utilisateurs, particulièrement les novices. Par conséquent, Nous avons cherché à trouver un équilibre entre la flexibilité et facilité d'utilisation . Notre 'objectif était de conserver la richesse des options tout en simplifiant l'interface utilisateur et en clarifiant le processus de sélection des options pour que les utilisateurs puissent profiter pleinement de toutes les fonctionnalités de l'application.