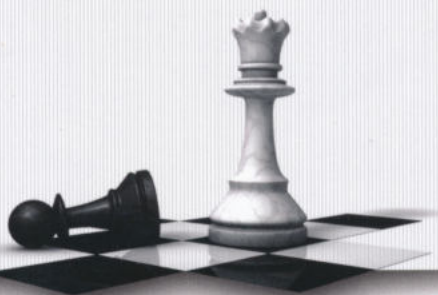


Apprenez le langage VBA

et devenez un expert sur Excel

Jean-Emmanuel CHAPARTEGUI

Fichiers complémentaires
à télécharger



Solutions Business



Solutions Business 

Apprenez le langage VBA

et devenez un expert sur Excel

Chapitre 1

Introduction

A. Introduction	9
1. Pourquoi apprendre VBA ?	9
2. Pourquoi ce livre ?	10
3. Quel est le niveau nécessaire pour lire ce livre ?	11
4. Quel sera votre niveau à la fin de la lecture de ce livre ?	11
5. Comment lire ce livre ?	11
6. L'auteur	12
B. Avant de commencer	13
1. Vocabulaire spécifique	13
2. Raccourcis-clavier	14
a. Manipulation d'un classeur	14
b. Manipulation du tableur	14
3. Versions de Microsoft Office Excel	15
a. Excel 2003	15
b. Excel 2007	16
c. Excel 2010	16
d. Excel 2013	16
e. Excel 2016	16
f. Office pour Mac	17
g. OpenOffice.org	17
h. Version du livre	17
i. Langue du produit Office	17

Chapitre 2

Gestion des employés : exploitation de données brutes

A. Formules Excel avancées	21
1. Description de l'exemple	21
a. Présentation de l'exemple	21
b. Présentation du classeur	22
c. Fonctionnalités	24
2. Notions de cours	25
a. Liste déroulante dans une cellule – Validation de données	25
b. Formule de recherche	29
c. Structure conditionnelle sur Excel : conditions et SI	32
d. Formule conditionnelle	32

e.	Gestion des cas d'erreur	34
f.	Calcul matriciel	35
3.	Réalisation de l'exemple	36
a.	Nommage des plages	36
b.	Fonctions de recherche : afficher le poste et le salaire de l'année précédente	38
c.	Gestion des erreurs et formules conditionnelles	42
d.	Gérer l'erreur sur le salaire des nouveaux arrivants	43
e.	Salaire moyen par grade et filière : calcul matriciel contre formule conditionnelle	44
f.	Création de la pyramide des âges	49
B.	Indicateurs clés et partages	57
1.	Description de l'exemple	57
a.	Présentation de l'exemple	57
b.	Présentation du classeur	57
c.	Fonctionnalités	57
2.	Notions de cours	57
a.	Formules Excel	57
b.	Création d'un graphique Sparkline	60
c.	Mise en forme conditionnelle simple	61
d.	Introduction au tableau	67
3.	Réalisation de l'exemple	68
a.	Mise en place du tableau	68
b.	Formules statistiques	69
c.	Mise en valeur des données	71
d.	Sparkline contre graphique classique	73

Chapitre 3

Gestion des ventes et formulaires VBA

A.	Formulaire de gestion des ventes	79
1.	Description de l'exemple	79
a.	Présentation de l'exemple	79
b.	Présentation du fichier	80
c.	Fonctionnalités	81
2.	Notions de cours	82
a.	Concept de programmation	82
b.	Concept de formulaire	84
c.	Rédaction du code	86

d. Le fonctionnement de l'éditeur Visual Basic	90
3. Réalisation de l'exemple	92
a. Création du formulaire	92
b. Création des contrôles sur le formulaire	96
c. Définition des procédures et événements	101
d. Rédaction du code : procédures et événements	103
B. Protéger le classeur	132
1. Description de l'exemple	132
a. Présentation de l'exemple	132
b. Présentation du fichier	133
c. Fonctionnalités	133
2. Notions de cours	133
a. Afficher/masquer une feuille	133
b. Protéger la structure	135
c. Protéger la feuille et ses cellules	137
d. Protéger le code VBA	140
3. Création de l'exemple	142
a. Masquer les feuilles Factures et Produits	142
b. Protéger la structure du classeur	143
c. Afficher les stocks via un formulaire	143
d. Protéger les cellules de la feuille Accueil	149
e. Protéger le code VBA	152

Chapitre 4

Gestion d'une campagne de test

A. Création de tableaux et graphiques croisés dynamiques (TCD et GCD)	155
1. Description de l'exemple	155
a. Présentation de l'exemple	155
b. Présentation du fichier	156
c. Fonctionnalités	159
2. Notions de cours	159
a. Créer un tableau croisé dynamique simple	159
b. Créer un tableau croisé dynamique avec l'assistant	162
c. Champs calculés et éléments calculés	169
d. Créer un graphique croisé dynamique	172
3. Réalisation de l'exemple	174
a. Stock d'anomalies	174
b. Nombre d'anomalies par projets (et par priorité)	178

c.	Avancement des cas de tests	184
d.	Revue des cycles de test	189
e.	Indicateur de situation des tests	194
B.	Automatisation de la création d'un rapport PowerPoint	200
1.	Description de l'exemple	200
a.	Présentation de l'exemple	200
b.	Présentation du fichier	201
c.	Fonctionnalités	202
2.	Notions de cours	203
a.	Enregistrement de macro	203
b.	Créer un tableau croisé dynamique avec VBA	205
c.	Créer un graphique avec VBA	207
d.	Manipuler PowerPoint	208
3.	Réalisation de l'exemple	210
a.	Actualiser et copier les graphiques	210
b.	Nombre de tests par personne	214
c.	Mise en forme du rapport	218
d.	Création du rapport PowerPoint	223
e.	Finalisation	227

Chapitre 5

Gestion des employés

A.	Calcul de la durée et du planning	233
1.	Description de l'exemple	233
a.	Présentation de l'exemple	233
b.	Présentation du fichier	233
c.	Fonctionnalités	235
2.	Notions de cours	235
a.	Formules de date	235
b.	Mise en forme conditionnelle avancée	237
3.	Réalisation de l'exemple	240
a.	Calcul de la durée de chaque tâche	240
b.	Mise en forme du diagramme de Gantt	249
B.	Gestion des présences - Outil d'administration	254
1.	Description de l'exemple	254
a.	Présentation de l'exemple	254
b.	Présentation du fichier	254
c.	Fonctionnalités	257

2. Notions de cours	258
a. Création dynamique de contrôle	258
b. Tableaux VBA	259
3. Réalisation de l'exemple	260
a. Initialisation du formulaire.	260
b. Bloquer l'accès à la feuille Planning.	270
c. Calculer le coût du projet.	271

Chapitre 6

Consolidation et partage de données

A. Consolidation de données diverses	283
1. Description de l'exemple	283
a. Présentation de l'exemple	283
b. Présentation des classeurs	284
c. Fonctionnalités.	286
2. Notions de cours	287
a. Manipulation de feuilles et classeurs	287
b. Sélection et ouverture d'un classeur Excel	287
c. Les boucles	288
d. Format de la cellule	289
e. Formule Excel dans le code VBA	289
f. Select Case et structure conditionnelle.	290
3. Réalisation de l'exemple	291
a. Structure du code	291
b. Déclaration des variables feuille et classeur	292
c. Définition de la boîte de dialogue d'ouverture de fichier	293
d. Parcours des feuilles.	294
B. Partage des données	309
1. Description de l'exemple	309
a. Présentation de l'exemple	309
b. Présentation des classeurs et outils utilisés	309
c. Fonctionnalités.	310
2. Notions de cours	311
a. Formulaire de tableau	311
b. OneDrive	316
c. Enquêtes	317
d. Envoyer un e-mail avec VBA via Outlook.	319

3. Réalisation de l'exemple	322
a. Créer un formulaire de saisie automatique pour faciliter la saisie des données	322
b. Créer une enquête partagée via OneDrive et la diffuser.	326
c. Envoyer un e-mail avec les statistiques des ventes aux agences.....	335
Index	343

Chapitre 1
Introduction

A. Introduction	9
B. Avant de commencer	13

A. Introduction

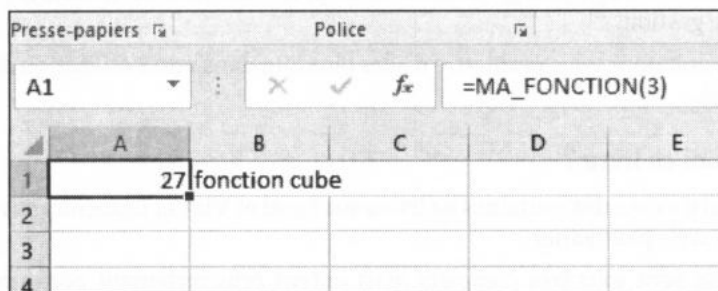
1. Pourquoi apprendre VBA ?

Si vous êtes intéressé par ce livre, c'est que vous ne connaissez pas forcément très bien ce qu'est Visual Basic for Application et surtout ce qu'il est possible de faire avec.

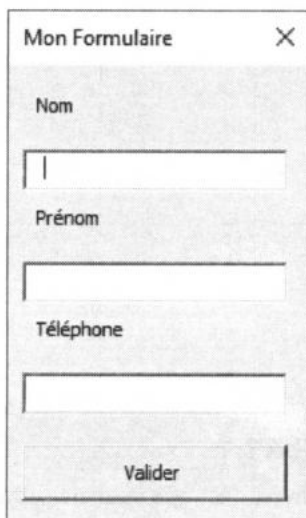
Visual Basic for Application est un langage de programmation basé sur Microsoft Visual Basic mis en place pour Microsoft Office. L'objectif premier était d'apporter des fonctionnalités supplémentaires aux outils de la suite Office et plus particulièrement à Microsoft Excel.

En effet, la principale fonction de VBA pour Excel est d'aider à l'automatisation du calcul dans le tableur Excel, mais VBA va bien au-delà :

- ▶ VBA permet de créer des fonctions Excel, gérées comme les formules Excel natives.



- ▶ VBA introduit la notion de formulaire utilisateur, ce qui permet à l'utilisateur d'interagir avec l'application.



The image shows a VBA form window titled "Mon Formulaire" with a close button (X) in the top right corner. The form contains three text input fields: "Nom", "Prénom", and "Téléphone". Below these fields is a button labeled "Valider".

- Mais surtout, VBA implémente de nombreuses fonctionnalités permettant par exemple : d'envoyer un mail (abordé dans le livre), de créer un rapport PowerPoint (abordé dans le livre), d'imprimer un document (abordé dans le livre), de lancer une application, d'ouvrir un fichier, de modifier des paramètres Windows...

VBA est un langage accessible c'est-à-dire qu'il ne requiert pas de connaissances poussées en programmation. La syntaxe a volontairement été simplifiée pour la rendre plus facilement accessible. Les utilisateurs de Microsoft Office ne sont pas forcément des programmeurs, il s'agit plus généralement de professionnels ou étudiants réalisant des activités de gestion.

Ce langage doit donc vous permettre d'aller plus loin dans votre utilisation de Microsoft Office Excel.

2. Pourquoi ce livre ?

Aujourd'hui il existe des centaines de livres sur Excel et VBA et beaucoup se ressemblent même si le style peut varier.

Les manuels sont tous très complets mais optent principalement pour une approche assez informatique de Microsoft Office Excel. Or qui sont les principaux utilisateurs d'Excel ?

Certainement pas les informaticiens ou autres programmeurs. Excel est l'outil qui est le plus utilisé dans le cadre professionnel, pour tout type de profession. Alors certes, les définitions et termes techniques sont nécessaires et seront abordés pour apporter des bases suffisantes pour aller plus loin si vous le souhaitez, mais le plus important c'est de faire un livre pour tous, adapté à votre utilisation.

Ce livre apportera une autre approche, basée principalement sur des exemples extraits de la vie professionnelle. Ils sont d'ailleurs issus directement de certaines expériences. Les termes et le vocabulaire sont volontairement plus accessibles, car vous, lecteurs, n'êtes pas forcément des informaticiens aguerris et les termes propres à l'informatique peuvent vous paraître compliqués.

Toutefois, l'objectif n'est pas de tomber dans la vulgarisation et la simplification à outrance. Il est nécessaire aussi de vous apporter les meilleures bases pour aller encore plus loin. En effet, il est indispensable d'avoir le bon vocabulaire et les bons acquis pour apprendre toujours plus et surtout être en mesure de vous adapter à chaque situation que vous rencontrerez.

3. Quel est le niveau nécessaire pour lire ce livre ?

Vous utilisez Excel, vous connaissez le fonctionnement et vous utilisez probablement quelques formules. Vous vous rendez compte que l'outil est bien plus puissant que l'usage que vous en faites mais vous ne savez pas forcément comment vous y prendre.

Si vous ne connaissez rien à Excel, il est peut-être intéressant de découvrir quelques bases avant d'entamer ce livre.

4. Quel sera votre niveau à la fin de la lecture de ce livre ?

Ce livre vous permet d'approfondir le langage de programmation Visual Basic mais aussi de développer vos compétences avec Microsoft Excel.

Au-delà de lister un certain nombre de fonctions Excel ou VBA, ce livre vous apporte des exemples concrets d'utilisation de ces fonctions dans différents contextes. Il vous apportera aussi une logique dans la résolution des cas pratiques que nous allons étudier.

5. Comment lire ce livre ?

Chaque chapitre correspond à un cas métier composé de deux exercices.

Chaque exercice comprend une présentation de l'objectif, puis l'explication des notions de cours, puis les manipulations permettant de réussir l'exercice.

L'objectif est d'être guidé le plus possible et de mettre tout de suite en pratique les notions de cours présentées.

Chaque exercice a son fichier énoncé (exemple : Enoncé_2-A.xlsx) et son fichier corrigé (exemple : Corrigé_2-A.xlsx). Vous pouvez démarrer chaque exercice à partir du fichier énoncé.

Dans certains cas, un fichier est également présent pour servir d'appui sur les notions de cours.

Ces fichiers sont disponibles en téléchargement sur le site des Editions ENI, www.editions-eni.fr :

- ❖ Accédez au site des Éditions ENI : www.editions-eni.fr
- ❖ Saisissez la référence du livre dans la zone de recherche : SOB16EXCVBA puis cliquez sur OK.
- ❖ Cliquez sur le titre du livre puis sur le lien de téléchargement.

L'approche est donc de mettre le cas métier au centre du livre bien qu'il ne requiert pas de connaissances préalables.

Toutefois, les notions principales d'Excel et Visual Basic Application sont présentes dans les chapitres et la difficulté des exercices est progressive.

Il est préférable de lire les différents chapitres dans l'ordre car ils s'enchaînent logiquement. L'index présent à la fin du livre peut vous permettre à tout moment de revenir vers une définition ou une explication.

6. L'auteur

Je ne suis pas informaticien mais j'ai toujours apprécié l'informatique. Ma spécialisation est la gestion de projet et la gestion d'entreprise, plus particulièrement dans le secteur informatique. Aujourd'hui, je suis consultant en solutions informatiques pour mon propre compte.

J'ai appris Excel au gré de mes expériences, renforcées par quelques cours et appuyées par de nombreuses heures sur les forums et autres supports Microsoft.

Mes expériences professionnelles m'ont permis de beaucoup capitaliser sur mes connaissances d'Excel, car elles furent très diverses et sur différents types de métier. Cela m'a conduit à donner des cours d'Excel notamment à l'Université Paris Dauphine pour former des contrôleurs de gestion. C'est notamment grâce à ce cours que j'ai compris que l'approche non informaticienne était d'une grande utilité pour les étudiants puisque j'avais tendance à simplifier l'utilisation de la programmation. Par la suite, j'ai pu donner diverses formations professionnelles, toujours sur le même ton.

Aujourd'hui, je souhaite partager cette vision à travers ce livre, en espérant surtout que cela va vous permettre de progresser significativement mais également vous donner les bases pour aller plus loin.

B. Avant de commencer

1. Vocabulaire spécifique

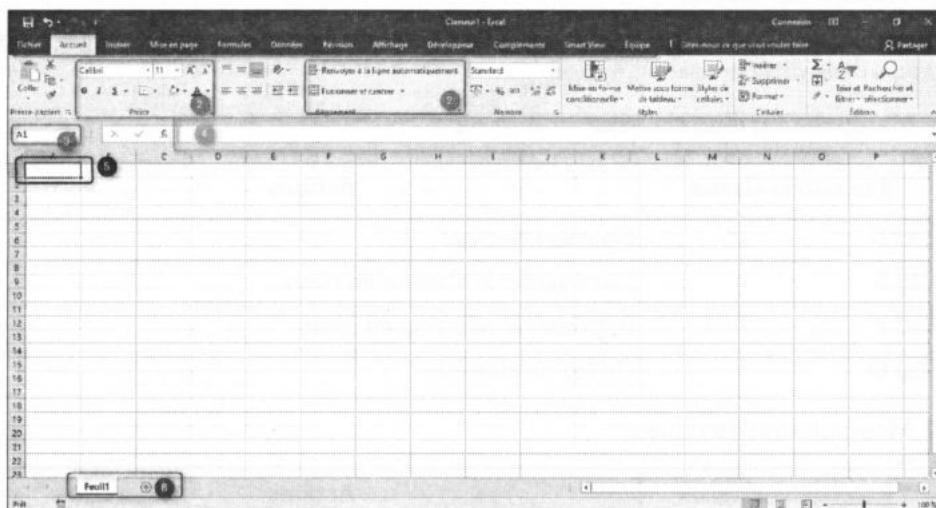
Pour être sûr de ne pas vous perdre lorsque vous lisez ce livre il est utile d'avoir le bon vocabulaire.

Le **fichier** est l'appellation utilisée pour décrire le document contenant l'exemple. Un fichier possède un nom décrivant l'objet (Énoncé, Corrigé, NotionsCours) et une extension correspondant au format du fichier (Excel : xlsx, Excel avec gestion des macros xlsxm).

Le **classeur** est un terme employé pour décrire le fichier Excel. On l'appelle classeur car il contient les feuilles Excel.

Une **feuille Excel** parfois nommée **feuille de calcul** est le tableur sur lequel nous allons travailler. Cette feuille possède un nom qui est inscrit sur l'onglet de la feuille. La plage de cellules est A1:XFD1048576 pour chaque feuille.

Un nouveau classeur s'affiche ainsi :



La fenêtre se décompose ainsi :

- ▶ La zone n°1 correspond aux onglets qui permettent d'accéder aux groupes de fonctionnalités.
- ▶ La zone n°2 correspond aux groupes de fonctionnalités. C'est dans les groupes de fonctionnalités que vous retrouverez les fonctionnalités d'Excel.
- ▶ La zone n°3 correspond à la zone de nom. Dans cette zone, vous pouvez appeler une cellule, une plage ou un tableau soit par sa référence absolue, soit par son nom.

- ▶ La zone n°4 correspond à la barre de formule.
- ▶ La zone n°5 correspond à la cellule sélectionnée.
- ▶ La zone n°6 correspond aux onglets de feuille.

Une cellule est nommée par sa colonne (en lettre) et sa ligne (en nombre). La cellule B3 correspond à la cellule de la ligne 3 et de la colonne 2. Ce nom est appelé référence absolue.

Une cellule a une **valeur** qui correspond à ce qui est affiché à l'écran.

Une cellule peut contenir une **formule** qui détermine la valeur de la cellule. Il est possible de reconnaître une formule car elle débute par le caractère = dans la **barre de formule**. Les formules peuvent faire référence à d'autres cellules et utilisent généralement des fonctions Excel.

Une cellule possède un **format d'affichage** qui correspond au format de restitution de la valeur. Par exemple, une date comme le 01/01/2016 est en fait une valeur numérique 42370 dont le format d'affichage est la date.

2. Raccourcis-clavier

Microsoft Excel possède de nombreux raccourcis-clavier vous permettant de gagner du temps dans votre utilisation d'Excel.

a. Manipulation d'un classeur

Raccourcis-clavier	Actions
Ctrl N	Nouveau classeur
Ctrl S	Sauvegarder le classeur en cours
Ctrl P	Imprimer la sélection en cours
Ctrl O	Ouvrir un classeur

b. Manipulation du tableur

Raccourcis-clavier	Actions
Ctrl A	Sélectionner une plage entière / Si effectuée 2 fois : sélectionner toute la page.
Ctrl G (ou Ctrl 2)	Mettre le texte en gras
Ctrl I (ou Ctrl 3)	Mettre le texte en italique
Ctrl U (ou Ctrl 4)	Mettre le texte en souligné
Ctrl (ou F5)	Atteindre une cellule/plage

Raccourcis-clavier	Actions
F2	Éditer la cellule en cours dans la barre de formule
Ctrl K	Ajouter un lien hypertexte
Ctrl L	Ajouter un tableau
Ctrl ;	Affiche la date du jour dans une cellule
Ctrl :	Affiche l'heure actuelle dans une cellule
Ctrl +	Ajoute une cellule
Ctrl F	Affiche la fenêtre « Rechercher »
Ctrl H	Affiche la fenêtre « Rechercher et Remplacer »
Ctrl 1	Éditer le format d'une cellule
Ctrl X / Ctrl C / Ctrl V	Couper / Copier / Coller
F9	Calcule le classeur : il est possible de désactiver le calcul automatique, cette action permet d'effectuer le calcul en mode manuel.
Ctrl Z	Annuler la dernière action
Ctrl Y	Répéter la dernière tâche

3. Versions de Microsoft Office Excel

La première version de Microsoft Office Excel date de 1985 sous Macintosh et 1987 sous Windows. Depuis, de nombreuses versions ont vu le jour pour s'adapter aux besoins des utilisateurs et aux nouvelles spécifications techniques. VBA est présent sur Excel depuis 1995, par conséquent toutes les versions modernes possèdent VBA. Aujourd'hui, seules quelques postes possèdent encore Excel 2003, la plupart des ordinateurs sont équipés des versions 2007, 2010 et 2013.

a. Excel 2003

Cette version d'Excel possède des feuilles de 65 536 lignes sur 256 colonnes. Les fichiers ont l'extension unique .xls. Cette extension ne différencie pas les fichiers incorporant des macros de ceux n'en incorporant pas. Les fichiers sont parfois lourds et coûteux en termes de mémoire.

b. Excel 2007

Cette version d'Excel possède des feuilles de 1 048 576 lignes sur 16 384 colonnes. Cette version apporte la plus grande modification, le changement des extensions des fichiers.

Type de classeur	Version 2003 et antérieure	Version 2007 et postérieure
Classeur sans macro	.xls	.xlsx
Classeur avec macro	.xls	.xlsm

Cette version d'Excel a connu de grandes difficultés puisqu'elle devait gérer la rétrocompatibilité entre les versions.

c. Excel 2010

La version 2010 apporte de nouvelles fonctionnalités comme les graphiques sparkline, l'apparition du plug-in PowerPivot qui est un outil de gestion des données, des améliorations portant sur la mise en forme conditionnelle ou la création de segments. À cette période, Microsoft a lancé son service Microsoft Office Online qui permet d'utiliser Microsoft Office (dont Excel) à travers un navigateur. La taille des feuilles est inchangée.

d. Excel 2013

La version 2013 d'Excel apporte une refonte visuelle de l'application et permet d'intégrer facilement les documents Office sur OneDrive, le service de stockage en ligne proposé par Microsoft. En termes d'évolutions applicatives, il y a peu de nouvelles fonctionnalités mais beaucoup d'améliorations concernant des fonctionnalités déjà existantes : représentation et analyse graphique, gestion des tableaux et des sources de données, gestion des tableaux croisés dynamiques, amélioration sur PowerPivot. La taille des feuilles est inchangée.

e. Excel 2016

La dernière version de Microsoft Office Excel inclut Power BI, un outil de visualisation et modélisation des données, mais n'apporte pas de nouveauté majeure. On notera en revanche de nombreuses évolutions sur OneDrive et son service applicatif en ligne Microsoft Office Online. Comme pour la version d'Excel 2013, les nouvelles fonctionnalités sont rares, toutefois une attention particulière a été accordée à la facilité d'utilisation des données.

Aujourd'hui Excel n'évolue pas fondamentalement, il s'adapte aux technologies en cours. Il ne faut pas attendre d'évolution majeure puisque le produit est déjà très complet. Les deux axes principaux d'amélioration sont l'intégration du mode Online qui repose sur le développement d'Office 365 (SharePoint, OneDrive, Office Online) et la facilité de représentation des données quelle qu'en soit la source.

f. Office pour Mac

Bien que Microsoft Office Excel soit associé à Microsoft Windows, Office Excel existe aussi pour Mac OS avec la version Excel pour Mac. Autrefois décalées d'un an, les versions Windows et Mac sont désormais synchronisées depuis la version 2013.

Excel pour Mac permet l'utilisation des macros.

L'interface est assez différente, mais vous pourrez suivre facilement le livre avec Excel pour Mac 2013 ou 2016.

g. OpenOffice.org

OpenOffice.org est le concurrent open source de Microsoft Office. Il propose un outil OpenOffice.org Calc qui est également un tableur. L'extension d'OpenOffice est `.odt`, même si l'extension `.xls` peut être compatible. Les extensions `.xlsm` et `.xlsx` ne sont pas compatibles avec OpenOffice Calc.

Calc propose son propre langage de programmation pour l'outil : Open Office BASIC. Il s'agit d'un langage propre à Open Office, qui s'appuie également sur une syntaxe Basic. Il est simple pour un utilisateur de s'adapter entre VBA et OOBASIC mais les deux langages ne sont pas compatibles entre eux.

h. Version du livre

La version que nous allons utiliser dans ce livre est la dernière version disponible au moment de l'écriture Microsoft Office Excel 2016. Le langage VBA n'a plus évolué depuis sa version 7.0 (2010) et cette dernière évolution permet d'adapter la version 6.0 (1998) aux nouvelles technologies. Le langage n'a donc que peu évolué depuis 1998, c'est pourquoi **les exemples présentés dans ce livre couvrent toutes les versions d'Excel.**

Il est cependant possible d'avoir des écarts de présentation entre les différentes versions d'Excel, puisque le design de l'application a évolué. Toutefois, Microsoft a pensé son outil pour qu'il soit accessible quelle que soit la version sur laquelle vous travaillez.

i. Langue du produit Office

La langue du produit a son importance puisqu'en France, deux versions principales de Microsoft Office circulent : la version française et la version anglophone.

Le problème est que les mots clés ne sont pas les mêmes notamment pour les formules (SOMME en français, SUM en anglais) et parfois la syntaxe peut être différente :

Dans un classeur français :

- ▶ Tapez 3,14 : cette cellule sera considérée par défaut comme un champ numérique ;
- ▶ Tapez 3.14 : cette cellule sera considérée par défaut comme un champ texte.

Dans un classeur anglais :

- ▶ Tapez 3,14 : cette cellule sera considérée par défaut comme un champ texte ;
- ▶ Tapez 3.14 : cette cellule sera considérée par défaut comme un champ numérique.

Autre particularité, même si votre version de Microsoft Office Excel est en français, VBA reste un outil en anglais.

Par exemple, pour sommer les nombres 10 et 20 sur Excel, dans la cellule A1, vous aurez la formule suivante :

```
=SOMME(10;20)
```

Par contre la même instruction en VBA se décrira différemment (utilisation de la fonction Sum) :

```
Range("A1").Value = WorksheetFunction.Sum(1,2)
```

Nous avons fait le choix d'utiliser l'application Microsoft Office Excel dans sa version française. Si vous travaillez sur l'outil en anglais, les deux adaptations présentées ci-dessus seront vos points d'attention :

- ▶ Traduire les formules : de nombreux sites proposent la traduction des formules. Vous aurez également la possibilité d'utiliser le support Office qui vous permet de changer la langue et donc d'avoir la formule dans une autre langue : <https://support.office.com/fr-fr/> en français et <https://support.office.com/en-us> en anglais.
- ▶ Changer les points en virgule et vice-versa.

Chapitre 2

Gestion des employés : exploitation de données brutes

A. Formules Excel avancées.....	21
B. Indicateurs clés et partages.....	57

A. Formules Excel avancées

1. Description de l'exemple

a. Présentation de l'exemple

📁 Ouvrez le classeur `Enoncé_2-A.xlsx`.

Cet exemple contient les données des employés d'une société informatique sur les années N et N-1. Cette liste se présente comme un tableau de données contenant la liste de tous les employés avec leurs informations. Il y a 3000 employés dans la société lors de l'année N. Ces 3000 employés ne sont pas forcément les mêmes que ceux de l'année N-1 puisqu'entre les deux années, certains sont partis, d'autres sont arrivés.

La société aimerait avoir plus de visibilité sur sa politique salariale et mieux connaître ses employés. L'objectif est donc d'avoir une vue complète et synthétique de la situation des employés.

Cette société informatique contient les quatre filières suivantes :

- ▶ Technique ;
- ▶ Fonctionnelle ;
- ▶ Business ;
- ▶ Fonctions transverses.

Chaque employé appartient à une filière. Ils possèdent également un grade (allant du grade G1, le plus bas, au grade G12, le plus haut). La combinaison d'un grade et d'une filière définit un poste. Par exemple, un employé qui a un grade G5 dans la filière **fonctionnelle** a le poste de « **Consultant fonctionnel junior** ».

Les employés sont identifiés uniquement grâce à la combinaison des valeurs Nom, Prénom, Date de naissance pour laquelle il n'y a pas de doublons.

L'objectif de cet exemple va être de combiner les différentes données présentes dans le classeur afin d'obtenir des informations plus pertinentes sur les employés et des représentations visuelles des données.

b. Présentation du classeur

Le classeur de l'exemple se décompose en trois feuilles :

- La première feuille **Employé** contient les informations des employés de l'année N.

Colonne	Champ	Description
A	Prénom	Prénom de l'employé
B	Nom	Nom de l'employé
C	Sexe	Sexe de l'employé : H pour homme, F pour femme
D	Date de naissance	Date de naissance au format JJ/MM/AAAA
E	Grade	Grade exprimé par un coefficient compris entre G1 et G12
F	Filière	Filière de l'employé au sein de l'entreprise : fonctionnelle, technique, business, fonctions transverses
G	Salaire brut annuel	Salaire de l'employé exprimé en montant brut annuel.
H	Prime année en cours	Prime reçue par l'employé durant l'année en cours
I	Ancienneté	La période écoulée, exprimée en année, durant laquelle un salarié travaille au sein d'une entreprise jusqu'à l'année N

- La feuille **Filières** comporte le poste associé à un grade et une filière. Cela se présente sous la forme d'un tableau Excel nommé **MatriceGrade**. Voici la représentation du tableau (plage A1:E13).

Grade	Technique	Fonctionnel	Fonctions transverses	Business
G1	Développeur novice	Analyste débutant	Agent	Apprenti business
G2	Développeur junior	Analyste junior	Agent confirmé	Consultant business junior
G3	Développeur	Analyste	Agent opérationnel	Consultant business junior

Grade	Technique	Fonctionnel	Fonctions transverses	Business
G4	Développeur confirmé	Analyste confirmé	Agent expérimenté	Consultant business junior
G5	Consultant technique junior	Consultant fonctionnel junior	Assistant agent de maîtrise	Consultant business
G6	Consultant technique	Consultant fonctionnel	Agent de maîtrise	Consultant business confirmé
G7	Consultant technique sénior	Consultant fonctionnel sénior	Directeur des agents de maîtrise	Consultant business sénior
G8	Expert technique	Chef de projet	Responsable d'entité	Responsable de consultant
G9	Directeur technique	Directeur de projet	Directeur d'agence	Directeur business
G10	Responsable technique Grands comptes	Responsable de Grands comptes	Responsable d'un secteur	Responsable secteur business
G11	Associé filière technique	Associé filière fonctionnelle	Associé filière fonctions transverses	Associé filière business
G12	Chef filière technique	Chef filière fonctionnelle	Chef filière fonctions transverses	Chef business

► La feuille **Employé N-1** permet d'avoir les informations des employés de l'année N-1.

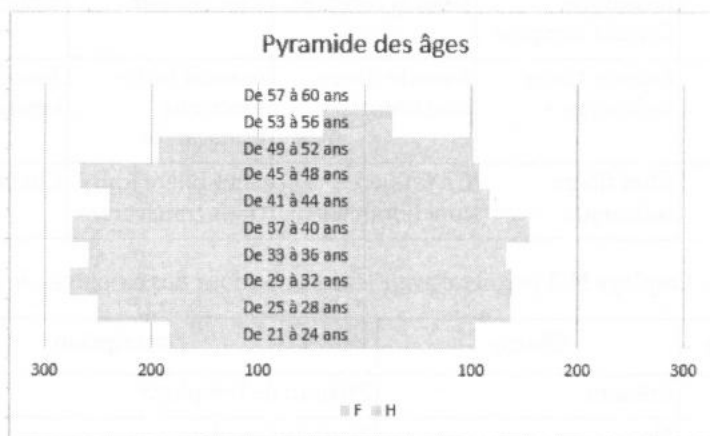
Colonne	Champ	Description
A	Prénom	Prénom de l'employé
B	Nom	Nom de l'employé
C	Sexe	Sexe de l'employé : H pour homme, F pour femme
D	Date de naissance	Date de naissance au format JJ/MM/AAAA
E	Grade	Grade exprimé par un coefficient compris entre G1 et G12

Colonne	Champ	Description
F	Filière	Filière de l'employé au sein de l'entreprise : fonctionnelle, technique, business, fonctions transverses
G	Salaire brut annuel	Salaire brut annuel de l'employé exprimé en montant brut annuel.
H	Ancienneté	La période écoulée, exprimée en année, durant laquelle un salarié travaille au sein d'une entreprise jusqu'à l'année N

c. Fonctionnalités

À partir de ce fichier Excel, les informations suivantes peuvent être calculées :

- ▶ Âge des employés ;
- ▶ Poste des employés ;
- ▶ Afficher les salaires de N-1 sur la même feuille que les salaires de N pour calculer le pourcentage d'augmentation des salaires entre l'année N et l'année N-1 ;
- ▶ Salaires moyens par grade et par filière ;
- ▶ Visualiser une pyramide des âges.

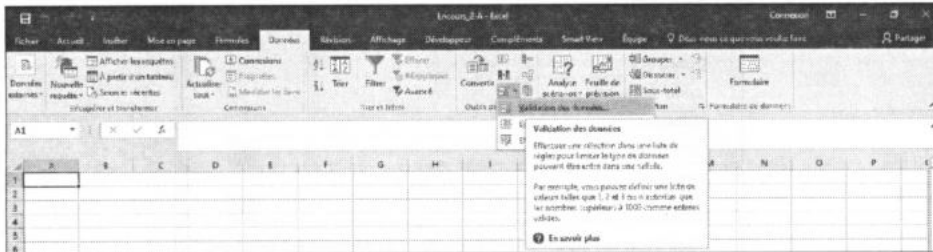


2. Notions de cours

a. Liste déroulante dans une cellule – Validation de données

Une liste déroulante dans une cellule permet de sélectionner une valeur pour la cellule parmi une liste de valeurs proposées. Cette possibilité s'intègre dans la fonctionnalité de Validation de données.

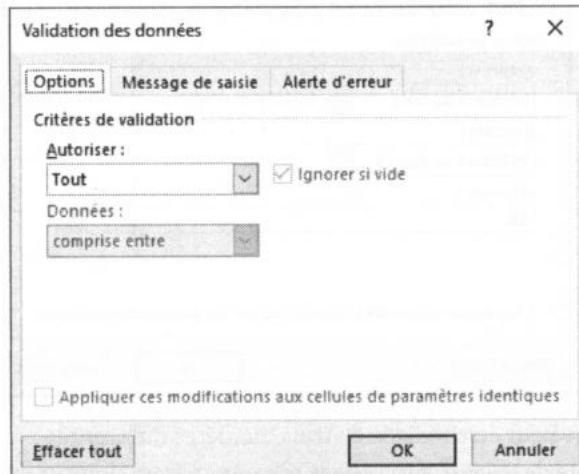
- ❖ Pour accéder à cette fonctionnalité, **sélectionnez une plage de cellules**. Dans l'onglet **Données**, le bouton **Validation de données** est accessible.



La validation de données signifie que les données sont contraintes à une valeur ou à un format spécifique, il s'agit d'une limitation et non d'une aide à l'utilisateur.

Dans la boîte à outils **Validation de données**, trois onglets sont disponibles :

- ▶ Options ;
- ▶ Message de saisie ;
- ▶ Alerte d'erreur.



- L'onglet **Options** permet de définir le format et/ou la valeur attendu(e) dans la cellule. Cela se présente avec trois niveaux : le format attendu, l'opérateur et la valeur attendue. Par exemple, vous pouvez définir un format nombre entier compris entre 100 et 200.

Validation des données

Options Message de saisie Alerte d'erreur

Critères de validation

Autoriser :
Nombre entier Ignorer si vide

Données :
comprise entre

Minimum :
100

Maximum :
200

Appliquer ces modifications aux cellules de paramètres identiques

Effacer tout OK Annuler

Vous pouvez définir la longueur du texte. Par exemple, vous pouvez limiter la longueur du texte à 10 caractères :

Validation des données

Options Message de saisie Alerte d'erreur

Critères de validation

Autoriser :
Longueur du texte Ignorer si vide

Données :
inférieure ou égale à

Maximum :
10

Appliquer ces modifications aux cellules de paramètres identiques

Effacer tout OK Annuler

La liste déroulante peut être rédigée de trois manières différentes :

- Saisie d'une liste de valeurs successives séparées par un point-virgule :
Valeur1 ;Valeur2 ; ... ;ValeurN ;
- Saisie d'une référence de cellules =A1 : A4 qui contient les valeurs ;

- Saisie d'une référence nommée =MaReFeRence qui contient les valeurs.

Validation des données

Options Message de saisie Alerte d'erreur

Critères de validation

Autoriser :
 Liste Ignorer si vide
 Liste déroulante dans la cellule

Données :
 inférieure ou égale à

Source :
 Valeur 1;Valeur 2;Valeur 3; Valeur 4

Appliquer ces modifications aux cellules de paramètres identiques

Effacer tout OK Annuler



Au niveau du champ Source, il n'est pas possible de saisir une référence de cellules dans une autre feuille que celle en cours, toutefois il est possible de contourner cette limitation avec une référence nommée.

- L'onglet **Message de saisie** correspond à une petite fenêtre qui est associée à la cellule. Celle-ci comporte un titre et un message. Elle peut être déclenchée à la sélection de la cellule ou visible en permanence grâce à la case à cocher **Quand la cellule est sélectionnée**.

Validation des données

Options Message de saisie Alerte d'erreur

Quand la cellule est sélectionnée

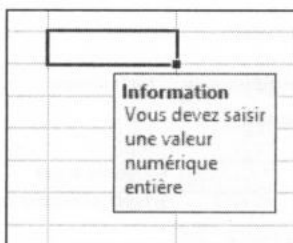
Afficher le message de saisie suivant

Titre :
 Information

Message de saisie :
 Vous devez saisir une valeur numérique entière.

Effacer tout OK Annuler

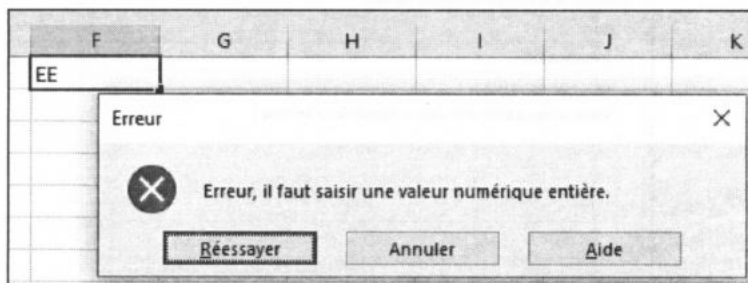
Le résultat est le suivant :



L'onglet **Alerte d'erreur** permet de qualifier le message d'erreur qui apparaîtra si la saisie n'est pas correcte par rapport au format attendu. Si elle n'est pas définie, une alerte par défaut est tout de même présente.



Lorsque vous saisissez une valeur qui ne correspond pas à ce qui est attendu, voici le résultat :



b. Formule de recherche

RECHERCHEH / RECHERCHEV

Les fonctions RECHERCHEV et RECHERCHEH permettent de faire une correspondance entre deux matrices à partir de leur en-tête dans le but de faire remonter une valeur de la matrice cible.

	F	G	H	I	J	K	L
1							
2	A	B	C				A
3	2	3	4				
4							

La formule RECHERCHEH correspond à la recherche horizontale, elle permet de rechercher des éléments dans une table ou une plage par colonne.

La formule RECHERCHEV correspond à la recherche verticale, elle permet de rechercher des éléments dans une table ou une plage par ligne.

La syntaxe est la suivante :

- ▶ RECHERCHEH(valeur_cherchée, table_matrice, no_index_col, [valeur_proche])
- ▶ RECHERCHEV(valeur_cherchée, table_matrice, no_index_ligne, [valeur_proche])

Les arguments de ces fonctions sont :

- ▶ La valeur recherchée (en-tête des matrices commun aux deux tableaux) ;
- ▶ La matrice ou la valeur est recherchée dans la colonne d'en-tête (tableau 1) ;
- ▶ La ligne/colonne dans laquelle la valeur est récupérée au sein de la matrice (tableau 1) ;
- ▶ Valeur booléenne (VRAI ou FAUX) pour savoir si la recherche est approximative ou non.

Exemple pour la cellule L3 :

=RECHERCHEH(L2;F2:H3;2;FAUX)

	F	G	H	I	J	K	L	M
1								
2	A	B	C				A	
3	2	3	4				=RECHERCHEH(L2;F2:H3;2;FAUX)	
4								

La formule RECHERCHEH permet de faire la correspondance entre la matrice de gauche et la matrice de droite. Il s'agit ici de rechercher dans les cellules F2 à H3 la valeur contenue dans la cellule L2 (la valeur "A") et d'afficher comme résultat la valeur de la seconde ligne de la colonne A (donc la valeur 2 contenue en F3).

La cellule L3 renverra ici 2.

```
=RECHERCHEV(D1;A1:B3;2;FAUX)
```

	A	B	C	D	E	F	G	H
1	A	2		B				
2	B	3	=RECHERCHEV(D1;A1:B3;2;FAUX)					
3	C	4	RECHERCHEV(valeur_cherchée; table_matrice; no_index_col; [valeur_proche])					
4								

La cellule B1 renverra ici 3.

INDEX / EQUIV et INDEX-EQUIV

La fonction INDEX permet de rechercher une valeur dans un tableau en fonction de ses coordonnées (ligne, colonne).

Les arguments de la fonction INDEX sont les suivants :

```
=INDEX(Matrice ; ligne ; [colonne])
```

Exemple

```
=INDEX(A1:D4;3;2)
```

	A	B	C	D
1	Ligne 1 Colonne 1	Ligne 1 Colonne 2	Ligne 1 Colonne 3	Ligne 1 Colonne 4
2	Ligne 2 Colonne 1	Ligne 2 Colonne 2	Ligne 2 Colonne 3	Ligne 2 Colonne 4
3	Ligne 3 Colonne 1	Ligne 3 Colonne 2	Ligne 3 Colonne 3	Ligne 3 Colonne 4
4	Ligne 4 Colonne 1	Ligne 4 Colonne 2	Ligne 4 Colonne 3	Ligne 4 Colonne 4
5				
6	=INDEX(A1:D4;3;2)			
7				

La cellule A6 renverra ici Ligne 3 Colonne 2.

La fonction EQUIV permet de connaître la position d'une valeur donnée au sein d'une ligne ou d'une colonne.

Les arguments de la fonction EQUIV sont les suivants :

```
=EQUIV(Valeur_recherchée ; Matrice ; Correspondance_exacte)
```

L'argument **correspondance exacte** peut prendre les valeurs suivantes :

- ▶ Correspondance exacte (0) ;
- ▶ Valeur inférieure(1) ;
- ▶ Valeur supérieure (-1).

Exemple :

	A	B	C	D	E	F	G
1	Etudiant	Base de données	Excel	Access	Anglais	Programmation	Gestion de projet
2	Thierry	12	14	12	14	14	10
3	Paul	13	12	15	11	16	16
4	Louis	11	16	10	10	9	11
5	Cécile	14	17	16	15	15	18
6	Marie	11	12	11	11	12	14
7	Claire	15	14	14	12	14	12
8							
9	Eleve	Matière	Note				
10	Paul		=EQUIV(A10;A1:A7;0)				
11							

La formule =EQUIV(A10;A1:A7;0) situé en C10 renverra ici 3 pour l'élève Paul.

La combinaison des fonctions INDEX et EQUIV peut être très intéressante car cela permet de rechercher une valeur dans un tableau à double entrée.

En effet, EQUIV renvoie une position au sein d'une ligne et d'une colonne, et INDEX se sert de coordonnées (ligne, colonne) pour obtenir une valeur dans une matrice, par conséquent, les deux peuvent être combinées :

Expression :

=INDEX(Matrice ;EQUIV(valeur ligne) ; EQUIV(valeur colonne))

Exemple :

	A	B	C	D	E	F	G	H
1	Etudiant	Base de données	Excel	Access	Anglais	Programmation	Gestion de projet	
2	Thierry	12	14	12	14	14	10	
3	Paul	13	12	15	11	16	16	
4	Louis	11	16	10	10	9	11	
5	Cécile	14	17	16	15	15	18	
6	Marie	11	12	11	11	12	14	
7	Claire	15	14	14	12	14	12	
8								
9	Eleve	Matière	Note					
10	Cécile	Excel	=INDEX(A1:G7;EQUIV(A10;A1:A7;0);EQUIV(B10;A1:G1;0))					

La formule =INDEX(A1:G7;EQUIV(A10;A1:A7;0);EQUIV(B10;A1:G1;0)) situé en C10 renverra 17 pour la note de Cécile sur Excel.



Attention, lorsque vous combinez les formules INDEX et EQUIV, les matrices des deux fonctions doivent avoir la même dimension. Si la matrice INDEX a la dimension 4L x 3C, la matrice EQUIV en ligne doit avoir la dimension 4L x 1C, la matrice EQUIV en colonne doit avoir la dimension 1L x 3C.

c. Structure conditionnelle sur Excel : conditions et SI

Pour tester une condition il suffit d'ouvrir une formule avec le signe = puis de tester la condition avec un opérateur :

=A1=A2 testera l'égalité entre les cellules A1 et A2 et renverra VRAI ou FAUX en fonction du résultat. Il est possible de le faire de manière plus élaborée :

=EQUIV(G7;G10:G20;0)=OU(1;4) teste si la valeur contenue en G7 est en première ou quatrième position dans la plage G10:G20.

La fonction SI permet de tester une condition et de lui attribuer un résultat si la condition est vraie ainsi qu'un résultat si la condition est fausse.

Sa structure est la suivante :

=SI(CONDITION ; Valeur si vrai ; valeur si faux)

Exemple :

=SI(A1>2;2;A1)

Notons qu'il est possible d'imbriquer plusieurs SI :

=SI(A>B;40;SI(A<B;50;45))

d. Formule conditionnelle

Pour comprendre les formules conditionnelles, voici un exemple basique :

	A	B
1	11	H
2	12	F
3	10	H
4	8	H
5	7	F
6	12	H
7	19	F
8	1	H
9	11	H

	A	B
10	8	F

SOMME.SI

La fonction `SOMME.SI` permet de faire la somme des valeurs d'une plage respectant la condition posée :

`=SOMME.SI(plage ;critère ;[somme_plage])`

L'élément de la plage sera sommé s'il respecte le critère : par exemple sommer les valeurs si elles sont supérieures à 10. Le critère est généralement écrit entre guillemets, mais peut être interprété à partir d'une cellule.

Pour définir le critère supérieur à 10, plusieurs expressions sont possibles dans la formule :

- ▶ `=SOMME.SI(A1:A10; ">10")`
- ▶ `=SOMME.SI(A1:A10; ">"&C4)` : la cellule C4 possédant la valeur 10, ce qui permet d'avoir un critère variable.
- ▶ `=SOMME.SI(A1:A10; C4)` : la cellule C4 possédant la valeur : ">10".

Par conséquent, toutes les valeurs de la plage A1:A10 supérieures à 10 seront sommées, le résultat est 65.

La plage sommée peut être différente de celle du critère :

`=SOMME.SI(B1:B10; "H"; A1:A10)` : toutes les valeurs de la plage A1:A10 seront sommés si la valeur de la cellule correspondante dans la plage B1:B10 est égal à H.

Le résultat est de 53.

NB.SI

La fonction `NB.SI` permet de compter les éléments d'une plage en fonction d'une condition posée :

`=NB.SI(plage_critere; critere)`

L'élément de la plage de critère sera compté s'il respecte le critère : par exemple compter les valeurs si elles sont égales à H.

`=NB.SI(B1:B10; "H")` : sera égal à 6.

SOMME.SI.ENS

Le principe de la fonction `SOMME.SI.ENS` est le même que `SOMME.SI`, mais avec plusieurs critères. La plage sommée doit en revanche être clairement exprimée et n'est pas facultative comme dans un `SOMME.SI` classique :

`=SOMME.SI.ENS(plage_sommée; plage_critere1; critere1; ... ;
plage_critere_N ; critere_N)`

Pour sommer les éléments de la plage A1:A10 qui sont inférieurs à 10 et dont la valeur de la plage B1:B10 est égale à F, la formule est la suivante :

```
=SOMME.SI.ENS(A1:A10;A1:A10;"<10";B1:B10;"F")
```

Le résultat est 15.

NB.SI.ENS

Le principe de la fonction NB.SI.ENS est le même que NB.SI, avec la possibilité de saisir plusieurs critères successivement. La comptabilisation des éléments porte sur la taille absolue de la plage et non sur l'ensemble des plages. L'ordre des arguments est toujours le même par rapport à la fonction NB.SI.

```
=NB.SI.ENS(plage_criterel;criterel;...;plage_criteren ;criteren)
```

Pour compter le nombre d'éléments qui ont une valeur supérieure à 10 dans la plage A1:A10 et la valeur H dans la plage B1:B10, la formule est la suivante :

```
=NB.SI.ENS(A1:A10;">10";B1:B10;"H")
```

Le résultat est 3.

e. Gestion des cas d'erreur

La gestion des cas d'erreur correspond au traitement des formules qui provoquent une erreur en guise de résultat.

En effet, de nombreuses formules sont liées à d'autres formules qui peuvent envoyer des résultats divers et variés. De ce fait, il n'est pas possible de couvrir tous les cas et d'anticiper toutes les erreurs possibles. Un cas très courant est l'absence de valeur trouvée dans une recherche, ce qui signifie que l'élément est absent de la matrice. Deux formules existent pour gérer les cas d'erreur :

- ▶ La formule ESTERR(Valeur) renvoie VRAI en cas d'erreur ou FAUX si absence d'erreur sauf pour l'erreur de type #N/A qui n'est pas considérée comme une erreur.
- ▶ La formule ESTERREUR(Valeur) renvoie VRAI en cas d'erreur ou FAUX.

La gestion d'une erreur sur une formule de recherche se présente donc ainsi :

```
=SI(ESTERREUR(RECHERCHEV(A1;Tableau1;3;FAUX));"Valeur non présente";RECHERCHEV(A1;Tableau1;3;FAUX))
```

Dans ce cas, si la recherche ne renvoie pas de résultat, il y aura un message pour signifier à l'utilisateur que la valeur n'est pas présente.



Dans ce cas précis, l'utilisation de la formule ESTERR n'aurait pas été possible car l'erreur retournée en cas d'absence d'un élément dans un tableau est #N/A.





f. Calcul matriciel

Le calcul matriciel permet de manipuler des plages comme des matrices indissociables. Les formules appliquées sur la matrice porteront donc sur l'ensemble des valeurs de la matrice.

Le calcul matriciel est caractérisé par des symboles { } affichés dans la barre de formule non modifiables par l'utilisateur :




{=A1:A4*B1:B4}

Pour effectuer un calcul matriciel, les étapes sont les suivantes :

- ✎ Rédigez la formule.
- ✎ Validez la formule : au lieu d'appuyer sur la touche , appuyez simultanément sur les touches   .

Un premier exemple pertinent est le calcul Prix/Quantité :

	A	B	C	D
1	10 €	21	A1	
2	12 €	41	A2	
3	13 €	70	B1	
4	14 €	19	B2	
5				

- ▶ La colonne A (A1:A4) contient les prix ;
- ▶ La colonne B (B1:B4) contient les quantités ;
- ▶ La colonne C (C1:C4) contient le code produit ;
- ▶ La colonne D (D1:D4) contient formule = A1:A4* B1:B4 validée par la combinaison des touches   .

Le résultat obtenu est le suivant :

	A	B	C	D
1	10 €	21	A1	210
2	12 €	41	A2	492
3	13 €	70	B1	910
4	14 €	19	B2	266

Le calcul matriciel permet d'effectuer plus facilement des opérations avec des conditions portant sur les valeurs de la matrice.

Il permet aussi de calculer dans une cellule une somme de matrice via la fonction SOMME. Cette somme peut inclure des critères.

Par exemple, pour faire le total des prix par les quantités des produits dont le code commence par B, la formule suivante est utilisée :

=SOMME (A1:A4*B1:B4* (GAUCHE (C1:C4;1) ="B")) validée par   .

L'affichage de cette formule se fera entre crochets dans la barre de formule :

{=SOMME (A1:A4*B1:B4* (GAUCHE (C1:C4;1) ="B")) }



Les plages sommées et plages de critères doivent toutes avoir la même dimension.

3. Réalisation de l'exemple

Cet exemple va nous permettre de synthétiser les informations contenues dans la feuille de calcul **Employé**. Nous allons utiliser les informations présentes dans les feuilles **EmployéN-1** et **Filières** pour agrémenter les données présentes dans la feuille **Employé**.

	A	B	C	D	E	F	G	H	I	J	K
1	Grade	CC				Age Min	Age Max	Libellé	H	F	
2	Filière	Technique				21	24	De 21 à 24 ans	136	33	
3	Méthode	Scrimo des salaires	Nombre d'employés	Moyenne des salaires		25	32	De 25 à 32 ans	134	129	
4	Formule conditionnelle	1161300	26	44605,38462		29	32	De 29 à 32 ans	177	144	
5	Matrice	1161300	26	44605,38462		33	36	De 33 à 36 ans	194	182	
6						37	40	De 37 à 40 ans	179	158	
7						41	44	De 41 à 44 ans	241	113	
8						45	48	De 45 à 48 ans	277	115	
9						49	52	De 49 à 52 ans	202	107	
10						53	56	De 53 à 56 ans	43	23	
11						57	60	De 57 à 60 ans	5	5	

De 57 à 60 ans	De 53 à 56 ans	De 49 à 52 ans	De 45 à 48 ans	De 41 à 44 ans	De 37 à 40 ans	De 33 à 36 ans	De 29 à 32 ans	De 25 à 32 ans	De 21 à 24 ans
5	43	202	277	241	179	194	177	134	136

🔗 Ouvrez le fichier **Enoncé_2-A.xls** qui contient les feuilles **Employé**, **EmployéN-1** et **Filières**. Les principales actions se déroulent sur la feuille **EmployéN-1**.

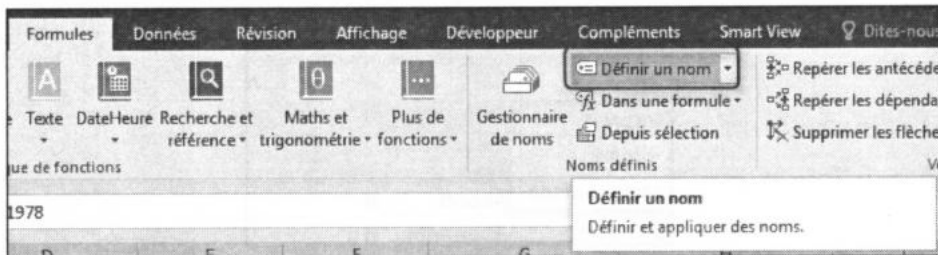
a. Nommage des plages

Le nommage des plages consiste à donner un nom à une plage de cellules. Cela permet d'appeler cette plage par le nom donné plutôt que par ses cellules.

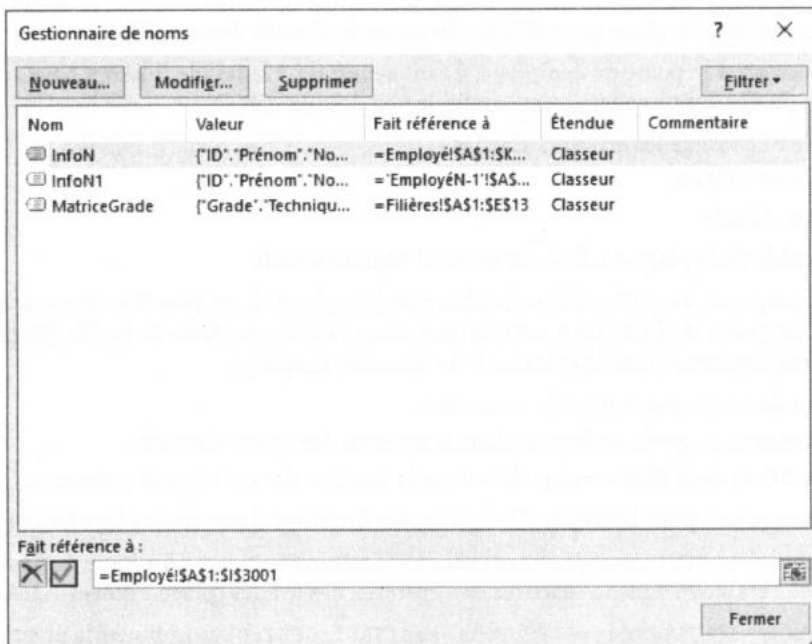
Afin de faciliter l'usage de formules au sein de cet exemple, les plages vont être nommées. Les tableaux contenant les informations des années N (feuille **Employé**) et N-1 (feuille **EmployéN-1**) vont être renommés :

🔗 Sur la feuille **Employé**, sélectionnez la cellule **A1** qui se trouve dans le tableau.

- Appuyez simultanément sur les touches **Ctrl** et **A** pour sélectionner l'ensemble du tableau en cours. Vous remarquerez que la plage A1:I3001 contient l'ensemble des données de la feuille **Employé**.
- Sélectionnez l'onglet **Formules** puis accédez au groupe **Noms définis** et cliquez sur **Définir un nom**.



- Accédez à la fenêtre de saisie d'un nom pour la plage de données sélectionnées :
Saisissez le nom **InfoN** pour la plage A1:I3001 de la feuille **Employé**.
Saisissez le nom **InfoN1** pour la plage A1:H3001 de la feuille **Employé-1**.
- Visualisez l'ajout des noms en cliquant sur **Gestionnaire des noms** de l'onglet **Formules** :



Les données sont ainsi nommées et il est possible de les appeler à tout moment dans le classeur avec ce nom.



Pour sélectionner le tableau, il est possible de le choisir dans la zone contenant le nom de la plage.

B7		X ✓ fx A	
InfoN	B	C	
InfoN1	Nom		Sexe
MatriceGrade	Abbott	H	
3	Robert	Abernethy	H

b. Fonctions de recherche : afficher le poste et le salaire de l'année précédente

Afficher le poste de l'employé en fonction de son grade et sa filière

La colonne J de la feuille **Employé** contiendra le poste.

☛ Commencez par saisir l'en-tête **Poste** dans la cellule J1.

Sur la plage J2:J3000, nous allons saisir une formule unique à appliquer sur l'ensemble de la plage pour afficher le poste de chacun des employés.

Pour récupérer le poste de l'employé, il faut se référer à la feuille **Filières** pour constater que les postes sont exposés selon un tableau à double entrée.

Ce tableau possède les en-têtes suivants :

- ▶ Colonne : Filière
- ▶ Ligne : Grade

L'ensemble de la plage A1:E13 est nommé **MatriceGrade**.

Par conséquent, avec le grade et la filière de l'employé, il est possible de récupérer son poste. Le grade de l'employé est contenu dans la colonne E de la feuille **Employé**, sa filière est contenue dans la colonne F de la feuille **Employé**.

La formule **EQUIV** permettra de connaître :

- ▶ la position du grade recherché dans la matrice des lignes d'en-tête,
- ▶ la position de la filière recherchée dans la matrice des en-têtes de colonnes.

Par conséquent, pour la cellule J2 de la feuille **Employé** dans notre plage **InfoN** :

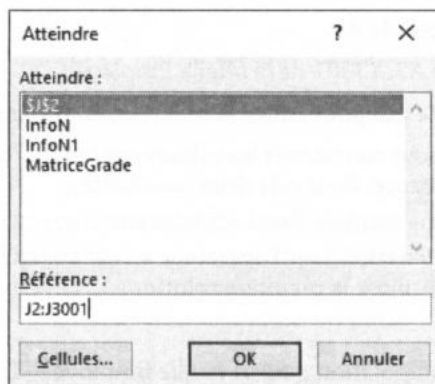
La formule `=EQUIV(Employé!E2;Filières!A1:A13;0)` va indiquer la position du grade de l'employé dans la matrices des en-têtes des grades (page **Filières!A1:A13**).

La formule `=EQUIV(Employé!F2;Filières!A1:E1;0)` va indiquer la position de la filière de l'employé dans la matrices des en-têtes des filières (page **Filières!A1:E1**).

En récupérant la ligne, la colonne et bien entendu la matrice initiale (matrice MatriceGrade), il est possible de récupérer le poste avec la formule INDEX(matrice;ligne;colonne).

Voici la manipulation à réaliser :

- ☞ Positionnez-vous sur la feuille **Employé**.
- ☞ Appuyez simultanément sur les touches **Ctrl** et **T**.
La fenêtre permettant d'atteindre les cellules apparaît.
- ☞ Saisissez la référence **J2:J3001** puis cliquez sur **OK**.



La plage **J2:J3001** de la feuille **Employé** est sélectionnée.

- ☞ Appuyez sur la touche **F2** pour éditer les valeurs de la plage, puis saisissez la formule suivante :

```
=INDEX(MatriceGrade ;EQUIV(Employé!E2;Filières!$A$1:$A$13;0);  
EQUIV(Employé!F2;Filières!$A$1:$E$1;0))
```

- ☞ Appuyez simultanément sur les touches **Ctrl** et **↵** pour appliquer la formule à l'ensemble de la plage.

Notez que l'utilisation des \$ pour figer la colonne et la ligne est très utile puisque cela permet d'effectuer une copie en série sur les lignes en dessous.

Récupérer le salaire d'un employé pour l'année précédente

Nous allons ajouter le salaire de l'année précédente d'un employé dans la feuille **Employé**.

Pour récupérer le salaire d'un employé de l'année précédente, il faut identifier la ligne contenant les informations de l'employé pendant l'année N-1, c'est-à-dire sur la feuille **EmployéN-1**. Pour faire la correspondance entre l'année N et l'année N-1 il est nécessaire d'avoir un en-tête commun qui ne contient pas de doublons entre les deux tableaux.

Le problème rencontré ici est qu'il n'y a pas d'en-tête commun entre les deux tableaux. La seule garantie d'avoir un en-tête commun n'ayant pas de doublons serait de créer une colonne identifiant uniquement chacun des employés.

Comment faire le lien entre les deux tableaux ?

La solution consiste à insérer une nouvelle colonne en début de chaque feuille et à utiliser un identifiant commun qui est la combinaison des colonnes Prénom, Nom et Date de naissance :

- ❖ Insérez une nouvelle colonne en début de la feuille **Employé** : Faites un clic droit sur l'en-tête de la colonne **A** et choisissez l'option **Insertion**.
- ❖ Saisissez **ID** dans la cellule **A1**.
- ❖ Sélectionnez la plage **A2:A3001** de la feuille **Employé**.
- ❖ Appuyez sur la touche **F2** pour éditer la cellule en cours à savoir la cellule **A2**.
- ❖ Saisissez la formule pour concaténer les valeurs contenues dans les colonnes Prénom, Nom et Date de naissance. Pour cela deux possibilités :
 - ▶ Solution 1 : avec une formule Excel `=CONCATENER (B2 ; C2 ; E2) .`
 - ▶ Solution 2 : Concaténation avec l'opérateur `&` : `=B2&C2&E2 .`
 Pour cet exemple, utilisez la première solution.
- ❖ Validez par **Ctrl** **↵**.
- ❖ Répétez la même manipulation pour la feuille **EmployéN-1**.



*Les colonnes ont été décalées sur la droite par conséquent, le prénom se trouve en colonne B, le nom en colonne C et la date de naissance en colonne E. Modifiez les plages nommées **InfoN1** et **InfoN** pour qu'elles aient la bonne dimension.*

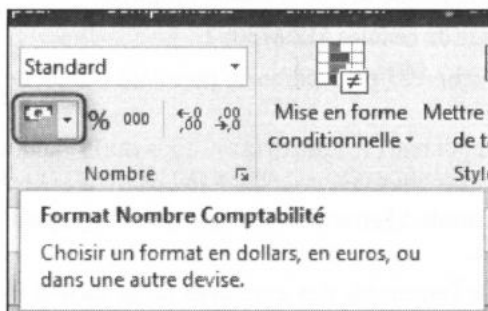


Une fois que les deux tableaux ont un en-tête commun, il est possible d'effectuer une recherche via la formule RECHERCHEV :

- ▶ La valeur recherchée est la colonne A de la feuille **Employé** ;
- ▶ La matrice de recherche est l'ensemble du tableau correspondant aux informations de la feuille **EmployéN-1** ;
- ▶ La colonne recherchée est la colonne H (correspondant à la colonne 8, celle contenant les salaires) ;
- ▶ La correspondance doit être exacte et non approximative.

Pour cela :

- ❖ Sélectionnez la cellule L1 et renommez-la en **Salaire de l'année précédente**.
- ❖ Sélectionnez l'ensemble des cellules de la colonne L jusqu'en bas du tableau (L2:L3001).
- ❖ Éditez la première cellule du tableau en appuyant sur la touche **F2** : vous serez ainsi positionné sur la cellule L2 en édition.
- ❖ Saisissez la formule de recherche suivante : `=RECHERCHEV (A2; InfoN1; 8; FAUX)`
- ❖ Appuyez simultanément sur les touches **Ctrl** et **↵** pour appliquer cette formule à l'ensemble des cellules sélectionnées.
- ❖ Appliquez le format monétaire à l'ensemble de la colonne.



Vous noterez que certaines formules apparaissent en erreur (#N/A), il s'agit des cas où il n'y a pas de correspondance entre les deux tableaux, c'est-à-dire les employés présents en année N mais absents en année N-1. Cette erreur sera corrigée dans une prochaine section.

c. Gestion des erreurs et formules conditionnelles

Formule conditionnelle : calcul de l'âge

Nous allons ajouter l'âge de l'employé dans la colonne M de la feuille **Employé**.

Le calcul de l'âge est simple, il s'agit de faire la différence entre l'année en cours et l'année de naissance en prenant en compte si l'anniversaire est passé ou à venir.

Sur Excel, le calcul se fait de la même manière : si l'anniversaire de l'employé a eu lieu, alors son âge correspond à la différence entre l'année en cours et l'année de naissance, par contre si l'anniversaire de l'employé n'est pas passé, alors son âge est égal à la différence entre l'année en cours et l'année de naissance à laquelle on soustrait un (l'année en cours).

Pour déterminer si l'anniversaire a eu lieu, il suffit de convertir la date de naissance de la personne en son anniversaire de l'année actuelle (pour la cellule M2) :

```
=DATE (ANNEE (AUJOURDHUI ( ) ) ; MOIS (E2) ; JOUR (E2) ) > AUJOURDHUI ( )
```

Cette formule renvoie **VRAI** si l'anniversaire est supérieur à la date du jour, et **FAUX** sinon.

En intégrant cette condition dans une formule conditionnelle, nous obtenons l'âge.

Pour afficher l'âge :

- ❖ Saisissez **Âge** dans la cellule **M1** de la feuille **Employé**.
- ❖ Sélectionnez la plage de cellules **M2:M3001**.
- ❖ Appuyez sur la touche **F2** pour éditer la première cellule de la plage et intégrez la formule suivante :

```
=SI (DATE (ANNEE (AUJOURDHUI ( ) ) ; MOIS (E2) ; JOUR (E2) ) > AUJOURDHUI ( ) ;  
ANNEE (AUJOURDHUI ( ) ) - ANNEE (E2) - 1 ; ANNEE (AUJOURDHUI ( ) ) - ANNEE (E2) )
```

- ❖ Appliquez cette formule à l'ensemble du tableau en appuyant simultanément sur les touches **Ctrl** et **↵**.

Vous obtenez l'âge de l'ensemble des employés de la société dans la colonne M pour l'année en cours.

Formule conditionnelle : calcul de l'augmentation

Nous allons afficher l'augmentation de salaire entre l'année N-1 et l'année N dans la colonne N de la feuille **Employé**.

Le calcul de l'augmentation correspond à la comparaison entre le salaire de N et le salaire de N-1 avec l'application de la formule classique du taux d'évolution :

$$\frac{\text{Valeur d'arrivée} - \text{Valeur de base}}{\text{Valeur de base}}$$

Le salaire de l'année N est contenu dans la colonne H de la feuille **Employé**, le salaire de l'année N-1 est contenu dans la colonne L de la feuille **Employé**.

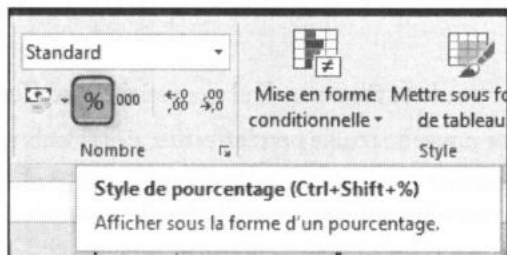
Pour modéliser cette formule sous Excel, cela représente :

= (H2-L2) / L2 associée à l'application d'un format pourcentage.

Cette formule n'est pas suffisante en l'état puisque tous les employés n'étaient pas présents l'année précédente, par conséquent la formule contenue dans la colonne L de la feuille **Employé** retourne parfois une erreur. Il est donc nécessaire de s'assurer que le salaire précédent contient bien une valeur numérique avec la formule ESTNUM. En associant cette formule à une structure conditionnelle, tous nos cas semblent corrects et il est possible d'avoir le pourcentage d'augmentation des employés présents en N-1 et l'information 0 pour les employés absents en N-1.

Pour avoir le pourcentage d'augmentation :

- ❖ Nommez la cellule N1 **Pourcentage d'augmentation**.
- ❖ Sélectionnez la plage N2:N3001.
- ❖ Appuyez sur la touche **F2** pour éditer la plage, et saisissez la formule suivante :
=SI(ESTNUM(L2); (H2-L2) / L2; 0)
- ❖ Appliquez cette formule à l'ensemble du tableau en appuyant simultanément sur les touches **Ctrl** **⇩**.
- ❖ Appliquez le format **Pourcentage** à l'ensemble de la colonne :



d. Gérer l'erreur sur le salaire des nouveaux arrivants

Vous avez pu constater que dans la colonne L, certains cas apparaissent en erreur. L'objectif est de gérer ces cas d'erreur.

En effet, lorsqu'un employé ne fait pas partie de l'entreprise en N-1, le salaire de l'année précédente ne peut être correctement remonté dans la feuille **Employé**. Le cas échéant, cela affiche une erreur de type #N/A signifiant que la recherche n'a pas trouvé de résultat.

Comment ne pas avoir de message d'erreur ?

Les fonctions ESTERR et ESTERREUR permettent de renvoyer VRAI si une erreur est rencontrée dans la fonction présente en argument :

- ▶ =ESTERREUR(notre_fonction) renvoie VRAI si notre fonction renvoie une erreur ;
- ▶ =SI(ESTERREUR(notre_fonction);message_erreur;notre_fonction)



La fonction ESTERR ne considère pas l'erreur #N/A comme une erreur (il s'agit en réalité d'un cas non couvert, pas d'une erreur), par conséquent dans le cas de la gestion des erreurs des RECHERCHEV ou RECHERCHEH, il est plus intéressant d'utiliser la fonction ESTERREUR.

La formule conditionnelle est la suivante :

- ❖ Sélectionnez la plage L2:L3001.
- ❖ Appuyez sur la touche **F2** pour éditer la plage, et saisissez la formule suivante :
=SI(ESTERREUR(RECHERCHEV(A2;InfoN1;8;FAUX));"Nouvel arrivant";RECHERCHEV(A2;InfoN1;8;FAUX))
- ❖ Appliquez cette formule à l'ensemble du tableau en appuyant simultanément sur les touches **Ctrl** **⇧**.

Dans ce cas-là, la formule RECHERCHEV est testée via la fonction ESTERREUR, en cas d'erreur, la cellule de la colonne L affichera le texte : **Nouvel arrivant**, sinon, la formule sera appliquée.

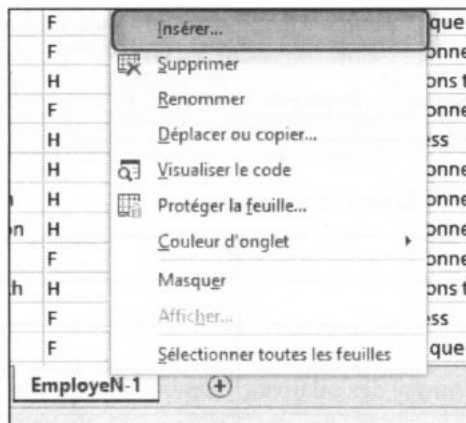
e. Salaire moyen par grade et filière : calcul matriciel contre formule conditionnelle

Nous allons créer une nouvelle feuille pour effectuer des calculs spécifiques.

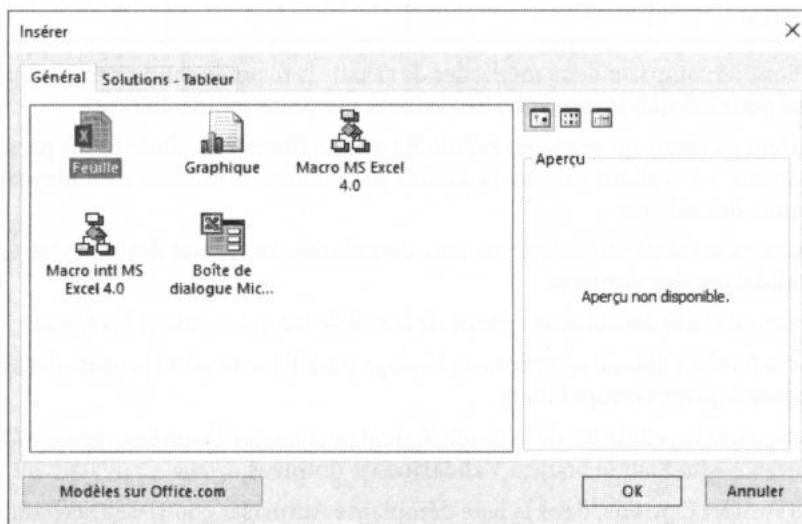
L'exemple suivant consiste à calculer le salaire moyen par grade et filière. Pour cela, il sera plus simple de travailler sur une nouvelle feuille nommée Calcul. Pour créer une nouvelle feuille :

- ❖ Faites un clic droit sur l'onglet de la feuille EmployéN-1.

☞ Cliquez sur **Insérer...** :



☞ Choisissez d'insérer une Feuille dans l'écran suivant :

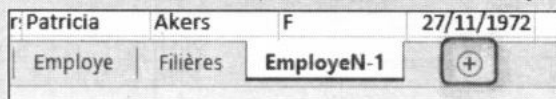


☞ Faites un clic droit sur l'onglet de la feuille tout juste insérée.

☞ Cliquez sur **Renommer** puis saisissez **Calcul**.



Pour insérer une feuille après la dernière feuille, cliquez sur le bouton +.:



☞ Remplissez la feuille de la manière suivante :

	A	B	C	D
1	Grade			
2	Filière			
3	Méthode	Somme des salaires	Nombre d'employés	Moyenne des salaires
4	Formule conditionnelle			
5	Matrice			

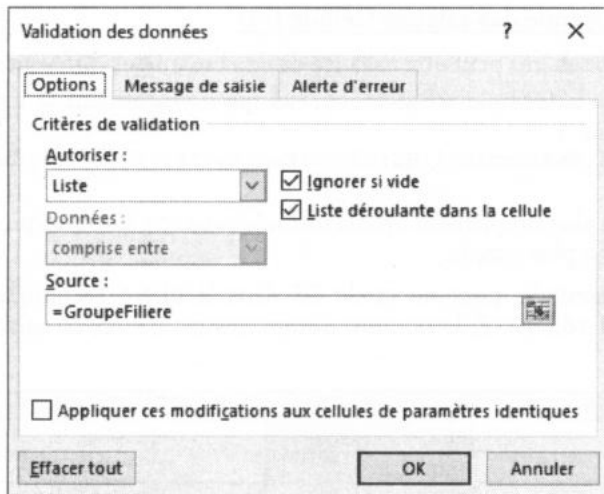
Nous allons ici comparer deux méthodes de calcul : la formule conditionnelle vs le calcul matriciel pour calculer la moyenne des salaires par poste (grade/filières).

L'utilisateur va saisir un grade en cellule B1 et une filière en cellule B2. À partir de ces informations, nous allons calculer la somme des salaires, le nombre d'employés et enfin la moyenne des salaires.

La filière sera attribuée à l'aide d'une liste déroulante. Le format des données est défini via la **Validation des données**.

Pour créer une liste déroulante à partir de la cellule B2 qui contient les filières :

- ☞ Dans la feuille **Filières**, sélectionnez la plage B1:E1 contenant l'intitulé des filières et nommez la plage **GroupeFiliere**.
- ☞ Sélectionnez la cellule B2 de la feuille **Calcul** puis onglet **Données** - groupe **Outils de données**, cliquez sur le bouton **Validation de données**.
- ☞ Dans l'onglet **Options**, dans la liste déroulante **Autoriser** choisissez le champ **Liste**.
- ☞ Pour les valeurs de la liste, saisissez au niveau du champ **Source** =**GroupeFiliere**



Pour terminer :

- ☞ Saisissez un grade en B1 (exemple : G6).
- ☞ Choisissez une filière dans la liste déroulante de B2 (exemple : Technique).

	A	B	C	D
1	Grade			
2	Filière			
3	Méthode	Technique Fonctionnel Fonctions transverses Business	Nombre d'employés	Moyenne des salaires
4	Formule conditionnelle			
5	Matrice			

Avec les formules conditionnelles

Calcul de la somme des salaires en fonction du grade (cellule B4)

Utilisez la formule conditionnelle `SOMME.SI` pour sommer les salaires s'ils respectent les conditions définies en B1 pour le grade (colonne H dans la feuille `Employé`) et en B2 pour la filière (colonne G dans la feuille `Employé`) :

- ☞ Saisissez dans la cellule B4 la formule suivante :

```
=SOMME.SI.ENS (Employé!H2:H3001;Employé!F2:F3001;B1;Employé!G2:G3001;B2)
```

Calcul du nombre d'employés en fonction du grade et de la filière (cellule C4)

Pour le nombre d'employés remplissant les critères définis, il faut utiliser la formule `NB.SI` avec les mêmes critères que ci-dessus. La cellule C4 prend la valeur suivante :

```
=NB.SI.ENS (Employé!F2:F3001;B1;Employé!G2:G3001;B2)
```

Calcul de la moyenne des salaires (cellule D4)

La moyenne des salaires peut être calculée de deux manières différentes :

- Utilisation de la formule MOYENNE.SI.ENS qui a exactement la même syntaxe que SOMME.SI.ENS :
=MOYENNE.SI.ENS (Employé!H2:H3001;Employé!F2:F3001;B1;Employé!G2:G3001;B2) ;
- Ou beaucoup plus simplement avec la formule : =B4/C4. Comme quoi sur Excel, il faut parfois aller au plus simple.

Résultat : par exemple, pour un grade G6 dans la filière Technique, la somme des salaires est de 1 163 300 €, le nombre d'employés est de 26. Le salaire moyen est de 44 665.38 €.

	A	B	C	D
1	Grade	G6		
2	Filière	Technique		
3	Méthode	Somme des salaires	Nombre d'employés	Moyenne des salaires
4	Formule conditionnelle	1161300	26	44665,38462
5	Matrice			

Avec le calcul matriciel

Nous allons effectuer le même calcul mais avec le calcul matriciel. Ces calculs seront sur la ligne 5.

Le calcul matriciel va permettre de calculer cette somme avec les mêmes conditions mais d'une manière différente. En effet, il sera possible d'intégrer directement les conditions dans la formule SOMME.

Les conditions seront séparées avec l'opérateur *.

```
{=SOMME(plage_sommée*(plage_condition et  
condition)*(plage_condition2 et condition 2)}
```

Il suffit donc de reprendre les mêmes conditions que celles précédemment décrites pour remplir la cellule B5 :

```
=SOMME ((Employé!F2:F3001=B1) *(Employé!G2:G3001=B2) *Employé!H2:H3001)
```

☛ Pour valider cette formule, appuyez simultanément sur les touches **Ctrl** **⇧** **↵**.

Une fois la formule, deux constats sont possibles :

La formule est encadrée d'accolades :

```
{=SOMME((Employe!F2:F3001=B1)*(Employe!G2:G3001=B2)*Employe!H2:H3001)}
```

Le résultat en B5 est le même qu'en B4.

Concernant la cellule C5, le principe est d'utiliser la formule **SOMME** également mais en ne précisant pas la matrice à sommer. Dans ce cas-là, la formule comptera le nombre d'éléments respectant les critères.

Les conditions seront séparées avec l'opérateur * :

```
{=SOMME(plage_condition et condition)*(plage_condition2 et condition 2)}
```

Par conséquent, la formule à appliquer est la suivante :

☞ Saisissez dans la plage la formule suivante :

```
=SOMME((Employé!F2:F3001=B1)*(Employé!G2:G3001=B2))
```

☞ Validez avec en appuyant simultanément sur les touches **Ctrl** **⇧** **↵**.

À l'affichage, la formule prend la valeur suivante :

```
{=SOMME((Employé!F2:F3001=B1)*(Employé!G2:G3001=B2))}
```

☞ Terminez en saisissant la formule **=B5/C5** dans la cellule D5 afin d'afficher la moyenne des salaires calculés avec le calcul matriciel.

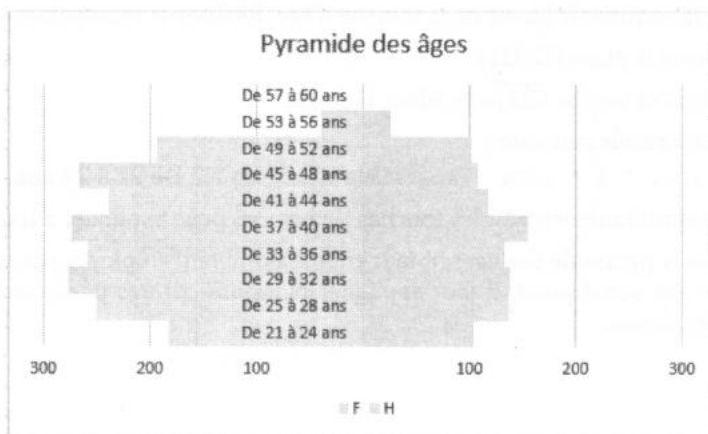
f. Création de la pyramide des âges

Définition et mise en place des données

Qu'est-ce qu'une pyramide des âges ?

Il s'agit d'un graphique représentant la proportion de personnes par âge distinguées par sexe. Il y a donc deux abscisses différentes pour les hommes et les femmes. L'ordonnée contenant les âges est triée de l'âge minimum vers l'âge maximum.

Le graphique doit son nom à l'aspect pyramidal de cette représentation :



Affichage de la source de données de la pyramide des âges

La première étape consiste à préparer le tableau source de données. Le tableau de source des données permet de centraliser les données de l'âge des employés regroupé par tranche.

Ce tableau se présentera de la manière suivante sur la feuille Calcul :

	F	G	H	I	J
1	Age Min	Age Max	Libellé	H	F
2	21	24			
3	25	28			
4	29	32			
5	33	36			
6	37	40			
7	41	44			
8	45	48			
9	49	52			
10	53	56			
11	57	60			

☞ Saisissez les données des colonnes F et G.

La colonne H contient le libellé de la tranche d'âge. Réalisez la manipulation suivante :

☞ Sélectionnez la plage H2:H11.

☞ Appuyez sur la touche **F2** pour éditer la plage.

☞ Écrivez la formule suivante :

= "De " &F2& " à " &G2& " ans". Cela affiche en H2 De 21 à 24 ans.

☞ Appuyez simultanément sur les touches **Ctrl** et **↵** pour appliquer à toute la plage.

Pour remplir la pyramide des âges, il faut trier le nombre d'employés par tranche d'âge et par sexe. Par conséquent, il faut appliquer plusieurs critères pour comptabiliser le nombre d'employés.

Il convient d'utiliser une fonction de type NB.SI.ENS pour compter le nombre d'employés répondant aux différents critères. Dans le cas présent, il y a trois critères actifs à traduire en formule Excel (vu de la cellule I2) :

Critères	Plage d'application	Critères au format Excel
Sexe correspond à l'en-tête de colonne	Employé!D2:D3001	Calcul!I1
Age supérieur ou égal à la colonne F	Employé!M2:M3001	« >= » & Calcul!F2
Age inférieur ou égal à la colonne G	Employé!M2:M3001	« <= » & Calcul!G2

Il sera important de « figer » les colonnes/lignes de certaines valeurs dans la formule à venir :

- ▶ Employé!\$D\$2:\$D\$3001 → figer la plage car elle n'évolue jamais.
- ▶ Employé!\$M\$2:\$M\$3001 → figer la plage car elle n'évolue jamais.
- ▶ Calcul!I\$1 → figer la ligne car on pointe toujours sur l'en-tête du tableau où figure le « H » ou le « F » correspondant au sexe.
- ▶ « >= » & Calcul!F2 et « <= » & Calcul!G2 → figer la colonne car c'est la colonne F qui donne l'âge minimum et la colonne G qui donne l'âge maximum pour valider le critère.

Faites la manipulation suivante :

- ☞ Sélectionnez la plage I2:J11 puis appuyez sur la touche **F2** pour éditer la formule.
- ☞ Saisissez la formule suivante :

```
=NB.SI.ENS (Employé!$D$2:$D$3001;Calcul!I$1;Employé!$M$2:$M$3001;
" >=" & Calcul!$F2;Employé!$M$2:$M$3001; " <=" & Calcul!$G2)
```

- ☞ Terminez en appuyant sur les touches **Ctrl** et **↵**. Il est possible de l'appliquer à l'ensemble de la plage I2:J11 puisque nous avons figé les cellules avec le caractère \$.

Le résultat peut ressembler à la plage suivante :

Age Min	Age Max	Libellé	H	F
21	24	De 21 à 24 ans	182	103
25	28	De 25 à 28 ans	250	137
29	32	De 29 à 32 ans	277	138
33	36	De 33 à 36 ans	259	132
37	40	De 37 à 40 ans	274	155
41	44	De 41 à 44 ans	240	117
45	48	De 45 à 48 ans	267	108
49	52	De 49 à 52 ans	194	102
53	56	De 53 à 56 ans	39	25
57	60	De 57 à 60 ans	0	1



L'affichage de ce tableau est propre au jour où il a été conçu. En effet, le tableau est basé sur l'âge de l'employé qui est lui-même calculé sur la date du jour. Le jour où ce tableau a été réalisé ne correspond pas au jour où vous le réaliserez.

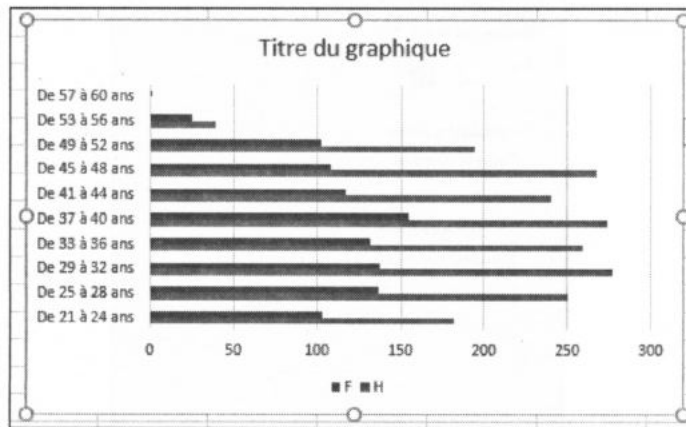
Création du graphique



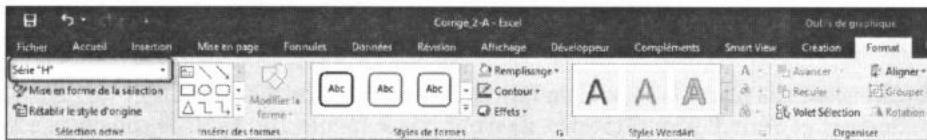
Microsoft Excel 2016 propose une présentation très différente des précédentes versions de Microsoft Excel. Toutefois, les options disponibles sont sensiblement les mêmes.

- ❖ À partir de la source de données constituée, sélectionnez la plage H1:J11.
- ❖ Dans l'onglet de menu **Insérer** - groupe **Graphiques**, sélectionnez **Insérer un histogramme ou un graphique à barres**, puis dans les **Barres 2D**, choisissez le graphique **Barres groupées**. Deux nouveaux onglets apparaissent : **Outils graphique** : **Création** et **Format**.

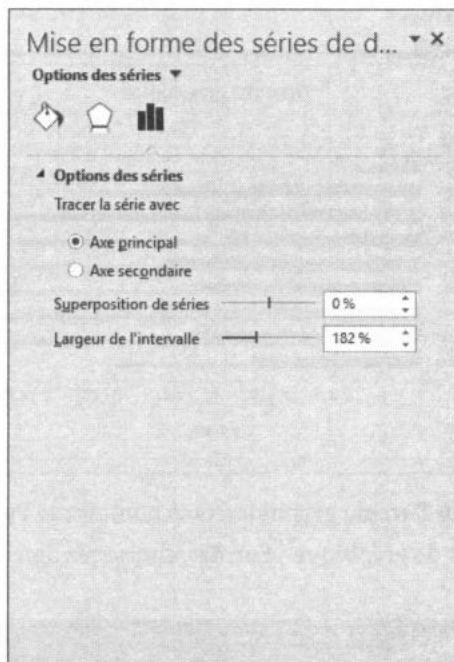
Le graphique s'affiche ainsi :



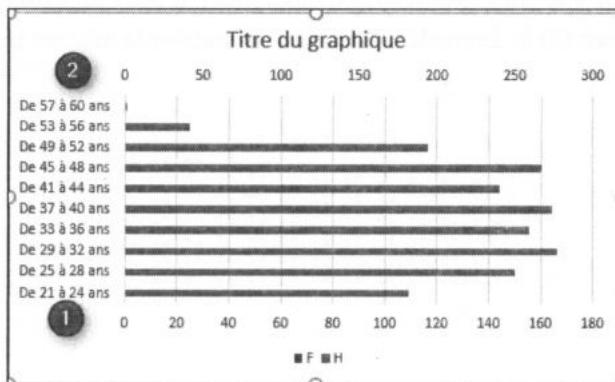
- ☞ Double cliquez sur le **Titre du graphique** puis nommez-le **Pyramide des âges**.
- ☞ Dans l'onglet **Outils de graphique - Format**, choisissez dans la zone **Sélection active** la série **Série "H"**.



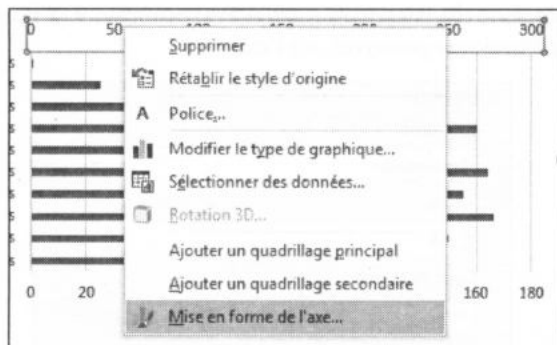
- ☞ Toujours dans le groupe **Sélection active**, cliquez sur **Mise en forme de la sélection**. Une zone apparaît sur la droite de la feuille pour modifier la mise en forme.



- ☞ Choisissez de placer la série H en tant qu'axe secondaire. À ce niveau-là, les deux axes Série H et Série F sont dissociés, ils n'ont pas le même axe. Nous le constatons avec les deux échelles (ici 1 et 2) :



- ☞ Cliquez sur l'axe 2 (en haut) pour le sélectionner puis faites un clic droit pour afficher le menu contextuel et sélectionner **Mise en forme de l'axe**.



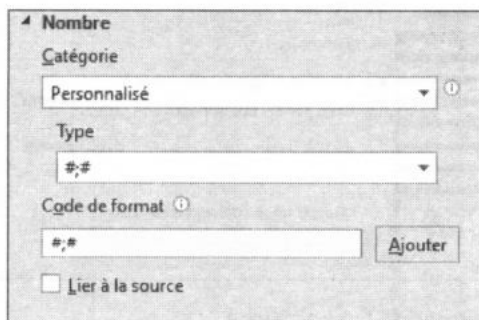
Sur le côté, le volet **Format de l'axe** s'affiche.

- ☞ Dans la zone **Option d'axe**, mettez les limites à minimum -300 et le maximum à 300.
- ☞ Enfin, cochez l'option **Valeurs en ordre inverse** pour que les données soient affichées dans l'autre sens.



- ☞ Refaites la même opération pour l'autre axe en positionnant les limites au minimum à -300 et le maximum à 300. Par contre, les valeurs doivent rester en ordre normal.

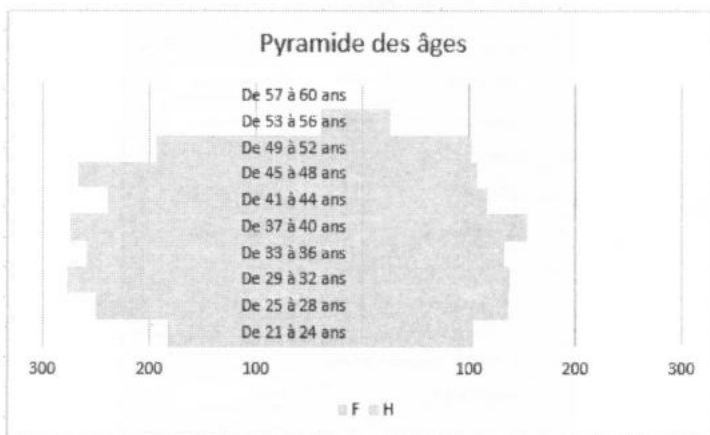
Dans la zone **Nombre**, choisissez un format **Personnalisé # ;#** pour faire en sorte que les valeurs soient toujours positives sur l'axe.



Les valeurs sont toujours positives sur l'axe :



- ❖ Sélectionnez l'axe supérieur (au-dessus du graphique) puis effacez-le en appuyant sur la touche **Supprimer**.
- ❖ Vous pouvez ajouter des touches de personnalisation pour le résultat suivant :



B. Indicateurs clés et partages

1. Description de l'exemple

a. Présentation de l'exemple

L'objectif de cet exemple est de repartir du tableau consolidé créé dans la première partie de l'exemple pour en faire ressortir les indicateurs clés. Il est donc intéressant d'aborder les différentes fonctionnalités permettant de mettre en valeur les chiffres clés du tableau contenu sur la feuille **Employé**.

b. Présentation du classeur

Le fichier **Enoncé_2-B.xlsx** est équivalent au fichier **Corrigé_2-A.xlsx**. Cet exemple intervient dans la continuité de la première partie. Il n'y a pas de modifications.

c. Fonctionnalités

Dans un premier temps, nous allons calculer les statistiques des salaires :

- ▶ Calculer le rang d'un salaire par rapport à l'ensemble des salaires ;
- ▶ Calculer le premier et le neuvième déciles des salaires pour identifier les 10 % de salaires les plus/moins élevés ;
- ▶ Mettre en valeur les salaires les plus élevés grâce à une mise en forme conditionnelle ;
- ▶ Afficher dans un graphique synthétique l'ensemble des salaires.

2. Notions de cours

a. Formules Excel

Par défaut, Excel propose de nombreuses statistiques. L'objectif n'est pas de détailler toutes les formules statistiques mais d'apprendre à les utiliser.

Où trouver les formules ?

Les formules se situent dans l'onglet **Formules** et sont triées par catégories : **Financier**, **Logique**, **Texte**, **Date/Heure**...



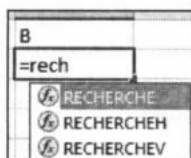
Il est possible de rechercher une fonction via le bouton **Insérer une fonction**, toutefois la recherche de fonction n'aboutit que trop rarement au résultat souhaité. Par conséquent pour insérer une fonction, il est recommandé d'effectuer une recherche par thème.

Comment les comprendre ?

Prenons l'exemple d'une recherche verticale :

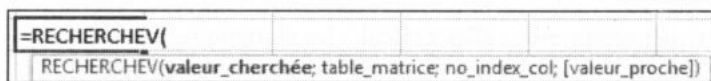
☞ Tapez =RECH.

La liste des formules apparaît :



☞ Avec les touches du clavier **[↑]** et **[↓]**, positionnez-vous sur la formule souhaitée, puis avec la touche **[↵]**, validez votre sélection.

La formule est affichée avec ses arguments.

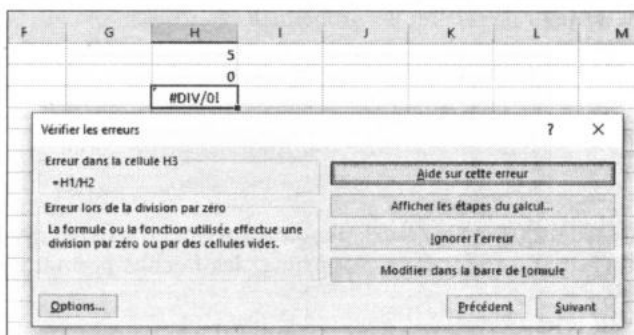


☞ Dans l'infobulle, cliquez sur le nom de la formule pour afficher l'aide en ligne et notamment les arguments associés.



Les arguments sont obligatoires, sauf s'ils sont entre crochets, comme ici [valeur_proche].

Le bouton **Vérification des erreurs** permet d'afficher une explication approfondie de l'erreur :



En cliquant sur **Afficher les étapes du calcul**, la fenêtre d'évaluation de la formule est affichée pour vérifier pas à pas les calculs.

b. Création d'un graphique Sparkline

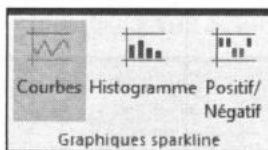
Un graphique de type Sparkline est une représentation graphique réalisée sur une seule cellule. Cette option est intéressante car elle permet de faire apparaître de nombreuses données dans un espace très restreint à savoir une cellule. Il y a trois types de graphiques possibles : **Positif/Négatif**, **Histogramme** et **Courbe**.

Pour réaliser un graphique de type Sparkline :

- ☞ Saisissez le tableau suivant dans la plage A1:A11 puis sélectionnez la plage de cellules contenant les données.

	A
1	5
2	3
3	4
4	10
5	12
6	10
7	18
8	21
9	21
10	19
11	25

- ☞ Dans l'onglet **Insérer**, choisissez le groupe **Graphiques sparkline** puis cliquez sur **Courbes**.



La fenêtre **Créer des graphiques sparkline** s'affiche comme ci-dessous.

- ☞ Sélectionnez la **Plage d'emplacements** (la **Plage de données** est déjà renseignée car elle a été sélectionnée à l'étape 1)



Si vous vous positionnez sur une cellule vide lors de la création du graphique, le système considère qu'il s'agit d'une plage d'emplacement et il vous faudrait renseigner la plage de données.

Créer des graphiques sparkline

Sélectionnez les données de votre choix

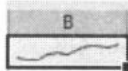
Plage de données : A1:A11

Sélectionnez l'emplacement des graphiques sparkline

Plage d'emplacements : SBS1

OK Annuler

Le graphique est inséré dans la cellule B1 :



c. Mise en forme conditionnelle simple

La mise en forme conditionnelle simple consiste à afficher une mise en forme spécifique prédéfinie en fonction du contenu des cellules d'une plage.

Après avoir sélectionné la plage à mettre en forme, cette fonctionnalité se trouve dans l'onglet **Accueil** puis le groupe **Styles**. En cliquant sur **Mise en forme conditionnelle**, vous pouvez choisir entre cinq types de règles prédéfinies :

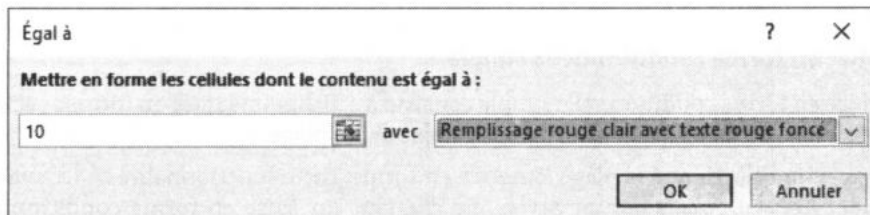
Règle de mise en surbrillance des cellules

Un format spécifique est appliqué lorsque dans une plage de cellule, une cellule remplit une condition donnée : par exemple, être supérieure à une valeur.



Pour mettre en valeur les cellules égales à 10, cliquez sur Règles de mise en surbrillance des cellules et choisissez Égal à....

- ❖ Saisissez la valeur du critère à appliquer, 10 ici, dans le champ Mettre en forme les cellules dont le contenu est égal à.
- ❖ Choisissez Remplissage rouge clair avec texte rouge foncé comme format de mise en forme conditionnelle puis cliquez sur OK.

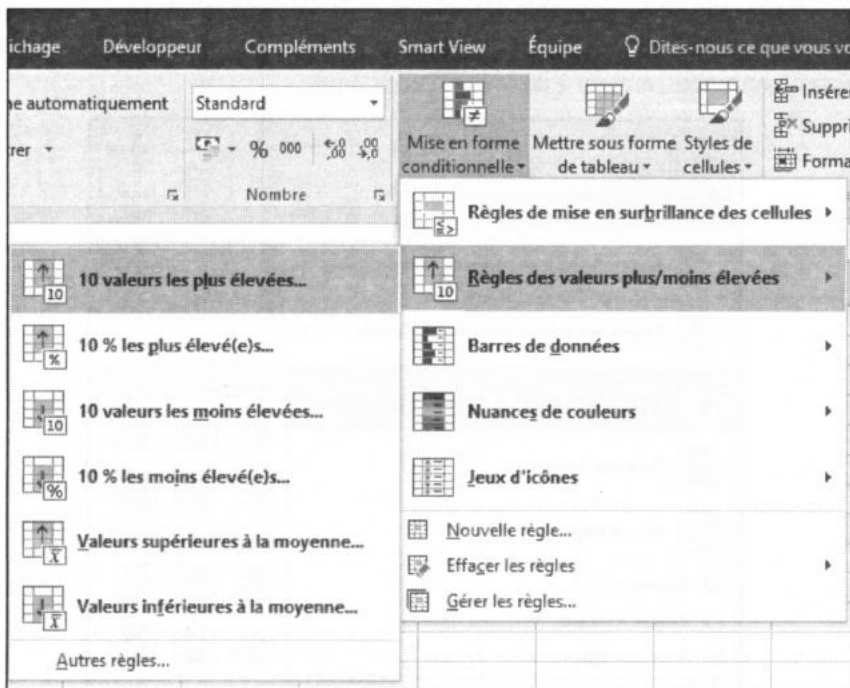


Vous obtenez le résultat suivant :

	A
1	5
2	3
3	4
4	10
5	12
6	10
7	18
8	21
9	21
10	19
11	25

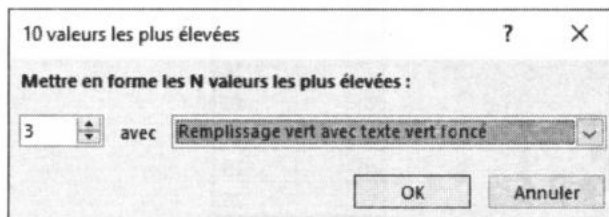
Règle de valeurs plus/moins élevées

Cela consiste à appliquer un format spécifique aux cellules faisant partie d'un échantillon déterminé. Par exemple, les valeurs faisant partie des 10 % les plus élevées.



Pour mettre en valeur les trois valeurs les plus élevées :

- ❏ Choisissez l'option Règles des valeurs plus/moins élevées, puis 10 valeurs les plus élevées.
- ❏ Dans la zone Mettre en forme les N valeurs plus élevées, saisissez 3 pour mettre en avant le Top 3.
- ❏ Choisissez le format Remplissage vert avec texte vert foncé puis cliquez sur OK.



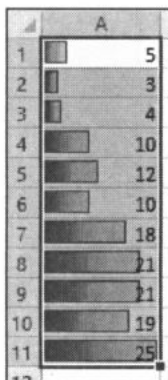
Barre de données

Les barres de données sont des effets visuels apparaissant directement dans les cellules de la plage sélectionnée. La longueur de la barre au sein d'une cellule dépend de la valeur de celle-ci par rapport à l'ensemble des valeurs de la plage de cellules.

Pour les barres de données, seule la couleur doit être choisie, la longueur des barres s'adaptent automatiquement à la valeur.



Voici une représentation d'une série de barres de données :



Nuance de couleurs

À l'instar des barres de données, les nuances de couleurs sont des effets visuels apparaissant directement dans les cellules de la page sélectionnée. La couleur varie d'un niveau moins intense à un niveau plus intense en fonction de la valeur de la cellule par rapport à l'ensemble des valeurs contenues dans la plage de cellules.

Seul le dégradé doit être choisi, les nuances s'adaptent automatiquement à la valeur.



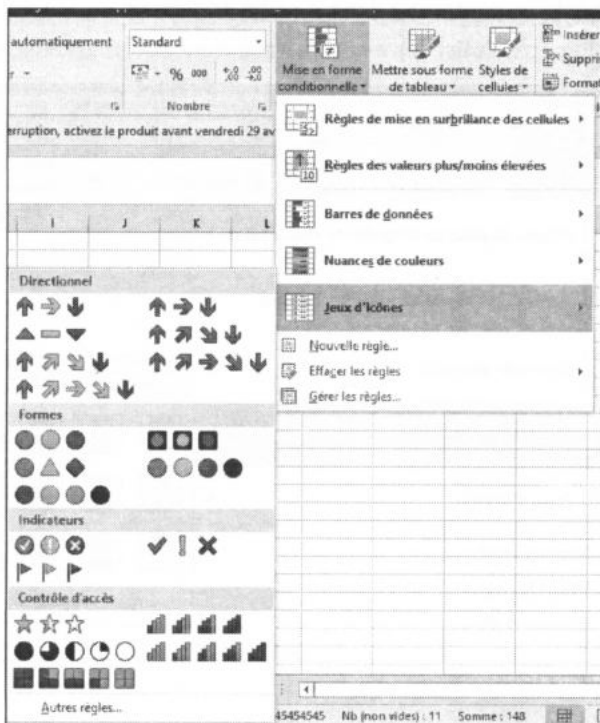
Voici une représentation d'une mise en forme conditionnelle avec nuance de couleurs.

	A
1	5
2	8
3	4
4	10
5	12
6	10
7	18
8	21
9	21
10	19
11	25

Jeux d'icônes

Les jeux d'icônes sont des effets visuels représentant un ensemble d'icônes appliquées aux valeurs des cellules contenues dans une plage sélectionnée. Les icônes de type « positive » seront appliquées aux valeurs les plus hautes, les icônes de type « négative » seront appliquées aux valeurs les plus basses.

Voici les icônes disponibles :



Avec l'application sur l'exemple suivant :

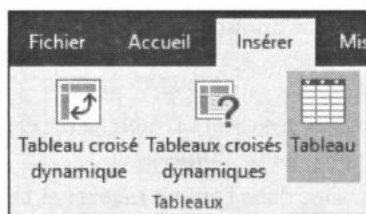
	A
1	5
2	3
3	4
4	10
5	12
6	10
7	18
8	21
9	21
10	19
11	25
12	

d. Introduction au tableau

Depuis Excel 2007, il est possible de créer un tableau à partir d'une plage de données. Le tableau (appelé aussi tableau de données) permet de manipuler plus facilement les données avec des filtres, des tris, et surtout l'application de formules à l'ensemble d'une colonne.

Où trouver cette fonctionnalité ?

La fonctionnalité est disponible dans le menu **Insérer** puis **Tableau**.



La fenêtre **Créer un tableau** s'affiche et permet de déterminer la plage qui deviendra un tableau. Il est également possible de choisir si le tableau comporte des en-têtes.

Créer un tableau ? X

Où se trouvent les données de votre tableau ?

=A1:H10

Mon tableau comporte des en-têtes

À la création d'un tableau, un nouvel onglet **Outils de tableau - Création** associé au tableau est disponible.

Formules

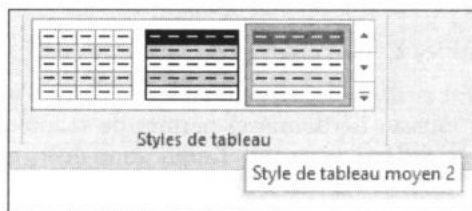
Lorsqu'une formule est appliquée au début d'une colonne, elle s'applique sur l'ensemble de la colonne.

Colonne

La colonne est nommée par son nom d'en-tête : [Nom En-tête]. Il est possible de faire référence à des cellules comme ici la cellule en cours de la colonne 1 : =[@[Colonne 1]

Mise en forme

Il est plus facile d'appliquer une mise en forme à un tableau puisque dans l'onglet **Outils de tableau - Création**, certains styles sont disponibles et applicables facilement :



3. Réalisation de l'exemple

❖ Tout d'abord, ouvrez le fichier **Enoncé_2-B.xlsx**.

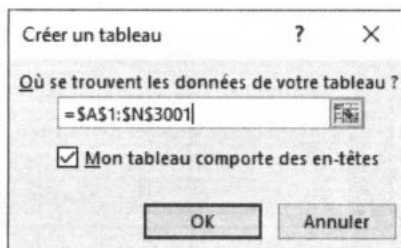
a. Mise en place du tableau

Dans un premier temps, nous allons créer un tableau Excel pour faciliter la manipulation des données contenues dans la feuille **Employé**.

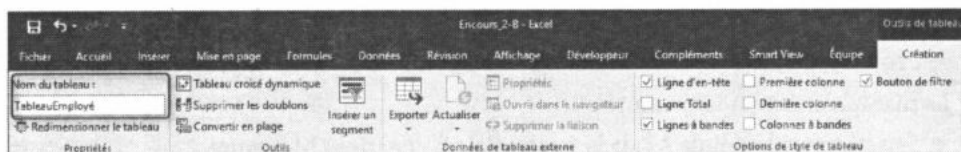
❖ Dans la feuille **Employé**, allez dans l'onglet **Insérer** et cliquez sur le bouton **Tableau**.

La fenêtre **Créer un tableau** apparaît.

❖ Sélectionnez la plage correspondant à l'ensemble du tableau et cochez la case que **Mon tableau comporte des en-têtes**.



- ❖ L'onglet Outils de tableau - Création apparaît.
- ❖ Dans le groupe Propriétés, modifiez le nom du tableau pour lui donner celui de TableauEmployé.



La plage de données \$A\$1:\$N\$3001 est désormais un tableau Excel nommé TableauEmployé.

b. Formules statistiques

À partir des formules statistiques, nous allons calculer trois indicateurs portant sur les salaires en cours.

Connaître le rang du salaire

Nous allons afficher en colonne O l'information du rang du salaire. Cela correspond à la position du montant du salaire par rapport à l'ensemble des salaires. Le plus haut salaire aura le rang 1, le plus bas aura le rang 3000.

Pour déterminer le rang du salaire, la fonction RANG donne la possibilité de connaître la position d'une valeur au sein d'une plage. Son utilisation est relativement intuitive :

= RANG(valeur ; plage)

Cette formule donnera la position du salaire parmi l'ensemble des 3000 salaires des collaborateurs.

Pour appliquer cette formule :

- ❖ Dans la cellule O1, tapez Rang pour initier la colonne.
En faisant cette manipulation, la colonne sera immédiatement ajoutée au tableau TableauEmployé.
- ❖ Dans la cellule O2, saisissez =RANG puis choisissez la cellule du salaire H2. Elle sera affichée ainsi : [=RANG([@[Salaire Brut annuel]])]
- ❖ Saisissez ;
- ❖ Toujours dans la formule RANG, choisissez la plage sur laquelle le rang est calculé en sélectionnant l'ensemble de la colonne Salaire Brut annuel.
La formule s'affiche ainsi : =RANG([@[Salaire Brut annuel]];TableauEmployé[[#Tout];[Salaire Brut annuel]])
- ❖ En validant la formule dans ce tableau, elle s'applique à l'ensemble de la colonne.

Calcul de la moyenne des salaires à partir d'un tableau

Dans la feuille **Calcul**, nous allons récupérer la moyenne globale des salaires. Nous allons pour cela nous appuyer sur les données du tableau.

Pour calculer la moyenne, la formule éponyme reste la plus appropriée :

=MOYENNE(plage)

La manipulation est la suivante :

- ☞ Sélectionnez la cellule **A7** de la feuille **Calcul** et écrivez **Moyenne**.
- ☞ Sélectionnez la cellule **A8** de la feuille **Calcul** et saisissez la formule suivante :
=MOYENNE(TableauEmployé[Salaire Brut annuel])

Calcul du premier et du dernier décile des salaires

Les déciles sont les valeurs qui permettent de répartir une liste de valeurs ordonnées en 10 parties égales. Par conséquent les salaires inférieurs au premier décile seront dans les 10 % des salaires les plus bas, les salaires supérieurs au neuvième décile seront dans les 10 % des salaires les plus élevés.

Nous allons faire figurer les déciles dans la plage **A9:A12** de la feuille **Calcul**.

La formule **DECILE** n'existe pas, toutefois, la formule **CENTILE** existe et avec une légère adaptation, nous allons pouvoir calculer le décile.

Le centile est un décile divisé par 10. Nous allons donc adapter la formule : 10 % pour le premier décile, 90 % pour le dernier.

La formule **CENTILE** a la structure suivante :

=CENTILE(plage ;centile)

Pour avoir les premier et neuvième déciles :

- ☞ Sur la feuille **Calcul**, dans la cellule **A9**, écrivez **1^{er} décile**.
- ☞ Sélectionnez la cellule **A10** et saisissez la formule suivante : =CENTILE(TableauEmployé[Salaire Brut annuel];0,1)
- ☞ Sur la feuille **Calcul**, dans la cellule **A11** écrivez **Dernier décile**.
- ☞ Sélectionnez la cellule **A12** et saisissez la formule suivante : =CENTILE(TableauEmployé[Salaire Brut annuel];0,9)

Vous obtiendrez les valeurs suivantes :

	A
1	Grade
2	Filière
3	Méthode
4	Formule conditionnelle
5	Matrice
6	
7	Moyenne
8	62 823,03 €
9	1er décile
10	36800
11	Dernier décile
12	94210

c. Mise en valeur des données

La mise en valeur des données va consister à appliquer un format spécifique à certaines données du tableau **TableauEmployé**. Nous allons appliquer des mises en forme conditionnelles pour mettre en valeur les données.

Top 20 des salaires

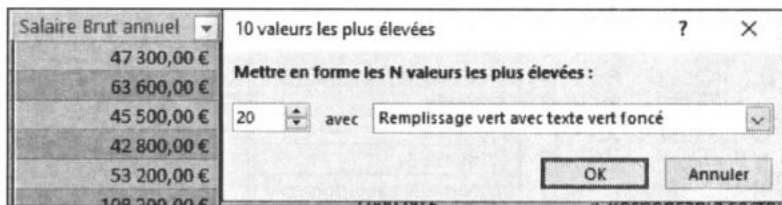
Pour afficher les 20 plus gros salaires de l'année N, il faut réaliser une mise en forme conditionnelle et utiliser l'option **Règles des valeurs plus/moins élevées**. La mise en forme sera à appliquer sur la colonne H de la feuille **Employé**.

Pour réussir cette mise en valeur :

- ☞ Sélectionnez la plage des salaires H2:H3001.
- ☞ Onglet **Accueil**, cliquez sur **Mise en forme conditionnelle**.
- ☞ Choisissez le sous-menu **Règles des valeurs plus/moins élevées** puis cliquez sur **10 valeurs les plus élevées**.

La fenêtre de configuration du paramétrage de la mise en forme conditionnelle s'affiche.

- ☞ Dans la partie **Mettre en forme les N valeurs les plus élevées** sélectionnez 20, puis pour le format, choisissez **Remplissage vert avec texte vert foncé**.



Constatez le résultat sur la feuille.

	F	G	H
1	Grade	Filière	Salaires Brut annuel
2	G12	Fonctions transv	199 500,00 €
3	G12	Fonctions transv	190 200,00 €
4	G12	Fonctions transv	170 600,00 €
5	G11	Fonctions transv	159 900,00 €
6	G11	Fonctions transv	155 900,00 €
7	G11	Fonctions transv	155 600,00 €
8	G11	Fonctions transv	155 500,00 €
9	G11	Fonctions transv	154 400,00 €
10	G12	Business	153 400,00 €
11	G11	Fonctions transv	151 900,00 €
12	G11	Fonctions transv	150 900,00 €
13	G11	Fonctions transv	150 800,00 €
14	G12	Business	150 300,00 €
15	G11	Business	148 700,00 €
16	G11	Business	148 300,00 €
17	G11	Business	148 200,00 €
18	G11	Fonctions transv	147 900,00 €
19	G11	Business	147 000,00 €
20	G11	Business	146 800,00 €
21	G11	Business	146 600,00 €
22	G11	Business	146 500,00 €
23	G11	Business	145 000,00 €

Mise en avant des employés non augmentés

Pour mettre en avant les employés qui n'ont pas été augmentés, il faut réaliser une mise en forme conditionnelle et utiliser l'option Règles de mise en surbrillance des cellules. En effet, il est possible de mettre en surbrillance les cellules qui contiennent une valeur nulle pour l'augmentation. La mise en forme sera à appliquer sur la colonne N de la feuille Employé.

Pour effectuer cette mise en valeur :

- ☛ Sélectionnez la plage des augmentations N2:N3001.

- ❖ Onglet **Accueil**, cliquez sur **Mise en Forme conditionnelle**.
- ❖ Choisissez le sous-menu **Règles de mise en surbrillance des cellules** puis cliquez sur **égal à**.
La fenêtre de configuration du paramétrage de la mise en forme conditionnelle s'affiche.
- ❖ Dans la partie **Mettre en forme les cellules dont le contenu est égal à** choisissez **0** puis choisissez **Remplissage rouge clair avec texte rouge foncé**.

Constatez le résultat sur la feuille.

M	N
Age	Pourcentage d'augmentation
40	3%
49	9%
40	4%
36	14%
33	8%
48	5%
25	0%
27	5%
23	5%



Les nouveaux arrivants sont considérés comme n'ayant pas eu d'augmentation.

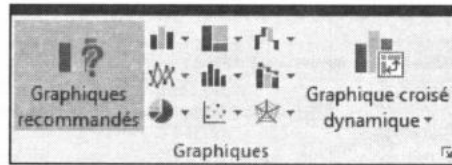
d. Sparkline contre graphique classique

L'objectif est d'afficher les données à travers un graphique de manière claire et visible pour les utilisateurs. Le graphique en courbe des salaires de l'entreprise permet de mettre en évidence la répartition des salaires. Les graphiques se baseront sur la colonne H de la feuille **Employé**.

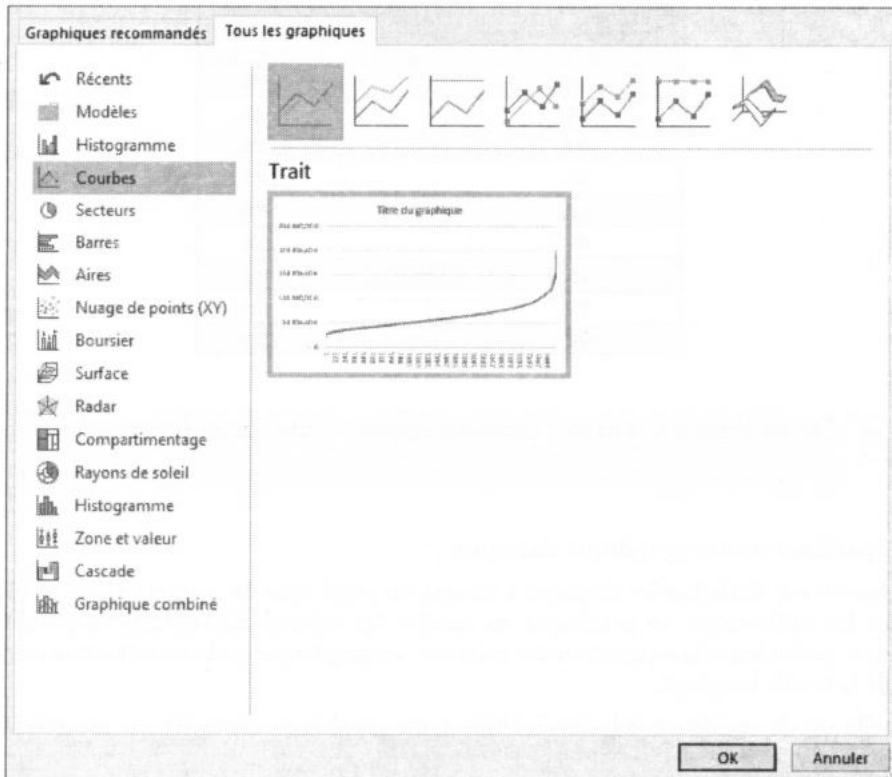
Quelle est la meilleure solution ? Utiliser un graphique classique ou un graphique Sparkline ?

Affichage d'un graphique normal

- ❖ Dans la feuille **Employé**, tout d'abord, triez la colonne H à savoir les salaires bruts annuels du plus petit au plus grand.
- ❖ Comme il s'agit d'afficher une courbe des salaires, le seul axe du graphique sera le salaire. Par conséquent, sélectionnez la plage H1:H3001.
- ❖ Cliquez sur l'onglet **Inserer** puis dans le groupe **Graphiques**, choisissez **Graphiques recommandés**.

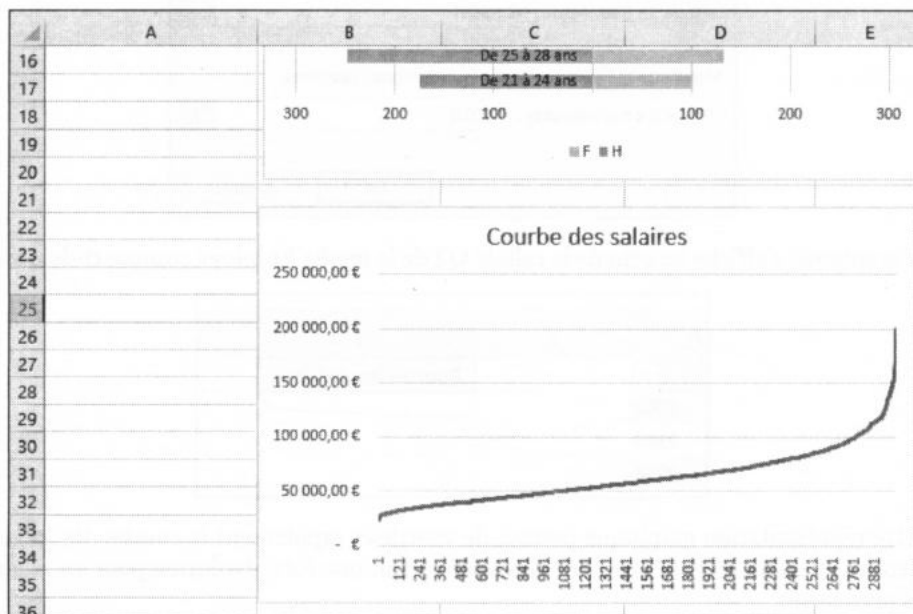



- ❖ Dans la fenêtre, choisissez **Tous les graphiques** puis choisissez **Courbes**. Le graphique en courbe s'affiche comme ci-dessous.



- ☞ Copiez le graphique sur la cellule B22 de la feuille Calcul et modifiez le titre en **Courbe des salaires**.

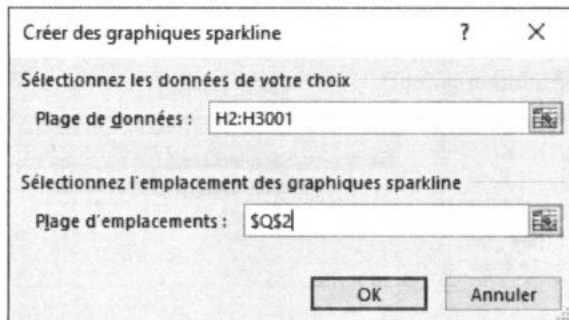
Vous obtenez le résultat suivant :



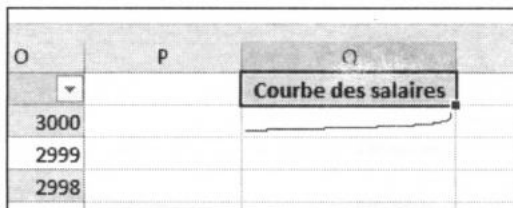
 *L'interface de création d'un graphique est différente dans les versions antérieures de Microsoft Office Excel mais il est possible de choisir directement le graphique en courbe.*

Création d'un graphique Sparkline

- ☞ Sélectionnez les valeurs de la plage des salaires bruts annuels H2:H3001 sans prendre l'en-tête.
- ☞ Dans l'onglet **Insérer** - groupe **Graphiques sparkline**, choisissez **Courbes**.
- ☞ La fenêtre **Créer des graphiques sparkline** apparaît. Dans la zone **Plage de données**, la plage H2:H3001 est sélectionnée. Dans la zone **Plage d'emplacements**, saisissez Q2. Il n'est pas possible d'afficher le graphique sur une autre feuille que la feuille **Employé**.



Le graphique s'affiche au sein de la cellule Q2 de la feuille **Employé** comme ci-dessous :



Cette représentation graphique permet de visualiser rapidement la courbe des salaires mettant en avant la lente évolution des salaires puis une forte évolution pour les salaires les plus élevés.

Chapitre 3

Gestion des ventes et formulaires VBA

A. Formulaire de gestion des ventes	79
B. Protéger le classeur	132

A. Formulaire de gestion des ventes

1. Description de l'exemple

a. Présentation de l'exemple

Nous sommes l'entreprise **SacEni**, un distributeur qui commercialise des sacs de sports, un sac en taille L, un autre en taille XL. L'activité débute et l'entreprise souhaite se doter d'un simple fichier pour suivre ses ventes et son stock.

L'outil que vous allez mettre en place va permettre au vendeur de cette petite entreprise de créer, stocker et télécharger des factures.

L'outil va se présenter comme un formulaire à remplir par le vendeur. Il sera accessible à partir d'un fichier Excel et du bouton **Accéder à l'outil de gestion des ventes** qui sera positionné sur la feuille d'ouverture du classeur.

Voici l'outil une fois qu'il sera finalisé :

Logiciel de gestion des ventes

Quantité	Produit
5	Sac taille XL
	Sac taille L
	Sac taille xl

Ajouter à la ligne d

4*Sac taille L=80€
5*Sac taille XL=150€

Supprimer la ou les lignes sélectionnées

Total HT 230 € Remise 10 % pro

Total après remise 230 € TVA 46 €

Total TTC 276 €

Sauvegarder et imprimer la commande

b. Présentation du fichier

Pour réaliser cet exemple, vous allez utiliser le fichier Excel **Enoncé_3-A.xlsm**. Le format **XLSM** signifie que c'est un fichier Excel qui prend en charge les macros (permettant l'utilisation de VBA, souvent désactivée par défaut).

Ce fichier contient trois feuilles Excel (appelées *sheets* en Visual Basic) :

Feuille Accueil

Cette feuille est destinée à contenir uniquement le bouton **Accéder à l'outil de gestion des ventes**. Il permettra l'accès à l'outil de gestion des ventes.

Initialement cette feuille est vide.

Feuille Produits

La feuille **Produits** recense les produits vendus par la société. Elle contient trois colonnes :

	Colonne A	Colonne B	Colonne C
Ligne 1	Nom du produit	Prix hors taxe	Quantité disponible
Ligne 2	Sac taille L	20 €	90
Ligne 3	Sac taille XL	30 €	50

Feuille Factures

Cette feuille contiendra les factures créées avec l'outil de gestion des ventes. Elles seront référencées par numéro et vous trouverez également la date/heure de l'édition, ainsi que le montant.

	Colonne A	Colonne B	Colonne C
Ligne 1	Numéro de facture	Date et heure	Montant de la facture

c. Fonctionnalités

L'objectif est donc d'avoir un outil qui permet de gérer les ventes des sacs de l'entreprise. Voici la retranscription des besoins sous forme d'exigences.

Exigences métiers

Les exigences métiers correspondent à la description des fonctionnalités globales de l'application, c'est le niveau de détail le plus faible :

- ▶ Créer une facture ;
- ▶ Tracer la facture ;
- ▶ Mettre à jour les stocks de produits.

Retranscription de ces exigences métiers en fonctionnalités

Les fonctionnalités permettent de détailler les exigences métiers en fonctionnalités qui correspondent aux actions utilisateurs et aux traitements du système. Cette liste doit être exhaustive afin de permettre de réaliser ces fonctionnalités sous forme d'application.

- ▶ Ajouter une ligne de commande
 - ▶ Choisir la quantité ;
 - ▶ Choisir le produit ;
 - ▶ Valider la ligne de commande ;
- ▶ Afficher la commande.
- ▶ Supprimer une ligne de commande.

- ▶ Faire le total et permettre l'application d'une remise de 10 %.
- ▶ Mettre à jour les stocks.

2. Notions de cours

Cet exemple contient de nombreux nouveaux concepts liés à la programmation avec le langage Visual Basic... Quelques repères sont donc utiles pour pouvoir démarrer sereinement.

a. Concept de programmation

Voici une description simplifiée de quelques notions de base de programmation permettant une meilleure approche des exemples proposés.

Objet et classe

Un **objet** est une entité informatique, il peut être de toute forme et chaque objet est unique. Il est caractérisé selon son type.

La **classe** correspond à la définition de l'objet, elle servira de canevas pour la création de nouveaux objets. Par conséquent, tous les objets d'une même classe auront les mêmes propriétés, ils se différencieront par les valeurs de leurs propriétés.

Exemple :

Classe	Objets
Cell (cellule d'une feuille Excel)	<ul style="list-style-type: none"> ▶ Cells("A1") : Cellule A1 de la feuille en cours ; ▶ Cells("C4") : Cellule C4 de la feuille en cours.
Sheet (Feuille de classeur)	<ul style="list-style-type: none"> ▶ Sheets(0) : première feuille du classeur en cours ; ▶ Sheets(1) : deuxième feuille du classeur en cours.
Textbox (zone de texte saisissable dans un formulaire)	<ul style="list-style-type: none"> ▶ Textbox1 : zone de texte saisissable nommée Textbox1 par l'utilisateur ▶ Textbox2 : zone de texte saisissable nommée Textbox2 par l'utilisateur.

Propriétés

Une **propriété** correspond à un attribut d'une classe. Lorsqu'un objet est créé, il a donc des valeurs assignées à ses propriétés.

Exemple : La cellule d'une feuille comporte de nombreuses propriétés, comme par exemple la valeur : `Cells("A1").value`

Méthode

Une **méthode** correspond à une action qui peut être réalisée par un objet. Par exemple, l'objet feuille de calcul (Sheets) propose une méthode Add qui permet d'ajouter une feuille.

Exemple : Sheets.Add

Collections

Une **collection** est une liste d'objets d'une même classe. Par exemple, la collection sheets correspond à l'ensemble des feuilles. En Visual Basic, les collections sont des objets à part entière avec leurs propres méthodes et propriétés.

Variables

Une **variable** est une entité informatique qui permet de stocker des informations au sein de l'application, elle se déclare de la manière suivante :

- ▶ Dim : permet de définir la variable (Public pour une variable publique) ;
- ▶ Nom_variable : permet de donner un nom à la variable ;
- ▶ As TypeVariable : permet de typer la variable.

Exemple :

```
Dim MaVariable As String
```

Cela signifie que la variable MaVariable est déclarée en tant que chaîne de caractères.

Les variables sont :

- ▶ Publiques : elles sont accessibles sur l'ensemble de l'application. Elles sont déclarées en dehors de toute procédure de code ;
- ▶ Privées : elles sont accessibles uniquement dans la procédure où elles sont déclarées (sur une procédure donnée).

Les variables sont typées principalement pour les trois motifs suivants :

- ▶ Cela permet d'avoir des méthodes (voir précédemment) adaptées à la variable : une addition de chaînes de caractères correspond à la concaténation alors qu'une addition de nombres correspond à la somme des valeurs :

Opérations	Valeur de la variable chaîne de caractères	Valeur de la variable nombre entier
MaVar = "A" + "E"	"AE"	Erreur
MaVar = 1 + 2	"12"	3

- ▶ Cela facilite le développement et l'usage de variable, le contenu de la variable est attendu.
- ▶ Chaque type de variable a une quantité de mémoire allouée, par conséquent, utiliser le bon type de variable permet d'économiser de la mémoire.

Voici les types de variable et le détail de chacune :

Nom	Type	Détails
Byte	Numérique	Nombre entier de 0 à 255
Integer	Numérique	Nombre entier de -32'768 à 32'767
Long	Numérique	Nombre entier de -2'147'483'648 à 2'147'483'647
Currency	Numérique	Nombre à décimale fixe de -922'337'203'685'477.5808 à 922'337'203'685'477.5807
Single	Numérique	Nombre à virgule flottante de -3.402823E38 à 3.402823E38
Double	Numérique	Nombre à virgule flottante de -1.79769313486232D308 à 1.79769313486232D308
String	Texte	Texte
Date	Date	Date et heure
Boolean	Boolean	True (vrai) ou False (faux)
Object	Objet	Objet Microsoft (exemple cellule, plage de cellule, feuille)
Variant	Tous	Valeur par défaut si non déclarée

b. Concept de formulaire

Formulaire

Un **formulaire** (dit *form*) est une fenêtre d'interaction entre l'utilisateur et le système. Il s'agit d'une interface visuelle permettant de restituer et/ou collecter de l'information dans le but de faire des traitements.

L'objet formulaire est le contenant des autres objets visuels. Cela signifie qu'il comporte d'autres objets visuels, ceux-là même proposés dans la boîte à outils.

Une même application peut contenir plusieurs formulaires.

Les contrôles




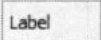

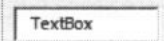

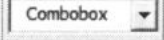



Un formulaire est un contenant de contrôles. Ces **contrôles** sont des objets visuels qui permettent l'interaction avec l'utilisateur.



Par défaut, une quinzaine de contrôles sont proposés dans la boîte à outils, et dans le cadre de cet exemple, sept d'entre eux seront détaillés. Toutefois, il est possible d'importer de nouveaux contrôles, mais il s'agit d'une utilisation plus avancée de l'application VBA.

Un contrôle est créé lorsqu'il est déposé sur un formulaire. Un formulaire peut contenir plusieurs contrôles du même type, les contrôles étant des objets, ceux-ci sont uniques. La propriété « *Name* » (nom) de chaque contrôle doit être unique au sein d'un même formulaire. Il est en revanche possible de trouver un contrôle avec la même valeur pour la propriété *Name* dans un autre formulaire :

```
\ Un objet Control1 positionné sur le Formulaire1
Formulaire1.Control1 \ Un objet Control1 positionné sur le
Formulaire2 Formulaire2.Control1
```

Voici quelques types de contrôles et leur utilisation :

Nom de l'objet	Icône dans la boîte à outil	Affichage sur un formulaire	Utilisation
CommandButton			Bouton de formulaire généralement déclencheur d'actions.
Label			Zone de texte non saisissable par l'utilisateur : elle peut être modifiée par le système.
Textbox			Zone de texte saisissable par l'utilisateur.
ComboBox			Liste déroulante avec la possibilité de choisir une seule valeur.
OptionButton			Bouton radio permettant de sélectionner une valeur dans un groupe : usuellement, il n'est pas utilisé seul (exemple : homme ou femme).
CheckBox			Case à cocher permettant de sélectionner une valeur ou plus dans un groupe (exemple : sélection de loisirs).

Nom de l'objet	Icône dans la boîte à outil	Affichage sur un formulaire	Utilisation
ListBox			Il s'agit d'une liste avec la possibilité de voir la liste complète à la différence d'un contrôle ComboBox où seule une valeur est affichée. De plus, cette liste permet de sélectionner plusieurs valeurs (à paramétrer dans les attributs de l'objet).

c. Rédaction du code

Cette sous-partie va vous apprendre les instructions de base pour coder. Où, quand et comment écrire du code ?

Module

Un **module** est une feuille dans laquelle il est possible d'écrire du code. Les modules peuvent contenir plusieurs procédures.

Procédure

Une **procédure** représente une portion de code nommée par un titre. Une procédure peut être appelée à tout moment dans l'application par l'instruction `Call`.

Elle peut avoir des paramètres en entrée appelés **arguments**. Ceux-ci peuvent être facultatifs.

Une procédure commence par l'instruction `Sub` avec le nom de la procédure puis termine par l'instruction `End Sub`. Par défaut la portée de celle-ci est publique, cela signifie qu'elle est visible (et donc qu'il est possible de l'appeler) à tout endroit dans l'application. Toutefois il est possible de spécifier la portée de la procédure en écrivant `Public` ou `Private` (privée) avant l'instruction `Sub`. L'instruction `Private Sub` limitera la visibilité de la procédure au module où elle est déclarée.

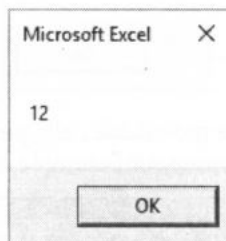
Voici un exemple de procédure affichant le produit d'un calcul au sein d'un module.

```
Public Sub MaProcedure()
'Définition de mes variables A et B en tant que nombre entier
Dim A As Integer
Dim B As Integer
'Affectation des valeurs aux variables
A = 3
B = 4
'Appel à la procédure Calculer avec les 2 arguments A et B
```

```
Call Calculer(A, B)
End Sub

Public Sub Calculer(Valeur1 As Integer, Valeur2 As Integer)
'Définition de la variable Produit qui représentera le calcul
Dim Produit As Integer
Produit = Valeur1 * Valeur2
'Affichage dans une pop-up de la valeur de la variable Produit
MsgBox (Produit)
End Sub
```

Résultat de l'exécution de la procédure MaProcédure :



Fonction

À l'instar de la procédure, la **fonction** est une portion de code qui possède sa portée et qui est positionnée dans un module. La fonction peut également avoir des arguments en entrée.

La fonction, contrairement à la procédure, dispose d'une valeur de retour. La valeur produite est stockée dans une variable portant le nom de la fonction.

Une fonction débute avec l'instruction `Function` et se termine par l'instruction `End Function`.

La fonction doit assigner une valeur en tant que résultat.

Voici le même exemple que précédemment mais avec l'utilisation d'une fonction.

```
Public Sub MaProcédureFunction()
'Définition de mes variables A et B en tant que nombre entier
Dim A As Integer
Dim B As Integer
'Affectation des valeurs aux variables
A = 3
B = 4
'Appel à la fonction FN_Calculer avec les 2 arguments A et B
MsgBox (FN_Calculer(A, B))
End Sub

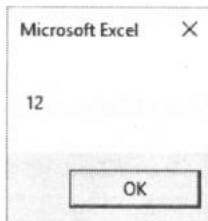
Public Function FN_Calculer(Valeur1 As Integer, Valeur2 As Integer)
```

```

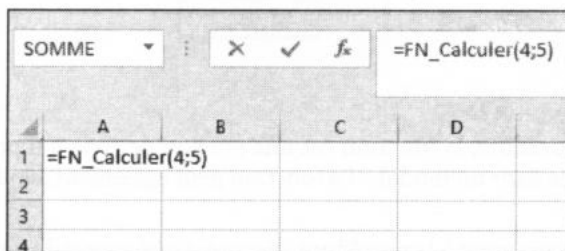
Dim Produit As Integer
Produit = Valeur1 * Valeur2
'Affectation de la valeur à la fonction
FN_Calculer = Produit
End Function

```

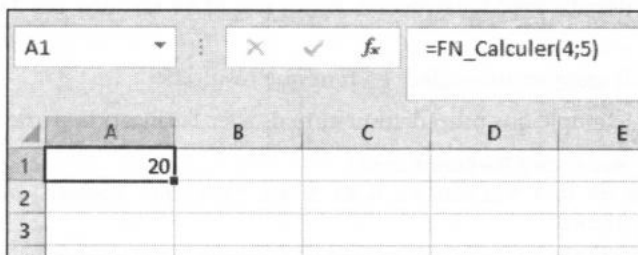
Résultat de l'exécution de la procédure MaProcEDUREFunction :



La fonction possède un avantage indéniable ; elle peut être utilisée également comme fonction sur une feuille Excel, comme sur l'exemple suivant :



Fonction FN_Calculer durant l'édition de la cellule A1



Fonction FN_Calculer durant l'exécution.

Événement

Un **événement** représente une action sur un objet au sein de programme. L'action peut être déclenchée par l'utilisateur comme un clic, ou par le système comme la modification de la valeur d'affichage.

L'événement peut ensuite conduire à l'appel ou l'exécution d'une procédure.

D'autres notions seront abordées progressivement via l'exemple.

Instructions

Une **instruction** est une commande donnée au système lors de l'exécution. Visual Basic for Application est basé sur une programmation événementielle, c'est-à-dire que c'est l'événement (un clic, une action) qui déclenche le code qui lui est associé. Le code est lu à partir du début de l'instruction jusqu'à la fin de l'instruction.

Les instructions sont diverses et il est nécessaire de savoir comment donner certaines instructions de base :

► Affecter une valeur

Affecter la valeur `Valeur_a_affecter` à une variable ou une propriété.

```
Sub Test
Variable = Valeur_a_affecter
Objet.Propriete = Valeur_a_affecter
End Sub
```

► Exécuter une fonction

Exécuter la fonction `MaFonction` qui possède deux arguments `Argument1` et `Argument2`

```
Sub Test
Variable = MaFonction(Argument1, Argument2)
End Sub
```

► Exécuter une procédure

Appeler l'exécution de la procédure `MaProcédure` au sein d'une autre procédure `Test`.

```
Sub Test
Call MaProcédure
End Sub
```

► Tester une condition

Tester si la variable `MaVariable` est supérieure ou égale à 5. Si oui, la variable est multipliée par 10, sinon `MaVariable` devient égale à 50.

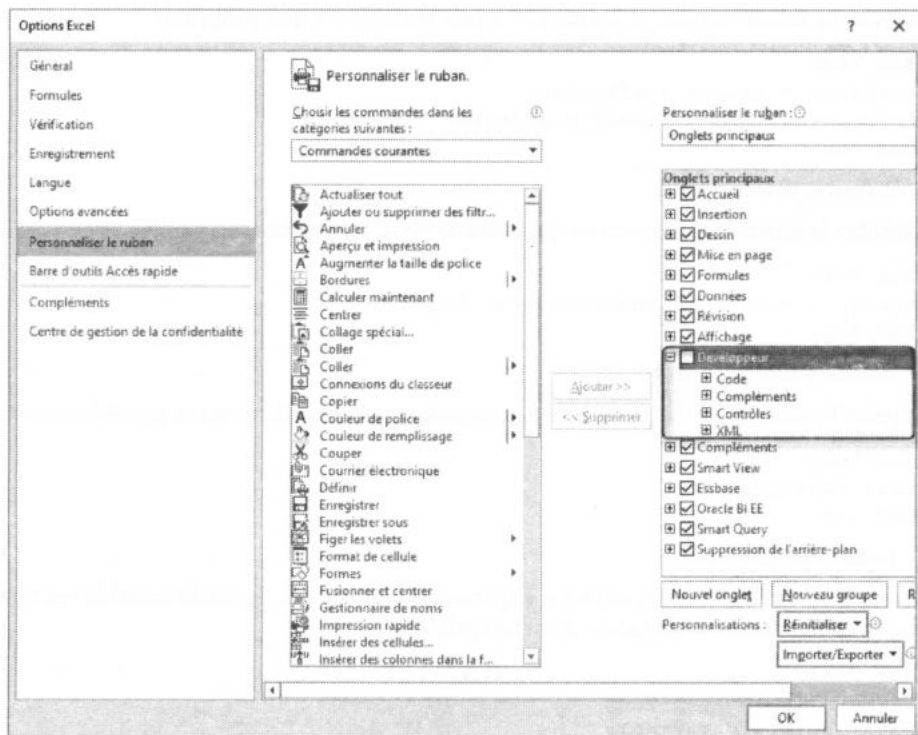
```
Sub Test
If MaVariable >= 5 Then
MaVariable = MaVariable * 10
Else
MaVariable = 50
End If
```


d. Le fonctionnement de l'éditeur Visual Basic

Accès à l'éditeur Visual Basic

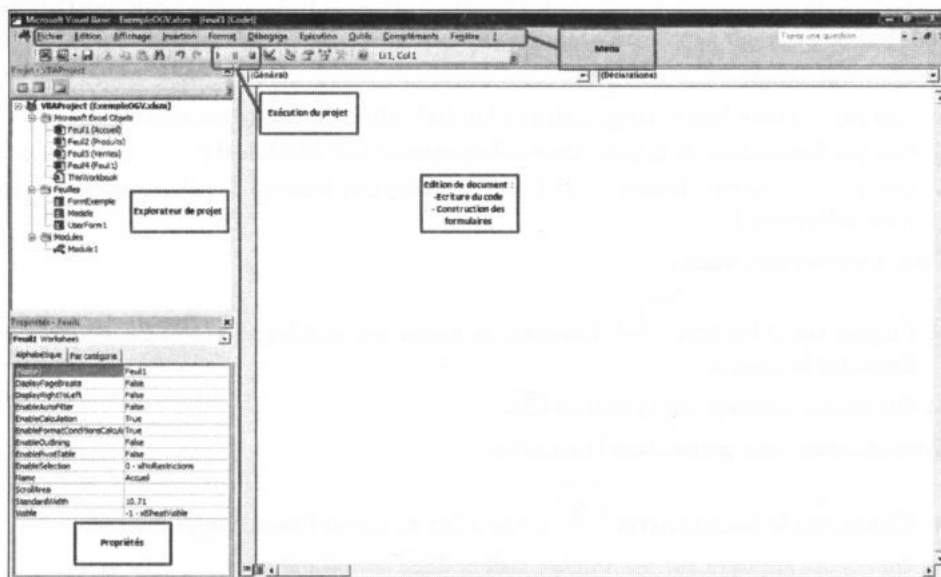
L'éditeur Visual Basic (nommé en anglais **Visual Basic Editor**) est l'interface de développement permettant de coder en Visual Basic. Pour y accéder, deux méthodes sont principalement utilisées :

- ▶ Raccourci-clavier : **Alt F11**
- ▶ Affichez l'onglet **Développeur** puis cliquez sur **Visual Basic**.
- ☞ Pour afficher l'onglet **Développeur** sous Excel 2013 ou 2016 : dans l'onglet **Fichier**, cliquez sur **Options** puis dans le sous-menu **Personnaliser le ruban**, cochez la case **Développeur**.



Présentation de l'éditeur Visual Basic.

Par défaut, l'éditeur se présente ainsi :



Présentation de l'éditeur Visual Basic

Menu : le menu permet d'accéder aux fonctionnalités disponibles comme pour n'importe quelle application. Le menu **Affichage** permet de faire apparaître les zones dont vous pourriez avoir besoin si vous ne les trouvez plus à l'écran.

Explorateur de projet : l'explorateur permet d'avoir l'arborescence du projet en cours. Vous y retrouverez vos feuilles Excel, vos formulaires et vos modules.

Exécution du projet : l'exécution vous permet de démarrer votre projet comme un utilisateur de l'application et non comme le concepteur.

Édition de document : cette zone permet de concevoir l'application. La conception consiste à soit écrire le code, soit déposer les éléments sur un formulaire.

Propriétés : la zone de propriété permet d'éditer les propriétés de l'élément sélectionné : soit un module/feuille/formulaire sélectionné dans l'explorateur de projet, ou un contrôle sélectionné dans la zone d'édition du document.


Exécution et débogage

Par défaut l'éditeur est en **Mode Création**, c'est-à-dire que vous êtes positionné sur l'interface de conception. L'exécution consiste à lancer le programme pour que l'utilisateur puisse interagir avec le programme.


L'exécution débute :

- ▶ Soit par la procédure correspondant à l'initialisation de la fenêtre sélectionnée ;
- ▶ Soit par l'exécution de la procédure sélectionnée (voir Modules) ;
- ▶ Soit par une macro choisie par l'utilisateur dans une fenêtre de sélection quand rien n'est sélectionné.


Pour exécuter une macro :

- ☞ Cliquez sur le bouton . **Exécuter la macro** ou accédez au menu **Exécution** puis **Exécuter la macro**.
- ☞ Ou encore appuyez sur la touche **F5**.

Pour effectuer une pause dans l'exécution :

- ☞ Cliquez sur le bouton **Arrêt**  ou accédez au menu **Exécution** puis **Arrêt** ;
- ☞ Ou encore appuyez sur les touches **Ctrl** et **Pause** simultanément.

Pour arrêter l'exécution :

- ☞ Cliquez sur le bouton **Réinitialiser**  ou accédez au menu **Exécution** puis **Réinitialiser**.
- ☞ Fermez la fenêtre active si une fenêtre est en cours d'exécution.

3. Réalisation de l'exemple

Toute la réalisation suivante va consister à créer l'outil de gestion des ventes. Les manipulations vont être décrites pas à pas.

- ☞ Commencez par ouvrir le fichier **Enoncé_3-A.xlsm**, puis accédez au Visual Basic Editor en appuyant simultanément sur les touches **Alt** et **F11** une fois le fichier ouvert.

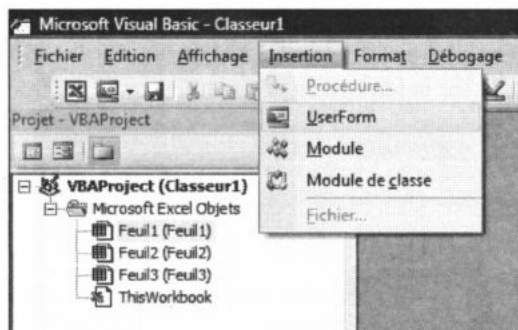
Vous êtes positionné sur le Visual Basic Editor du fichier **Enoncé_3-A.xlsm**.

a. Création du formulaire

Insertion du formulaire

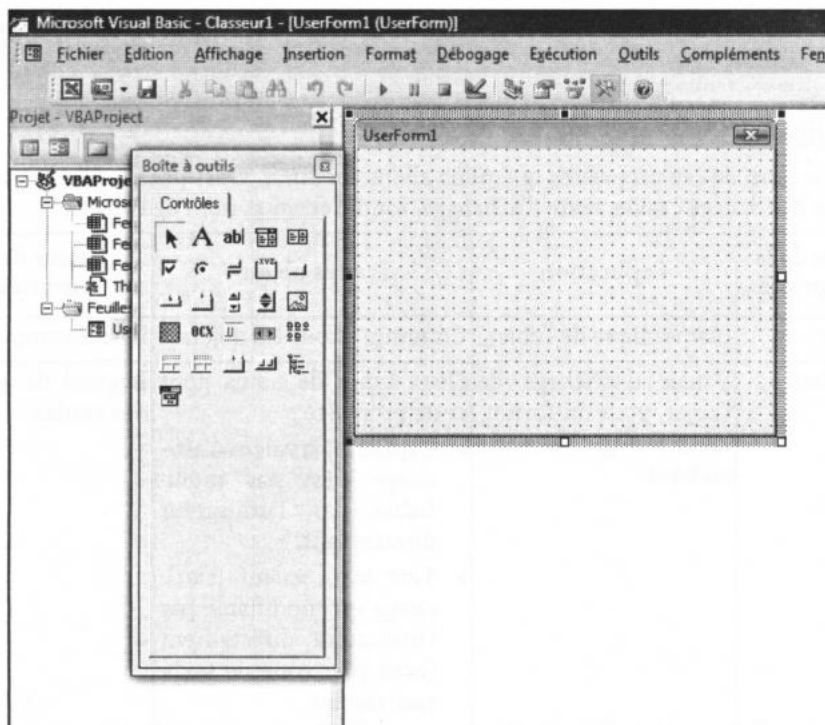
Dans cet exemple, un seul formulaire utilisateur est suffisant. Il contiendra l'ensemble des objets permettant de créer la facture.

- ☞ Afin de le créer dans l'éditeur visuel qu'est VBA, cliquez sur le menu **Insertion** puis **UserForm**.



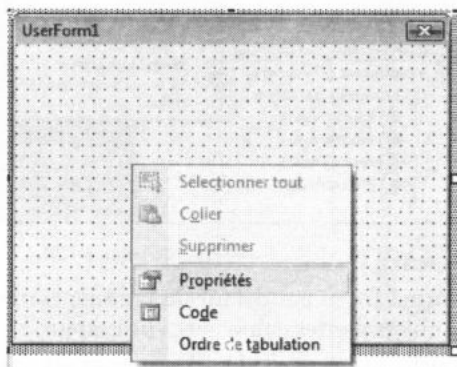
Deux éléments apparaissent à l'écran :

- ▶ Le formulaire UserForm1 (nom par défaut du premier formulaire) sur lequel il sera possible de créer des objets d'interaction avec l'utilisateur.
- ▶ La fenêtre Boîte à outils qui permet de sélectionner les contrôles à déposer sur le formulaire.



Pour afficher/modifier les propriétés du formulaire **UserForm1**.

- ☞ Sélectionnez le formulaire, faites un clic droit sur le formulaire puis cliquez sur **Propriétés** pour afficher la fenêtre de propriété.



Cette action est également possible via la touche **F4** ou le menu **Affichage puis Fenêtre Propriétés**.

Modification des propriétés

Voici le nom de ces propriétés que nous allons modifier. C'est principalement les propriétés d'affichage : taille, texte d'affichage, fond d'écran et nom de l'objet.

Nom de la propriété	Explication	Autres objets	Valeur de la propriété
Name	Nom unique de l'objet	Commun à tous les objets	FormExemple
Caption	Valeur d'affichage de l'objet, pour un formulaire, il s'agit du texte en haut.	Deux types de noms pour cette propriété : <ul style="list-style-type: none"> ▶ Caption si la valeur d'affichage n'est pas modifiable par l'utilisateur directement. ▶ Text si la valeur d'affichage est modifiable par l'utilisateur directement (exemple : zone de texte saisissable) 	Logiciel de gestion des ventes
Height	Hauteur (Taille)	Commun à tous les objets	400 (pixels)

Nom de la propriété	Explication	Autres objets	Valeur de la propriété
Width	Largeur (Taille)	Commun à tous les objets	220 (pixels)
BackColor	Couleur de fond : choix d'une couleur de fond de l'objet parmi la palette.	Commun à tous les objets	&H00FFFFFF&

☞ Modifiez ainsi les valeurs assignées aux propriétés pour obtenir le résultat suivant en exécution :



b. Création des contrôles sur le formulaire

Quels contrôles créer ?

À partir de la maquette fournie en entrée, il faut identifier les objets nécessaires à créer sur le formulaire nommé FormExemple.

Logiciel de gestion des ventes

Quantité 1 Produit 1

5 3 Sac taille XL 4

Ajouter à la ligne de commande 2

4*Sac taille L=80€
5*Sac taille XL=150€

6

Supprimer la ou les lignes sélectionnées 2

Total HT	230 €	<input type="checkbox"/> Remise 10 % pro	5
Total après remise	230 €	TVA	46 €
1	Total TTC	276 €	1

Sauvegarder et imprimer la commande 2

- ▶ 1 - les contrôles Label ;
 - ▶ 2 - les contrôles CommandButton ;
 - ▶ 3 - le contrôle TextBox ;
 - ▶ 4 - le contrôle Combobox ;
 - ▶ 5 - le contrôle CheckBox ;
 - ▶ 6 - le contrôle ListBox.
- ☞ Pour créer un objet, il suffit de le sélectionner dans la fenêtre Boite à outils puis de le déposer sur le formulaire (ici FormExemple).



Si la boîte à outils ne s'affiche pas, sélectionnez un formulaire puis cliquez sur le menu Affichage puis Boîte à outils.

La position de l'objet peut être déterminée par l'utilisateur sur l'éditeur graphique. Les propriétés `Left` (positionnement par la gauche) et la propriété `Top` (positionnement par le haut) ne seront pas abordées.

Création des contrôles Label

Les contrôles Label sont utilisés de deux façons différentes :

- ▶ Certains contrôles Label sont utilisés exclusivement pour de la présentation : ils n'ont aucune interaction avec le système.
- ▶ Certains contrôles Label sont utilisés dans le cadre d'un calcul : leur affichage sera modifié par l'application.

Ils vont être nommés différemment afin de bien les distinguer : `Aff_Nom_du_Label` pour un label d'affichage et `Calc_Nom_du_Label` pour un label utilisé par l'application.

N°	Label	Utilisation	Propriétés	
1	Quantité	Affichage du libellé Quantité au-dessus de la zone de texte saisissable.	Name	Aff_Produit
			Caption	Produit
2	Produit	Affichage du libellé Produit au-dessus de la liste déroulante permettant de sélectionner le produit.	Name	Aff_Produit
			Caption	Produit
3	Total HT (affichage)	Affichage du libellé Total HT à côté de la valeur du total hors taxes.	Name	Aff_TotalHT
			Caption	Total HT
4	Total HT (calcul)	Valeur calculée du total hors taxes.	Name	Calc_TotalHT
			Caption	0 €
5	Total après remise (affichage)	Affichage du libellé Total après remise à côté de la valeur du total après remise.	Name	Aff_TotApRemise
			Caption	Total après remise
6	Total après remise (Calcul)	Valeur calculée du total après remise.	Name	Calc_TotApRemise
			Caption	0 €
7	TVA (affichage)	Affichage du libellé TVA à côté de la valeur de la TVA	Name	Aff_TVA
			Caption	TVA

N°	Label	Utilisation	Propriétés	
8	TVA (Calcul)	Valeur calculée de la TVA	Name	Calc_TVA
			Caption	0 €
9	Total TTC (affichage)	Affichage du libellé Total TTC	Name	Aff_TotalTTC
			Caption	Total TTC
			ForeColor (couleur du texte)	&H000000FF& (rouge)
10	Total TTC (calcul)	Valeur calculée du total TTC	Name	Calc_TotalTTC
			Caption	0 €
			ForeColor (couleur du texte)	&H000000FF& (rouge)
			Font	Gras

☞ Créez les différents contrôles **Intitulé** pour obtenir le résultat suivant :

Logiciel de gestion des ventes ×

Quantité **1** Produit **2**

3 **4**
Total HT 0 €

5 **6** **7** **8**
Total après remise 0 € TVA 0 €

Total TTC 0 €
9 **10**

Création des boutons

Dans l'exemple, il y a trois boutons à créer :

N°	Label	Utilisation	Propriétés	
			Name	
1	Ajouter à la ligne de commande	Permet d'ajouter la ligne de commande dans la liste.	Name	AjLigComm
			Caption	Ajouter à la ligne de commande
2	Supprimer la ligne de commande	Permet de supprimer les lignes de commandes sélectionnées dans la liste	Name	SuppLigComm
			Caption	Supprimer la ou les lignes sélectionnées
3	Sauvegarder la facture et l'imprimer	Permet de sauvegarder la facture et de l'imprimer.	Name	SaveComm
			Caption	Sauvegarder et imprimer la commande.

☞ Dans la Boîte à outils, cliquez sur **Bouton de commande** pour ajouter les trois boutons.

Autres contrôles

Enfin pour terminer, il y a également quatre autres contrôles.

☞ Dans la Boîte à outils choisissez les contrôles suivants :

- ▶ 4 : Zone de texte
- ▶ 5 : Zone de liste modifiable
- ▶ 6 : Zone de liste
- ▶ 7 : Case à cocher

Logiciel de gestion des ventes

Quantité Produit

4 5

Ajouter à la ligne de commande 1

6

Supprimer la ou les lignes sélectionnées 2

Total HT 0 € Remise 10 % pro 7

Total après remise 0 € TVA 0 €

Total TTC 0 €

Sauvegarder et imprimer la commande 3

N°	Label	Utilisation	Propriétés	
			Name	Quantite
4	Quantité	Permet de saisir la quantité souhaitée.	Name	Quantite
			Text	vide
5	Liste des produits	Permet de sélectionner un produit parmi la liste des produits disponibles	Name	Produit
			Style (style de liste déroulante)	2 - fmStyle-DropDownList (oblige l'utilisateur à chercher une valeur dans la liste)

N°	Label	Utilisation	Propriétés	
			Name	ListeLigComm
6	Liste des lignes de commande	Afficher le détail de la facture avec l'ensemble des lignes de commande	Name	ListeLigComm
			MultiSelect (donne la possibilité à l'utilisateur de choisir plusieurs lignes)	1 - fmMultiSelect_Multi (sélection possible de plusieurs lignes, notamment pour la suppression)
7	Remise pro	Case à cocher pour déterminer si on applique une remise de 10 % au professionnel	Name	Remise
			Caption	Remise 10 % pro

c. Définition des procédures et événements

Les procédures présentes dans les macros vont détailler le code à exécuter. Toutefois celui-ci ne se déroulera qu'à la suite d'un événement.

Voici la liste des événements de l'application avec les actions associées. En gras les actions qui apparaissent plusieurs fois :

Initialisation de la feuille	Actions : <ul style="list-style-type: none"> ▶ Réinitialisation de la liste des produits ; ▶ Réinitialisation des objets : ▶ Zone de texte Quantité à vide ▶ Aucun produit sélectionné ▶ Aucun élément dans la liste des commandes
Clic sur le bouton pour ajouter une ligne de commande	Actions : <ul style="list-style-type: none"> ▶ Contrôle de la quantité saisie ▶ Contrôle de la sélection d'un produit ▶ Contrôle de la présence de stock ▶ Mise à jour du stock ▶ Ajout de la ligne de commande à la liste de la commande ▶ Mise à jour des totaux ▶ Affichage des totaux ▶ Affichage d'un message de confirmation de l'ajout.

Clic sur le bouton pour supprimer une ligne de commande	Actions : <ul style="list-style-type: none"> ▶ Vérification de la sélection d'une ligne de commande ▶ Mise à jour du stock ▶ Mise à jour des totaux ▶ Affichage des totaux ▶ Affichage d'un message de confirmation de la suppression
Clic sur le bouton pour valider la ligne de commande	Actions : <ul style="list-style-type: none"> ▶ Affichage d'un message de demande de confirmation à l'utilisateur ▶ Affichage d'un message de demande d'impression à l'utilisateur ▶ Réinitialisation de la liste des produits ▶ Réinitialisation des objets : <ul style="list-style-type: none"> ▶ Quantité à vide ▶ Aucun produit sélectionné ▶ Aucun élément dans la liste des commandes
Cocher la case de la remise	Actions : <ul style="list-style-type: none"> ▶ Mise à jour des totaux

Les actions à coder sont souvent appelées plusieurs fois dans l'application. Par conséquent, il faut créer plusieurs procédures indépendantes qui seront appelées par les différents événements grâce à l'instruction `Call`.

Voici la liste des procédures à créer :

- ▶ Initialisation du formulaire
- ▶ Mise à jour des stocks
- ▶ Ajouter une ligne de commande
- ▶ Mise à jour du montant total
- ▶ Supprimer une ou plusieurs lignes de commande
- ▶ Sauvegarder la facture
- ▶ Imprimer la facture.

La rédaction du code va être répartie en dix étapes permettant un meilleur découpage des fonctionnalités.

d. Rédaction du code : procédures et événements

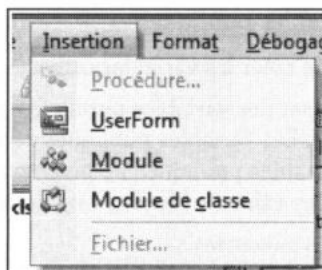
Voici une description des différentes étapes de la rédaction du code :

- ▶ Étape 1 : Création du module
- ▶ Étape 2 : Création des variables publiques
- ▶ Étape 3 : Initialisation du formulaire
- ▶ Étape 4 : Ajout d'une ligne de commande
- ▶ Étape 5 : Mise à jour des stocks
- ▶ Étape 6 : Mise à jour du montant total
- ▶ Étape 7 : Suppression d'une ou plusieurs lignes de commande
- ▶ Étape 8 : Sauvegarde de la facture
- ▶ Étape 9 : Relier les événements aux procédures
- ▶ Étape 10 : Création d'un bouton sur la feuille Accueil

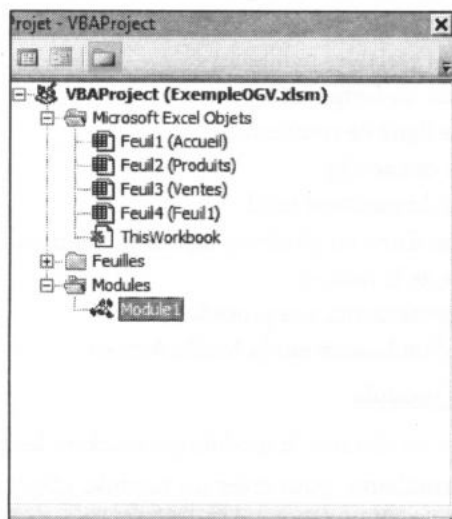
Étape 1 : Création du module

L'objectif de cette étape est de créer le module qui stockera les procédures.

- ☞ Comme pour les formulaires, pour créer un module, cliquez sur le menu **Insertion** puis choisissez **Module** comme indiqué ci-dessous :



Dans l'explorateur de projet, le module prend par défaut le nom Module1 et est affiché ainsi :



Étape 2 : Création des variables publiques

L'objectif de cette étape est de créer les variables utilisées tout au long de l'application.

Pour cet exemple, il faut utiliser des variables publiques pour stocker les valeurs saisies et effectuer les calculs dessus. Les variables publiques sont décrites en haut du module, avant les procédures. Les variables publiques ne peuvent pas être créées dans une procédure. Les variables publiques existent tout au long de l'exécution de l'application.

Il faut donc créer les variables suivantes :

Nom	Type	Détails
TotalHT	Double	Cette variable aura pour but de stocker le total hors taxe de la facture.
TotalTTC	Double	Cette variable aura pour but de stocker le total avec TVA de la facture.
TotalAprèsRemise	Double	Cette variable aura pour but de stocker le total hors taxe de la facture après l'application d'une éventuelle remise.

Le code s'écrit ainsi, avant les premières procédures.

```
Public TotalHT As Double
Public TotalTTC As Double
Public TotalApresRemise As Double
```

Étape 3 : Initialisation du formulaire

L'objectif de cette étape est de créer la procédure qui initialisera le formulaire.

La première action consiste à créer la procédure `Init`. Comme indiqué plus haut, elle démarre par `Sub Init` et termine par `End Sub`. Le code de la procédure sera entre le `Sub` et le `End Sub`

```
Sub Init()
'Code à insérer
End Sub
```

Ensuite, il faut initialiser la valeur d'affichage du contrôle `TextBoxQuantité`, définir que la case à cocher `Remise` de la remise est décochée et donner la valeur 0 à la variable `TotalHT`.

Par conséquent, il faut affecter la propriété `Text` à 0 pour le contrôle `Label` de la quantité et la propriété `Value` de la case à cocher du contrôle `CheckBox` de la remise sera à `False` (faux, vrai si c'est coché). La variable `TotalHT` aura également la valeur 0.

☞ Saisissez le code suivant au sein de la procédure `Init`.

```
'Quantité à 0
FormExemple.Quantite.Text = 0
'Remise non cochée :
FormExemple.Remise.Value = False
'Mettre le total hors taxe à 0
TotalHT = 0
```

La mise à jour des variables `TotalTTC` et `TotalApresRemise`, ainsi que l'affichage de leurs valeurs dans les contrôles `Label` s'effectueront dans une autre procédure. En effet, comme les actions de mise à jour et d'affichage de ces valeurs sont appelées plusieurs fois dans l'application, il est préférable de l'écrire dans une procédure différente qui sera appelée lorsque nécessaire. Cette procédure se nommera `AfficherTotal` (voir étape 6). Elle sera appelée avec l'instruction `Call`.

☞ Écrivez ensuite le code suivant.

```
'Appel à la procédure AfficherTotal qui permettra de calculer
les totaux et les afficher
Call AfficherTotal
```


Pour s'assurer qu'à chaque ouverture du formulaire, il n'y ait aucune trace de la vente précédente que ce soit les lignes de commande ou encore la liste des produits qui aurait pu évoluer (ajout d'un produit sur Excel ou stock d'un produit nul). Il faut effacer le contenu (les Items) du contrôle ListBox contenant la liste des lignes de commandes et effacer le contenu (les Items) du contrôle ComboBox contenant la liste des produits. La même méthode Clear propose d'effacer le contenu de la liste pour les contrôles ComboBox et ListBox.

☞ Effacez le contenu des contrôles de la manière suivante :

```
'Effacer le contenu de la liste de ligne de commande
FormExemple.ListeLigComm.Clear
'Mise à jour de la liste de produits (effacer puis ajouter les
éléments dans la liste).
'Suppression de la liste
FormExemple.Produit.Clear
```

Pour ajouter les produits de la feuille **Produits**, il faut récupérer la liste complète des produits saisis sur la feuille. La problématique est que nous ne savons pas exactement quelles valeurs récupérer.

Nous ne savons combien de produits nous allons récupérer ; en effet, il peut y avoir 2, 10 ou même plus de produits dans la liste. Par conséquent, il faut parcourir la feuille à partir de la **ligne 2** jusqu'à la dernière ligne de la liste des produits. La fin de la liste n'est pas déterminée, par conséquent nous déroulerons toutes les lignes jusqu'à trouver une ligne ou la cellule de la première colonne a une valeur nulle.

Nous allons donc introduire le concept de boucles. Les boucles permettent d'exécuter une portion de code plusieurs fois. Le nombre d'exécution dépend des conditions fixées par l'utilisateur.

Concept de boucle

Le **concept de boucle** permet de répéter une instruction un certain nombre de fois définies par les conditions de la boucle. Le code se situe entre l'instruction de début et l'instruction de fin de la boucle.

Il existe plusieurs types de boucle :

Instruction de début de boucle	Instruction de fin de boucle	Condition de sortie de la boucle	Exemple
For	Next	Au niveau de l'instruction For, le nombre d'itérations est défini avec une variable	For I = 1 to 10 Next S'exécute 10 fois.

Instruction de début de boucle	Instruction de fin de boucle	Condition de sortie de la boucle	Exemple
While	Wend	La condition de sortie de la boucle est définie au niveau de l'instruction While. Une condition est posée, si elle est vraie, s'arrête la boucle.	<pre>I = 1 While I = 5 I = I + 1 Wend</pre> S'exécute 4 fois.
Do	Loop	La condition de sortie n'est pas exprimée dans une instruction de la boucle. Déconseillée.	<pre>I = 0 Do I = I - 1 If I = -3 Then Exit Do Loop</pre> S'exécute 3 fois puis sort.
For Each	Next	La boucle parcourt l'ensemble des objets d'une collection d'objets. La boucle se termine lorsque l'ensemble des objets a été parcouru.	<pre>For Each Sh in Sheets Next</pre> S'exécute autant de fois qu'il y a de feuille dans le classeur.

Dans le cas d'une boucle de type `While-Wend` où `While` est l'instruction de début et `Wend` est l'instruction de fin, la boucle s'exécute tant que la condition décrite suite à l'instruction `While` est vraie.

Pour parcourir les lignes, il est nécessaire de créer une variable `Ligne` de type nombre entier qui sera initialisée à 2 (première ligne où il y a un produit dans la feuille `Produits`). À chaque itération de la boucle, la valeur de la variable `Ligne` sera incrémentée de 1, ce qui permettra d'analyser la ligne suivante.

```
Cells(Ligne, 1).value
```

- ▶ Si la variable `Ligne` est égale à 4, l'instruction ci-dessus remonte la valeur de la cellule `A4`.
- ▶ Si la variable `Ligne` est égale à 8, l'instruction ci-dessus remonte la valeur de la cellule `A8`.

La boucle s'écrit ainsi :

```
'sélection de la feuille Produits
Sheets("Produits").Select
'création de la variable Ligne
```

```
Dim Ligne As Integer
Ligne = 2
'On parcourt la feuille Produits pour chercher tous les
produits disponibles
While Cells(Ligne, 1).Value <> "" 'tant que la cellule est
différente de nul
    'insérer le code
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur
de 1 pour la variable Ligne à chaque passage
Wend
```

Utilisation de l'instruction conditionnelle

L'ajout d'un élément à la liste se fera sous la condition qu'il y ait suffisamment de stock, par conséquent il faut tester si la valeur du stock est suffisante (supérieure à 0). Nous allons utiliser l'instructions conditionnelle permettant d'effectuer une liste d'instructions si la condition est remplie et d'effectuer une autre liste d'instructions si la condition n'est pas remplie :

```
If 'condition Then
'code si condition OK
Else
'code si condition non remplie
End if
```



Le Else n'est pas obligatoire et l'instruction peut être écrite sur une seule ligne de la manière suivante :

```
If 'condition Then 'code si condition OK
```

La dernière étape consiste à ajouter le produit dans le contrôle ComboBox contenant la liste des produits. Pour cela, il est nécessaire d'utiliser la méthode AddItem(valeur) où l'argument valeur correspond à ce qui va être ajouté dans la liste. On obtient donc le code suivant à l'intérieur de la boucle :

```
While Cells(Ligne, 1).Value <> "" 'tant que la cellule est
différente de nul
    If Cells(Ligne, 3).Value > 0 Then 'on teste si le stock est
supérieur à 0
        FormExemple.Produit.AddItem (Cells(Ligne, 1).Value)
'ajout de la valeur de la cellule dans la liste Produit.
    End If
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur
de 1 pour la variable Ligne à chaque passage
Wend
```

La procédure entièrement rédigée est donc la suivante :

```

Sub Init()
'Quantité à 0
FormExemple.Quantite.Text = 0
'Remise non cochée :
FormExemple.Remise.Value = False
'Mettre le total hors taxe à 0
TotalHT = 0
'Appel à la procédure afficher total qui permettra de calculer
les totaux et les afficher
Call AfficherTotal
'Effacer le contenu de la liste de ligne de commande
FormExemple.ListeLigComm.Clear
'Mise à jour de la liste de produit (effacer puis ajouter les
éléments dans la liste).
'Suppression de la liste
FormExemple.Produit.Clear
'Ajout des produits dans la liste
Sheets("Produits").Select
Dim Ligne As Integer
Ligne = 2
'Parcourir la feuille Produits pour chercher tous les produits
disponibles
While Cells(Ligne, 1).Value <> "" 'tant que la cellule est
différente de nul
    If Cells(Ligne, 3).Value > 0 Then 'tester si le stock est
supérieur à 0
        FormExemple.Produit.AddItem (Cells(Ligne, 1).Value)
    End If
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur
de 1 pour la variable Ligne à chaque passage
Wend
End Sub

```

Étape 4 : Ajout d'une ligne de commande

L'objectif de cette étape est d'ajouter une ligne de commande à la liste de la commande en cours. La ligne de commande se compose d'une quantité et d'un produit. Une fois saisie, la ligne de commande pourra être ajoutée à la liste de la commande en cours grâce au bouton **Ajouter la ligne de commande**.

L'ajout d'une ligne de commande est une étape assez longue puisqu'il va falloir effectuer différents contrôles sur les champs quantité et produit. Pour cela, nous allons créer une procédure nommée `AjouterLigneCommande` qui effectuera l'ensemble des contrôles.

La première étape consiste à déclarer les variables nécessaires dans cette procédure :

Nom	Type	Détails
TotalLigneCommande	Double	A pour but de stocker la valeur de la ligne de commande (prix * quantité), valeur nulle par défaut
Libelle	String	Cette variable aura pour but de stocker le libellé de la ligne de commande à ajouter dans la liste des lignes de commande. Champ non renseigné par défaut
EstErreur	Boolean	Indicateur Vrai/Faux permettant de savoir si une erreur fonctionnelle a été rencontrée durant l'exécution du code
LibelleErreur	String	Message d'erreur. Champ non renseigné par défaut
ProduitString	String	Produit récupéré de la ComboBox
QuantiteString	String	Quantité récupérée de la TextBox
QuantiteInteger	Integer	Quantité convertie en nombre entier

Valeur par défaut

La valeur par défaut d'une variable est nulle, cela se matérialise par un "" (double guillemets, c'est-à-dire champ vide) pour une variable de type `String`, une valeur `False` pour une variable de type `Boolean` et un 0 pour une variable de type nombre (`Integer`, `Long`, `Double`...). Pour la procédure en cours, les valeurs par défaut conviennent parfaitement et il n'est pas nécessaire d'initialiser les variables différemment.

☞ Décrivez le début de la procédure ainsi :

```
Sub AjouterLigneCommande()
Dim TotalLigneCommande As Double
Dim Libelle As String
Dim EstErreur As Boolean
Dim LibelleErreur As String
Dim ProduitString As String
Dim QuantiteString As String
Dim QuantiteInteger As Integer
'Suite du code
End Sub
```

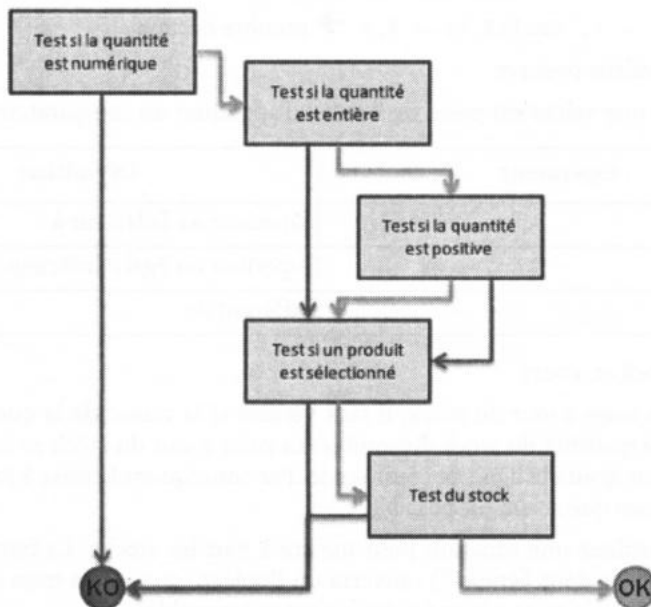
Une série de contrôles va être effectuée pour tester si l'ajout de la ligne de commande peut correctement se faire.

En effet, la quantité saisie doit être un entier numérique positif et un produit doit être sélectionné dans la liste. Enfin, il faut s'assurer que la quantité saisie est inférieure ou égale au stock en cours pour ne pas être en rupture de stock.

Les contrôles successifs sont les suivants :

- ▶ Tester si la valeur saisie par l'utilisateur pour le contrôle Quantité est un numérique ;
- ▶ Tester si la valeur saisie par l'utilisateur pour le contrôle Quantité est un entier ;
- ▶ Tester si la valeur saisie par l'utilisateur pour le contrôle Quantité est positif ;
- ▶ Tester si un élément est bien sélectionné dans la liste déroulante Produits ;
- ▶ Tester si le stock du produit sélectionné est supérieur à la valeur saisie par l'utilisateur pour le contrôle Quantité.

Les tests seront ordonnancés de la manière suivante :



En cas d'erreur, la variable de type **Boolean** `EstErreur` prendra la valeur `True` et le libellé d'erreur `LibelleErreur` aura le message d'erreur. Cela se code de la manière suivante :

```

EstErreur = True
LibelleErreur = LibelleErreur & "Contenu de l'erreur
rencontrée." & Chr(13) 'ajout d'un saut le ligne à la fin

```

☞ Test de la valeur numérique

Pour tester la valeur numérique, il faut utiliser l'opération `IsNumeric(valeur)` qui renvoie le Boolean `True` (Vrai) si la valeur est numérique, et qui renvoie `False` (Faux) si la valeur n'est pas numérique. Dans le cas actuel, il faut afficher un message d'erreur si la valeur n'est pas numérique.

🔗 Test de la valeur entière

Pour tester si une valeur numérique est un nombre entier, la solution consiste à utiliser deux convertisseurs de valeur :

- ▶ `CInt(valeur)` convertit la valeur en `Integer` ;
- ▶ `CDBl(valeur)` convertit la valeur en `Double`.

Par conséquent, si `CInt(valeur)` est égal à `CDBl(valeur)`, cela signifie qu'il est entier :

- ▶ `CInt(4) = 4 ; CDBl(4) = 4` → nombre entier ;
- ▶ `CInt(4.5) = 4, CDBl(4.5) = 4.5` → nombre décimal.

🔗 Test de la valeur positive

Pour tester si une valeur est positive, il suffit d'appliquer un comparateur logique :

Opérateur	Définition
> / <	Supérieur à / Inférieur à
>= / <=	Supérieur ou égal / Inférieur ou égale
<>	Différent de

🔗 Test du stock en cours

Pour tester la mise à jour du stock, il faut vérifier si la valeur de la quantité saisie ne dépasse pas la quantité du stock disponible. La mise à jour du stock se fera instantanément lorsqu'on ajoute la ligne de commande. Par conséquent la mise à jour du stock ne doit être réalisée que si elle est possible.

Nous allons utiliser une fonction pour mettre à jour les stocks. La fonction nommée `MAJStock` (définie dans l'étape 5) renverra un Boolean : `True` si la mise à jour du stock est OK et `False` si la mise à jour du stock n'est pas possible.

Le code des contrôles est le suivant :

```
'Contrôler la quantité
QuantiteString = FormExemple.Quantite.Text
'Tester si la valeur du champ Quantite est bien numérique avec
l'opérateur IsNumeric
'Not IsNumeric = True signifie que la valeur testée n'est pas
numérique
If Not IsNumeric(QuantiteString) = True Then
    EstErreur = True
    LibelleErreur = LibelleErreur & "Le format de la quantité
```

```

saisie n'est pas numérique." & Chr(13) 'ajout d'un saut de
ligne à la fin
Else
'Tester si la quantité saisie est un nombre entier
  If Cint(QuantiteString) <> CDbI(QuantiteString) Then
'comparer entre la quantité convertie en nombre entier et la
quantité saisie
  EstErreur = True
  LibelleErreur = LibelleErreur & "Le format de la
quantité saisie n'est pas un nombre entier." & Chr(13)
  Else
'Convertir la quantité en nombre entier
  QuantiteInteger = Cint(QuantiteString)
'Tester si la quantité est inférieure à 1
  If QuantiteInteger < 1 Then
    EstErreur = True
    LibelleErreur = LibelleErreur & "La quantité saisie
ne peut pas être négative ou nulle." & Chr(13)
  End If
  End If
'Récupérer le produit sélectionné dans la liste déroulante
  ProduitString = FormExemple.Produit.Value
'Tester qu'un produit est bien sélectionné
  If ProduitString = "" Then
    EstErreur = True
    LibelleErreur = LibelleErreur & "Vous devez choisir un
produit." & Chr(13)
  End If
'Tester la mise à jour du stock en appelant la fonction
MAJStock. Si le retour est OK, la mise à jour est prise en compte.
Cette opération n'est possible que s'il n'y a pas d'erreur jusque-là.
  If EstErreur = False Then
    If MAJStock(ProduitString, QuantiteInteger) = False
Then
      EstErreur = True
      LibelleErreur = LibelleErreur & "Pas assez de stock
en cours." & Chr(13)
    End If
  End If
End If

```

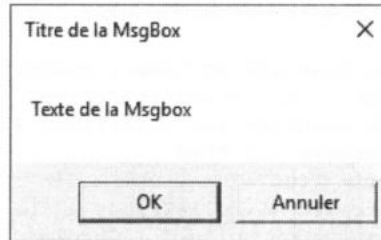
Une fois les contrôles terminés, deux scénarios sont possibles :

- ▶ Soit il y a une erreur et il faut afficher le message d'erreur ;
- ▶ Soit les contrôles sont OK et il faut ajouter l'élément à la liste.

Cas où un message d'erreur est affiché

Dans le cas où il y a un message d'erreur, il faut afficher une boîte de dialogue MsgBox. La boîte de dialogue de type MsgBox permet d'afficher un pop-up contenant un intitulé, un titre et surtout des boutons ce qui permet à l'utilisateur d'interagir avec l'application. La syntaxe est la suivante :

- ▶ `MaVariable = MsgBox(Texte de la MsgBox, Type de bouton (vbOKCancel), Titre de la MsgBox)`

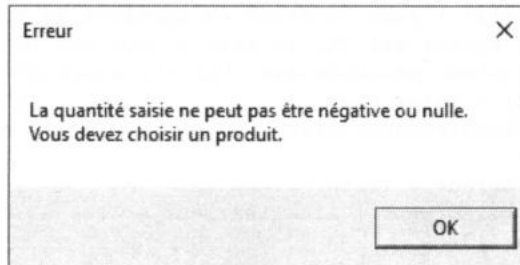


Le code se présente donc de la manière suivante :

```
'Affichage d'un message d'erreur  
ValErreur = MsgBox(LibelleErreur, vbOKOnly, "Erreur")
```

Le libellé de cette boîte de dialogue est la valeur de la variable `LibelleErreur`. Enfin, un seul bouton sera présent sur cette boîte de dialogue puisque pour le type de bouton, nous avons saisi la valeur `vbOKOnly` qui signifie qu'il n'y a que le bouton OK.

Cela peut se matérialiser ainsi :



Cas où il n'y a pas d'erreur

Dans le cas où il n'y a pas d'erreur, il faudra :

- ▶ Calculer le total de la ligne de commande ;
- ▶ Créer le libellé de la ligne de commande ;
- ▶ Ajouter la ligne de commande à la liste ;
- ▶ Mettre à jour le TotalHT ;
- ▶ Calculer les totaux ;

- Afficher le message de confirmation.

Calculer le total de la ligne de commande

Pour calculer le total de la ligne de commande, il faut affecter à la variable `TotalLigneCommande` le produit de la variable `QuantiteInteger` avec le prix. Pour récupérer le prix, il est nécessaire de créer une fonction permettant de récupérer le prix avec la référence du produit qui sera nommée `RecupererPrix(Produit)`.

Le détail de la fonction `RecupererPrix` est expliqué après cette procédure..

- ✎ Rédigez le code ainsi :

```
'Calcul du montant de la ligne de commande / utilisation de la
fonction récupérerPrix pour avoir le prix par produit
    TotalLigneCommande = QuantiteInteger *
RecupererPrix(ProduitString)
```

Avec les champs : produit, quantité et prix total de la ligne, il est possible de créer le texte qui sera ajouté à la liste de ligne de commandes.



L'opérateur & permet de concaténer des chaînes de caractères.

Création du libellé de la ligne de commande

```
'Création de la ligne de commande au format texte
    Libelle = QuantiteInteger & "*" & ProduitString & "=" &
TotalLigneCommande & "€"
```

Ajouter la ligne de commande à la liste

Ensuite, il convient d'utiliser la méthode `AddItem (valeur)` du contrôle `ListBox ListeLigComm` pour ajouter le libellé ci-dessus dans la liste des commandes.

La méthode `AddItem` s'applique ainsi :

```
ObjetListBox.AddItem Item
```

Dans le cadre de notre procédure, rédigez la syntaxe suivante :

```
'Ajout de la ligne de commande à la liste
FormExemple.ListeLigComm.AddItem (Libelle)
```

Mettre à jour des totaux et message de confirmation :

Pour terminer, la mise à jour des totaux est réalisée suite à la somme de `TotalLigneCommande` avec la variable publique `TotalHT`.

- ✎ Appelez ensuite la procédure `AfficherTotal` qui permettra d'afficher les totaux dans le formulaire. Cette procédure sera détaillée dans l'étape 6.

En fin de traitement, un message via une fenêtre `Msgbox` signale à l'utilisateur que l'opération s'est correctement déroulée. Cette boîte de dialogue ne propose que le bouton « OK ».

```
'Mise à jour du Total HT puis calcul des totaux
  TotalHT = TotalHT + TotalLigneCommande
  Call AfficherTotal
  'Message de confirmation OK
  ValOK = MsgBox("Ligne ajoutée", vbOKOnly, "Info")
```

Voici le bloc de code dans son ensemble :

```
'Les contrôles sont finis, soit ajouter la valeur à la liste,
soit afficher l'erreur
If EstErreur Then
  'Affichage d'un message d'erreur
  ValErreur = MsgBox(LibelleErreur, vbOKOnly, "Erreur")
Else
  'Calcul du montant de la ligne de commande / utilisation de
la fonction récupérerPrix pour avoir le prix par produit
  TotalLigneCommande = QuantiteInteger *
  RecupererPrix(ProduitString)
  'Création de la ligne de commande au format texte
  Libelle = QuantiteInteger & "*" & ProduitString & "=" &
TotalLigneCommande & "€"
  'Ajout de la ligne de commande à la liste
  FormExemple.ListeLigComm.AddItem (Libelle)
  'Mise à jour du Total HT puis calcul des totaux
  TotalHT = TotalHT + TotalLigneCommande
  Call AfficherTotal
  'Message de confirmation OK
  ValOK = MsgBox("Ligne ajoutée", vbOKOnly, "Info")
End If
End Sub
```

La procédure `AjouterLigneCommande` est donc la suivante :

```
Sub AjouterLigneCommande()
  Dim TotalLigneCommande As Double
  Dim Libelle As String
  Dim EstErreur As Boolean
  Dim LibelleErreur As String
  Dim ProduitString As String
  Dim QuantiteString As String
  Dim QuantiteInteger As Integer

  'Contrôler la quantité
  QuantiteString = FormExemple.Quantite.Text
  'Tester si la valeur du champ Quantite est bien numérique avec
```

```

l'opérateur IsNumeric
'Not IsNumeric = True signifie que la valeur testée n'est pas
numérique
If Not IsNumeric(QuantiteString) = True Then
    EstErreur = True
    LibelleErreur = LibelleErreur & "Le format de la quantité
saisie n'est pas numérique." & Chr(13) 'ajout d'un saut de
ligne à la fin
Else
'Tester si la quantité saisie est un nombre entier
    If CInt(QuantiteString) <> CDBl(QuantiteString) Then
'comparer la quantité convertie en nombre entier avec la
quantité saisie
        EstErreur = True
        LibelleErreur = LibelleErreur & "Le format de la quantité
saisie n'est pas un nombre entier." & Chr(13)
    Else
'Convertir la quantité en nombre entier
        QuantiteInteger = CInt(QuantiteString)
'Tester si la quantité est inférieure à 1
        If QuantiteInteger < 1 Then
            EstErreur = True
            LibelleErreur = LibelleErreur & "La quantité saisie
ne peut pas être négative ou nulle." & Chr(13)
        End If
    End If
'Récupérer le produit sélectionné dans la liste déroulante
    ProduitString = FormExemple.Produit.Value
'Tester qu'un produit est bien sélectionné
    If ProduitString = "" Then
        EstErreur = True
        LibelleErreur = LibelleErreur & "Vous devez choisir un
produit." & Chr(13)
    End If
'Tester la mise à jour du stock en appelant la fonction
MAJStock. Si le retour est OK, la mise à jour est prise en compte.
Cette opération n'est possible que s'il n'y a pas d'erreur rencontrée
jusque-là.
    If EstErreur = False Then
        If MAJStock(ProduitString, QuantiteInteger) = False Then
            EstErreur = True
            LibelleErreur = LibelleErreur & "Pas assez de stock
en cours." & Chr(13)
        End If
    End If
End If
'Les contrôles sont finis, soit ajouter la valeur à la liste,

```

```

    soit afficher l'erreur
If EstErreur Then
    'Affichage d'un message d'erreur
    ValErreur = MsgBox(LibelleErreur, vbOKOnly, "Erreur")
Else
    'Calcul du montant de la ligne de commande / utilisation de
    la fonction récupérerPrix pour avoir le prix par produit
    TotalLigneCommande = QuantiteInteger *
    RecupererPrix(ProduitString)
    'Création de la ligne de commande au format texte
    Libelle = QuantiteInteger & "*" & ProduitString & "=" &
    TotalLigneCommande & "€"
    'Ajout de la ligne de commande à la liste
    FormExemple.ListeLigComm.AddItem (Libelle)
    'Mise à jour du Total HT puis calcul des totaux
    TotalHT = TotalHT + TotalLigneCommande
    Call AfficherTotal
    'Message de confirmation OK
    ValOK = MsgBox("Ligne ajoutée", vbOKOnly, "Info")
End If
End Sub

```

► Fonction Récupérer le prix

Comme vu précédemment, une fonction va être utilisée pour récupérer le prix d'un produit.

La fonction possède en argument d'entrée le produit.

La fonction de récupération des prix permet de parcourir la feuille **Produits** pour récupérer le prix du produit sélectionné.

La fonction a pour valeur de retour le prix du produit.

```

| PrixProduit = RecupererPrix(Produit)

```

☞ Commencez par créer la fonction `RecupererPrix` avec en argument le nom du produit dans le `Module1` comme écrit ci-dessous :

```

| Function RecupererPrix(Produit As String)
| End Function

```

Au début de la fonction, il faut lui affecter la valeur nulle.

```

| RecupererPrix = 0

```

Ensuite, il faut parcourir la feuille **Produits** avec une boucle. Cette boucle doit avoir deux conditions de sortie :

- La liste des produits est terminée sans qu'on ait trouvé le produit correspondant ;
- Le produit a été récupéré.

Dans ce cas, la boucle `while-wend` convient avec comme condition de maintien dans la boucle : la valeur de la cellule analysée est différente du nom du produit.

Dans le cas où le produit ne figure plus dans la liste, un test est positionné afin de quitter la fonction en cours si la cellule analysée est vide. Si c'est le cas, le résultat de la fonction serait 0

L'instruction `Exit Function` est utilisée afin de sortir de la fonction :

```
If Cells(Ligne,1).Value = "" then Exit Function 'sortie de la  
fonction si rencontrer une valeur nulle.
```

Cela donne la boucle suivante :

```
Sheets("Produits").Select  
Dim Ligne As Integer  
Ligne = 2  
While Cells(Ligne, 1).Value <> Produit 'tant que la cellule est  
différente du produit le programme continue  
    If Cells(Ligne,1).Value = "" then Exit Function 'sortie de  
la fonction s'il y a une valeur nulle  
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur  
de 1 pour la variable Ligne à chaque passage  
Wend
```

À la sortie de la boucle, la variable `Ligne` va correspondre à la ligne où se situe le produit dans la feuille **Produits**. Par conséquent, pour récupérer le prix, il suffira d'affecter la valeur du prix (colonne 2) à la fonction `RecupererPrix` de la manière suivante :

```
RecupererPrix = Cells(Ligne, 2).Value
```

Voici la fonction dans son ensemble :

```
Function RecupererPrix(Produit As String)  
RecupererPrix = 0  
Sheets("Produits").Select  
Dim Ligne As Integer  
Ligne = 2  
While Cells(Ligne, 1).Value <> Produit ' tant que la cellule  
est différente du produit le programme continue  
    If Cells(Ligne,1).Value = "" then Exit Function 'sortie de  
la fonction s'il y a une valeur nulle  
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur  
de 1 pour la variable Ligne à chaque passage  
Wend  
RecupererPrix = Cells(Ligne, 2).Value  
End Function
```

Étape 5 : Mise à jour des stocks

La fonction mise à jour des stocks possède les éléments suivants en argument d'entrée :

- ▶ Quantité ;
- ▶ Produit.

La fonction de mise à jour du stock va effectuer les actions suivantes :

- ▶ Rechercher le produit en entrée dans la feuille **Produits** ;
- ▶ Retirer du stock la quantité en argument d'entrée de la fonction.

La fonction de mise à jour du stock renvoie une valeur de type Boolean qui correspond au statut de l'opération de la mise à jour du stock :

- ▶ Elle renvoie la valeur `True` (vrai) si la mise à jour s'est correctement réalisée ;
- ▶ Elle renvoie la valeur `False` (faux) si la mise à jour ne s'est pas correctement réalisée.

☞ Créez la fonction `MAJStock` avec pour argument le `Produit` et la `Quantité`.

```
Function MAJStock(Produit As String, Quantite As Integer)
End Function
```

☞ Initialisez la valeur de la fonction à la valeur `False`. Elle passera à `True` dès que la mise à jour du stock est réussie.

```
'Affectation de la valeur par défaut à False
MAJStock = False
```

Ensuite, il faut parcourir la feuille **Produits** avec une boucle.

☞ Utilisez la boucle `While-Wend`. Restez dans la boucle tant que le produit parcouru (colonne A) est différent du produit recherché.

```
While Cells(Ligne, 1).Value <> Produit
End While
```

À l'intérieur de la boucle, si la liste des produits est terminée, sortez de la boucle avec l'instruction `Exit`.

```
If Cells(Ligne,1).Value = "" Then Exit Function 'sortie de
la fonction s'il y a une cellule vide
```

Le code sera le suivant :

```
'Sélection de la feuille Produits
Sheets("Produits").Select
'Définition de la variable Ligne permet de parcourir la
feuille.
Dim Ligne As Integer
Ligne = 2
While Cells(Ligne, 1).Value <> Produit ' tant que la cellule
est différente du produit le programme continue
If Cells(Ligne,1).Value = "" Then Exit Function 'sortie de
```

la fonction s'il y a sur une cellule vide.

```
Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur
de 1 pour la variable Ligne à chaque passage
Wend
```

À la sortie de la boucle, la variable `Ligne` va correspondre à la ligne où se situe le produit dans la feuille **Produits**.

La variable `StockEnCours` sera utilisée afin de faire le calcul du nouveau stock. Si le stock est négatif, l'opération ne sera pas validée. Si le nouveau stock est supérieur ou égal à 0, l'opération sera prise en compte :

```
StockEnCours = Cells(Ligne, 3).value 'attribution de la
variable à la quantité actuelle
StockEnCours = StockEnCours - Quantite 'calcul du nouveau stock
If StockEnCours >= 0 Then
    Cells(Ligne,3).value = StockEnCours 'affectation du le
nouveau stock au produit
    MAJStock = True 'La fonction est OK et prend la valeur
True
End if
```

La fonction une fois terminée se présente ainsi :

```
Function MAJStock(Produit As String, Quantite As Integer)
Dim StockEnCours As Integer
'Affectation de la valeur par défaut à False
MAJStock = False
'Sélection de la feuille Produits
Sheets("Produits").Select
'Définition de la variable Ligne permet de parcourir la
feuille.
Dim Ligne As Integer
Ligne = 2
While Cells(Ligne, 1).Value <> Produit 'tant que la cellule est
différente du produit le programme continue
    If Cells(Ligne,1).Value = "" Then Exit Function 'sortie de
la fonction s'il y a une cellule vide.
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la valeur
de 1 pour la variable Ligne à chaque passage
Wend
'sortie de la boucle : cela signifie que la variable ligne
correspond à la ligne où se trouve le produit
StockEnCours = Cells(Ligne, 3).value 'attribution de la
variable à la quantité actuelle
StockEnCours = StockEnCours - Quantite 'calcul du nouveau stock
If StockEnCours >= 0 Then
    Cells(Ligne,3).value = StockEnCours 'On affecte le nouveau
stock au produit
```



```
MAJStock = True 'La fonction est OK et prend la valeur True
End if
End Function
```

Étape 6 : Mise à jour du montant total

L'objectif de cette étape est de créer une procédure permettant de mettre à jour les différentes variables contenant des totaux et d'afficher les valeurs au sein d'un formulaire.

Il faut donc créer une procédure `AfficherTotal` qui sera appelée lors de la mise à jour des différents totaux dans l'application (voir événements).

Cette procédure se décompose en deux parties.

► Partie Calcul

Une variable va être créée pour calculer le montant après remise :

Nom	Type	Détails
Coeff	Double	Prendra la valeur 1 s'il n'y a pas de remise, prendra la valeur 0.9 si une remise de 10 % est appliquée.

```
Dim Coeff As Double
Coeff = 1
If FormExemple.Remise.Value = True Then 'on teste la case à
cocher de la remise
    Coeff = 0.9
End If
```

La variable `TotalHT` sera utilisée pour calculer les autres totaux.

```
TotalApresRemise = TotalHT * Coeff 'Coefficient d'application
de la remise (1 ou 0.9)
TVA = TotalApresRemise * 0.2 '(application du taux de TVA de
20%)
TotalTTC = TotalApresRemise + TVA
```

► Partie Affichage des valeurs

Pour afficher les valeurs, il suffit d'affecter la valeur de la variable à la propriété `Caption` des contrôles `Label` concernés.

```
'Mise à jour des valeurs à l'affichage sur l'écran
FormExemple.Calc_TotalHT.Caption = TotalHT & " €"
FormExemple.Calc_TotalTTC.Caption = TotalTTC & " €"
FormExemple.Calc_TotApRemise.Caption = TotalApresRemise & " €"
FormExemple.Calc_TVA.Caption = TVA & " €"
```

✎ Écrivez la procédure `AfficherTotal` ainsi :

```
Sub AfficherTotal()
'Le coeff sera 1 si pas de remise et 0.9 si 10% de remise
Dim Coeff As Double
Coeff = 1
If FormExemple.Remise.Value = True Then 'Tester la case à
cocher de la remise
    Coeff = 0.9
End If
'Mise à jour des différentes variables
TotalApresRemise = Coeff * TotalHT
TVA = TotalApresRemise * 0.2
TotalTTC = TotalApresRemise + TVA
'Mise à jour des valeurs à l'affichage sur l'écran
FormExemple.Calc_TotalHT.Caption = TotalHT & " €"
FormExemple.Calc_TotalTTC.Caption = TotalTTC & " €"
FormExemple.Calc_TotApRemise.Caption = TotalApresRemise & " €"
FormExemple.Calc_TVA.Caption = TVA & " €"
End Sub
```

Étape 7 : Suppression d'une ou plusieurs lignes de commande

L'objectif de cette procédure est de supprimer la ou les lignes de commande sélectionnées dans l'objet `ListBox ListeLigComm`.

Cette procédure nommée `SupprimerLigneCommande` se décompose en trois parties :

- ▶ La première consiste à récupérer les lignes sélectionnées dans l'objet `ListeLigComm`.
- ▶ La seconde consiste à récupérer le prix et la quantité à partir de la ligne sélectionnée.
- ▶ La dernière consiste à mettre à jour les stocks et les totaux suite à la suppression de l'élément.

Récupérer les lignes sélectionnées

Pour récupérer les lignes sélectionnées, il faut faire une boucle sur l'ensemble des éléments de la liste des lignes de commande.

La solution est d'utiliser une boucle de type `For-Next` qui permet de parcourir une liste avec un nombre d'itérations défini par une variable.

Exemple

```
For I = ValeurDebut To ValeurFin Step Pas
    'code
Next
```

Application

```
For I = 1 to 10 Step 1
Debug.Write(I) 'Affichera 12345678910 dans la console de Debug
Next
```

Dans cet exemple, il convient de parcourir les éléments de la liste et de les supprimer s'ils ont été sélectionnés.

☞ Parcourez cette liste du dernier au premier élément et supprimez progressivement de la liste des éléments sélectionnés.

Pour faire la boucle en ordre inverse, il suffit d'adapter la boucle `For` :

- ▶ Tout d'abord la boucle doit partir de l'index le plus élevé vers l'index le moins élevé.
- ▶ Ajouter un pas (`Step`) de -1 permet à chaque itération de la boucle de faire diminuer la variable de la boucle de 1.

Pour récupérer le nombre d'éléments dans la liste, il faut utiliser la valeur `ListCount` de `ListeLigComm` :

```
FormExemple.ListeLigComm.ListCount
```

Les éléments de la liste sont identifiés de la manière suivante :

```
FormExemple.ListeLigComm.List(Index)
```

Le premier élément de la liste a pour index 0, puis 1 et ainsi de suite jusqu'au dernier élément qui a pour index le nombre total d'éléments dans la liste -1 (car nous partons de 0 et pas de 1).

Pour identifier si un élément est sélectionné, il faut utiliser la propriété `Selected` de chaque élément, qui renverra `True` si l'élément est sélectionné et renverra `False` si l'élément n'est pas sélectionné.

```
If FormExemple.ListeLigComm.Selected(Index) = True Then
End if
```

Enfin, pour supprimer un élément, il faut utiliser la méthode `RemoveItem` `Index` de l'élément.

Voici comment le code se présentera :

```
For I = FormExemple.ListeLigComm.ListCount - 1 To 0 Step -1
    If FormExemple.ListeLigComm.Selected(I) = True Then
        'Instruction pour récupérer le prix et la quantité et
mettre à jour le stock et les totaux.
        'Suppression de la ligne de commande à la liste
        FormExemple.ListeLigComm.RemoveItem I
    End If
Next
```

- ▶ Extraire le produit et la quantité à partir du texte sélectionné

Le texte sélectionné se décompose de la manière suivante : $\text{Produit} * \text{Quantité} = \text{Prix total}$

Il est nécessaire d'extraire des parties du texte pour récupérer le produit et la quantité présents dans le texte :

- ▶ Le produit se situe entre le premier caractère et le caractère *
- ▶ La quantité se situe après le caractère * jusqu'au caractère =

L'instruction `Instr` va permettre de rechercher une chaîne de caractères au sein d'une autre chaîne de caractères et de remonter la position de la première occurrence rencontrée.



*Cet exemple ne peut pas fonctionner s'il y a un caractère * ou = dans le libellé du produit.*

L'instruction `Instr` se définit ainsi :

```
Position = Instr(TexteDeBase, TexteRecherche)
```

Avec par exemple :

```
Position = Instr("N", "ENI") 'Position sera égal à 2.
```

Une fois la position des caractères * et = trouvée, l'instruction `Mid` est utilisée pour récupérer une partie de la chaîne de caractères.

L'instruction `Mid` se définit ainsi :

```
VariableTexte = Mid(TexteDeBase, Debut, Fin)
```

Avec par exemple :

```
VariableTexte = Mid("ENI-Edition", 3, 4) 'VariableTexte sera égal à « I-Ed »
```

Par conséquent la quantité est égale à :

```
Quantite = Mid(Element, 1, PositionCaractereAsterisque - 1)
```

Et le produit :

```
Produit = Mid(Element, PositionCaractereAsterisque + 1, PositionCaractereEgal - PositionCaractereAsterisque - 1)
```

Mettre à jour les stocks et les totaux.

Pour mettre à jour les stocks, l'appel à la fonction `MAJStock` sera nécessaire avec pour argument le produit et la quantité trouvée précédemment. La fonction `MAJStock` que vous avez conçue précédemment permet de diminuer le stock, alors que notre objectif est d'ajouter du stock. Par conséquent, la variable quantité sera multipliée par -1.

```
MAJStock(Produit, Quantite * -1)
```

Si cette fonction est OK, la liste de ligne de commande sera correctement mise à jour.

Comme pour l'ajout d'une ligne, il faudra récupérer le prix de cette ligne de commande (via la fonction créée `RecupererPrix`) puis le soustraire du Total hors taxe. Pour finir, l'appel à la procédure `AfficherTotal` va mettre à jour les différents totaux et les afficher dans le formulaire.

L'ensemble de la procédure se matérialise donc ainsi :

```

Sub SupprimerLigneCommande()
Dim TotalLigneCommande As Double
Dim Produit As String
Dim Quantite As Integer
For I = FormExemple.ListeLigComm.ListCount - 1 To 0 Step -1
    If FormExemple.ListeLigComm.Selected(I) = True Then
        'Faire une analyse de la ligne sélectionnée pour
        récupérer la quantité et le produit.
        'Pour déterminer la quantité, il suffit de reprendre
        les caractères entre le début et l'astérisque
        PositionCaractereAsterisque =
        InStr(FormExemple.ListeLigComm.List(I), "*")
        PositionCaractereEgal =
        InStr(FormExemple.ListeLigComm.List(I), "=")
        Quantite = Cint(Mid(FormExemple.ListeLigComm.List(I),
        1, PositionCaractereAsterisque - 1))
        Produit = Mid(FormExemple.ListeLigComm.List(I),
        PositionCaractereAsterisque + 1, PositionCaractereEgal -
        PositionCaractereAsterisque - 1)
        If MAJStock(Produit, Quantite * -1) = False Then
            VbMessage = MsgBox("Mise à jour KO", vbOKOnly,
            "Erreur")
        Else
            'Calcul du montant de la ligne de commande /
            utilisation de la fonction récupérerPrix pour avoir le prix par
            produit
            TotalLigneCommande = Quantite * RecupererPrix(Produit)
            'Mise à jour du Total HT puis calcul des totaux
            TotalHT = TotalHT + TotalLigneCommande
            Call AfficherTotal
            'Suppression de la ligne de commande à la liste
            FormExemple.ListeLigComm.RemoveItem I
        End If
    End If
Next
End Sub

```

Étape 8 : Sauvegarde de la facture

Une procédure nommée `Sauvegarder` va être créée pour permettre de valider la facture saisie. Elle sera exécutée lors du clic sur le bouton `Sauvegarder et imprimer la commande`.

La sauvegarde du document va consister à :

- ▶ Donner un numéro de facture ;
 - ▶ Inscrire la date et l'heure ;
 - ▶ Écrire le montant TTC de la facture ;
 - ▶ Proposer l'impression ;
 - ▶ Réinitialiser la facture.
- ❖ Créez la procédure `Sauvegarder` dans le `Module1` :

```
Sub Sauvegarder()  
End Sub
```

Dans un premier temps, il est nécessaire de faire une boucle pour trouver la première ligne non vide dans la feuille `Factures`.

```
'Sélection de la feuille  
Sheets("Ventes").Select  
'Définir une variable Ligne qui nous permettra de parcourir la  
feuille en commençant par la ligne 2.  
Dim Ligne As Integer  
Ligne = 2  
'Parcourir la feuille Ventes pour rechercher la première ligne  
disponible  
While Cells(Ligne, 1).Value <> "" ' tant que la valeur cellule  
est différente de nullé  
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la  
valeur de 1 pour la variable Ligne à chaque passage  
Wend
```

Une fois la ligne identifiée :

- ▶ La colonne `A` va reprendre le numéro de ligne moins 1 (la première facture étant sur la ligne 2) ;
- ▶ La colonne `B` va prendre la date et l'heure du moment avec l'instruction `Now` ;
- ▶ La colonne `C` va prendre la valeur du Total TTC avec la valeur de la variable `TotalTTC`.

```
Cells(Ligne, 1).Value = Ligne - 1  
Cells(Ligne, 2).Value = Now 'instruction pour récupérer la date  
et l'heure.  
Cells(Ligne, 3).Value = TotalTTC
```

Pour imprimer la ligne de facture sur un document, la méthode à utiliser est `PrintOut` de l'objet `Range`. Cette méthode va imprimer la plage en cours, et peut porter sur l'objet `Worksheet`.

Le code consiste à proposer à l'utilisateur l'impression de la facture en utilisant le résultat de la `MsgBox` :

```
'Rep contiendra la réponse de la MsgBox
Dim Rep As String
'Msgbox avec bouton OK et Cancel
Rep = MsgBox("Voulez-vous imprimer la facture ?", vbOKCancel,
"Impression")
```

Il ne restera plus qu'à tester la valeur de `Rep` pour savoir s'il faut lancer l'impression.

```
'tester si la réponse de la MsgBox est le bouton OK
If Rep = vbOK Then
'lancer l'impression si la condition est remplie.
Range(Cells(Ligne, 1), Cells(Ligne, 3)).PrintOut
End If
```

☞ Terminez la procédure en appelant la procédure `Init` qui permet de réinitialiser le formulaire.

```
'Réinitialiser le formulaire
Call Init
```

La procédure sera donc rédigée ainsi :

```
Sub Sauvegarder()
'sélectionner la feuille
Sheets("Factures").Select
'définir une variable Ligne qui permettra de parcourir la
feuille en commençant par la ligne 2.
Dim Ligne As Integer
Ligne = 2
'parcourir la feuille Ventes pour rechercher la première ligne
disponible
While Cells(Ligne, 1).Value <> "" ' tant que la cellule est
différente de nul
    Ligne = Ligne + 1 'itérateur : permet d'augmenter la
valeur de 1 pour la variable Ligne à chaque passage
Wend
Cells(Ligne, 1).Value = Ligne - 1
Cells(Ligne, 2).Value = Now 'instruction pour récupérer la date
et l'heure.
Cells(Ligne, 3).Value = TotalTTC
'Rep contiendra la réponse de la MsgBox
Dim Rep As String
'Msgbox avec bouton OK et Cancel
```

```

Rep = MsgBox("Voulez-vous imprimer la facture ?", vbOKCancel,
"Impression")
'tester si la réponse de la MsgBox est le bouton OK
If Rep = vbOK Then
'lancer l'impression si la condition est remplie.
Range(Cells(Ligne, 1), Cells(Ligne, 3)).PrintOut
End If
'Reinitialiser le formulaire
Call Init
End Sub

```

Étape 9 : Relier les événements aux procédures

Cette étape consiste à relier les événements présents sur les objets (voir Définitions des procédures et événements) avec les procédures créées précédemment.

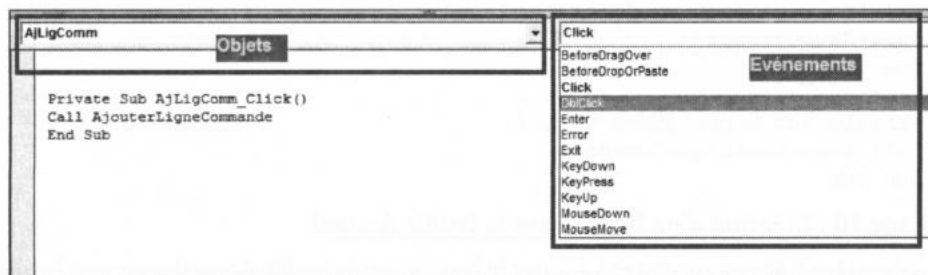
Pour accéder à l'événement d'un objet :

☛ Faites un clic droit sur l'objet puis sélectionnez **Code**.

Vous serez positionné par défaut sur l'événement principal de l'objet (par exemple l'événement **Click** pour le bouton).

Pour changer l'événement sélectionné :

☛ Choisissez dans la liste déroulante l'événement qui convient comme décrit ci-dessous



Voici la liste des événements et la procédure associée :

Événement	Procédure
Clic sur le bouton « Ajouter la ligne de commande » <i>Événement : Click</i>	Appel à la procédure <code>AjLigComm</code>
Clic sur le bouton « Supprimer la ou les lignes sélectionnées » <i>Événement : Click</i>	Appel à la procédure <code>SupprimerLigneCommande</code>

Événement	Procédure
Clic sur le bouton « Sauvegarder et imprimer le document » <i>Événement : Click</i>	Appel à la procédure <code>Sauvegarder</code>
Cocher/Décocher la case à cocher sur la remise. <i>Événement : Change</i>	Appel à la procédure <code>AfficherTotal</code>
Clic sur le bouton « Accéder à l'outil de gestion des ventes » sur la feuille « Accueil »	Voir étape 10

Le résultat est le suivant :

```
Private Sub AjLigComm_Click()
Call AjouterLigneCommande
End Sub

Private Sub Remise_Change()
Call AfficherTotal
End Sub

Private Sub SaveComm_Click()
Call Sauvegarder
End Sub

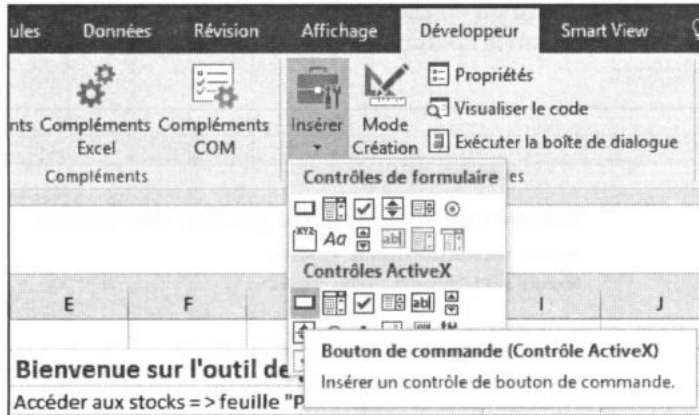
Private Sub SuppLigComm_Click()
Call SupprimerLigneCommande
End Sub
```

Étape 10 : Création d'un bouton sur la feuille Accueil

La dernière étape va consister à ajouter le bouton sur la feuille **Accueil** pour que l'utilisateur puisse cliquer dessus dès qu'il ouvre le fichier.

La manipulation est la suivante :

- ☞ Dans Excel allez sur l'onglet **Développeur** et dans le groupe **Contrôles** choisissez **Insérer**.
- ☞ Cliquez sur le **Bouton de commande** dans les **Contrôles ActiveX**.
- ☞ Dessinez l'objet sur la feuille.



- ❏ Faites un clic droit sur le bouton puis choisissez **Propriétés**.
- ❏ Changez la propriété **Caption** du `CommandButton` ; saisissez **Accéder à l'outil de gestion des ventes**.
- ❏ Faites un clic droit sur le bouton puis choisissez l'option **Visualiser le code**.



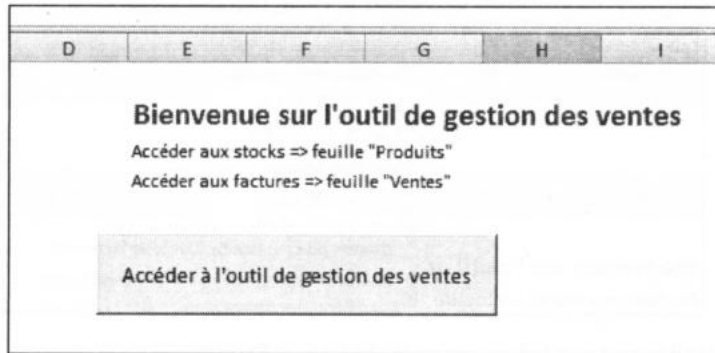
*Si le clic droit n'est pas possible, c'est que le **Mode Création** n'est pas activé. Cliquez sur **Mode Création** dans l'onglet **Développeur** pour passer en **Mode Création**.*

Lorsque vous cliquez sur **Visualiser le code**, vous êtes par défaut conduit vers le code sur l'événement du clic sur le bouton. Dans cette procédure :

- ❏ Appelez la procédure `Init` qui initialise le formulaire.
- ❏ Affichez ce formulaire avec la méthode `Show` sur le formulaire.

```
Private Sub CommandButton1_Click()
    Call Init
    FormExemple.Show
End Sub
```

- ☞ Désactivez le Mode Création en cliquant sur le bouton du même nom dans l'onglet **Développeur**. Cliquez sur le bouton **Accéder à l'outil de gestion des ventes** et testez votre application.



B. Protéger le classeur

1. Description de l'exemple

a. Présentation de l'exemple

La sécurité des données est un problème omniprésent aujourd'hui et plus particulièrement celle des fichiers. Les fichiers sont désormais très accessibles puisqu'ils sont généralement stockés sur le web et facilement partagés. De nombreux utilisateurs sont confrontés à cette problématique et souhaitent avoir davantage de protection sur leurs fichiers, notamment pour éviter de dévoiler des informations confidentielles.

Aujourd'hui, il existe de nombreuses solutions pour sécuriser les fichiers élaborés spécifiquement pour crypter les données.

En effet, il existe plusieurs fonctionnalités permettant d'éviter d'exposer le code à tous les destinataires, de limiter le partage des feuilles du classeur, d'empêcher les modifications sur une feuille ou uniquement sur certaines cellules.

Il faut reconnaître qu'Excel n'est pas l'outil qui offre le plus de garantie en termes de sécurité, mais avec les quelques options suivantes, il est au moins possible de décourager un bon nombre de personnes à s'attaquer aux données que vous aurez protégées.

L'objectif de cet exemple est donc de sécuriser le code, la feuille et les données.

b. Présentation du fichier

Le fichier **Enoncé_3-B.xlsm** est la suite de l'exemple 3-A, les onglets sont les mêmes. Il n'y a donc pas de nouveaux éléments à apporter en présentation.

c. Fonctionnalités

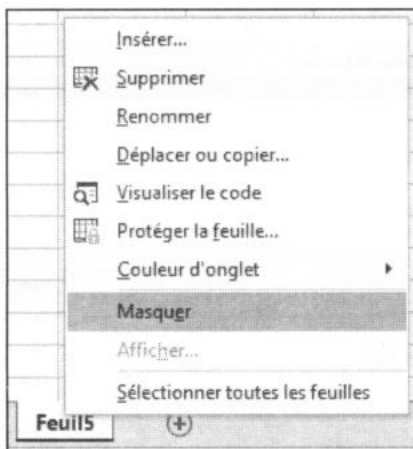
Les fonctionnalités qui vont être développées lors de cet exemple sont les suivantes :

- ▶ Masquer les feuilles Factures et Produits ;
- ▶ Protéger la structure du classeur ;
- ▶ Afficher les stocks via un formulaire ;
- ▶ Protéger les cellules de la feuille Accueil ;
- ▶ Protéger le code VBA.

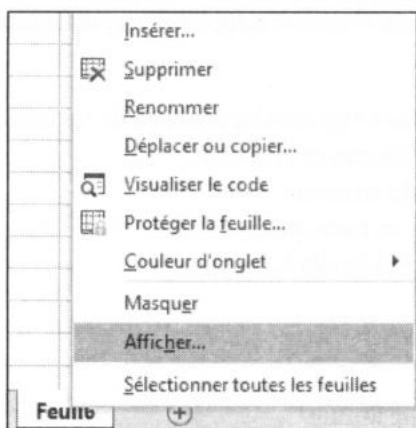
2. Notions de cours

a. Afficher/masquer une feuille

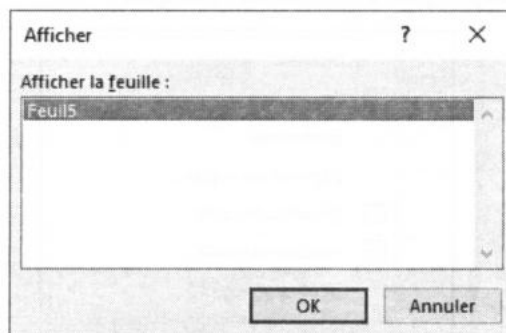
Pour masquer une feuille, l'opération consiste à faire un clic droit sur l'onglet correspondant à la feuille à masquer puis à sélectionner **Masquer** :



Pour afficher une feuille, faites un clic droit sur l'onglet d'une feuille non masquée, puis cliquez sur **Afficher** :



La fenêtre **Afficher** apparaît et permet de choisir les feuilles à afficher.

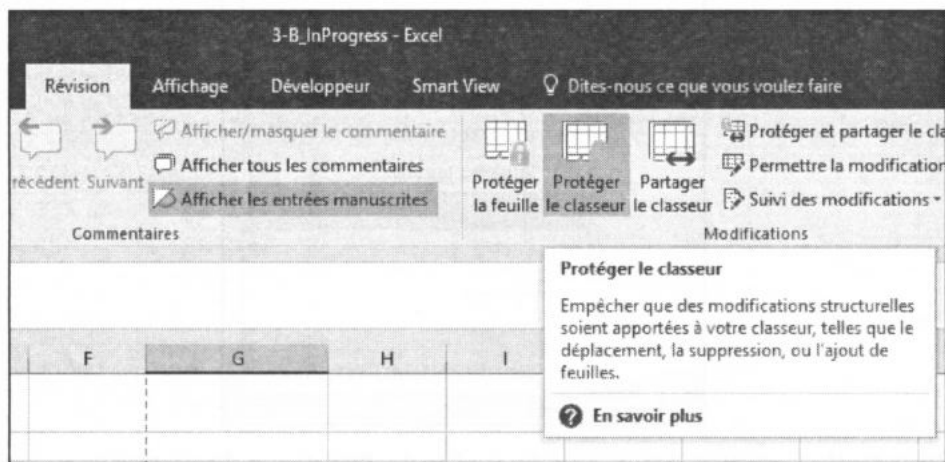


Terminez en cliquant sur OK.

b. Protéger la structure

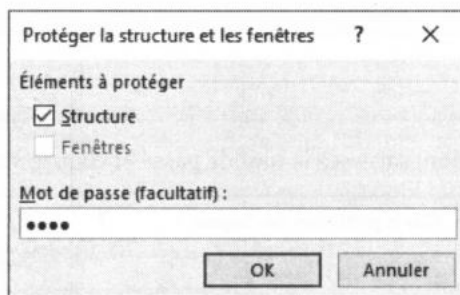
Protéger la structure d'un classeur permet d'empêcher l'ajout, la modification et la suppression de feuilles au sein du classeur.

La fonctionnalité se situe dans l'onglet **Révision**, avec le bouton **Protéger le classeur** :

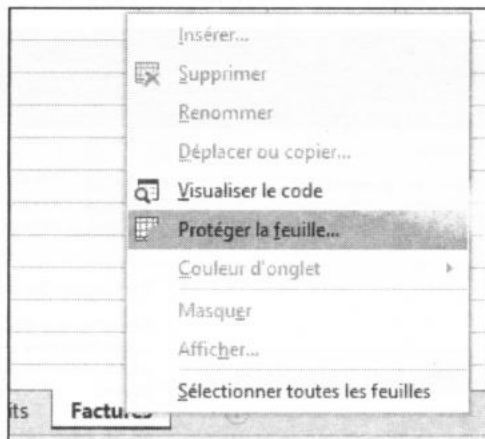


La fenêtre de protection du classeur s'affiche. Elle permet de sélectionner la protection de la structure et d'y associer un mot de passe.

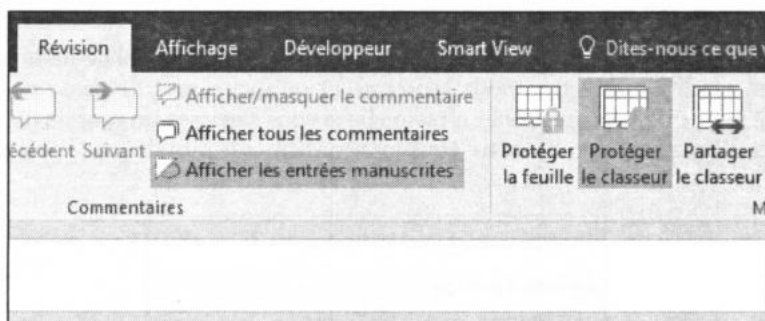
Le mot de passe est facultatif, c'est-à-dire que vous n'êtes pas obligé de le saisir pour mettre en place la protection de la structure et qu'aucun mot de passe ne vous sera demandé pour ôter la protection. En revanche si vous saisissez un mot de passe dans la fenêtre ci-dessus, une autre fenêtre vous fera confirmer le mot de passe précédemment saisi.



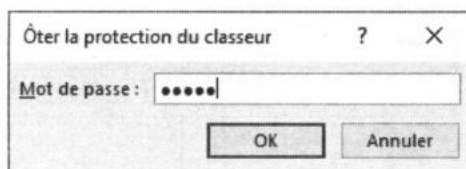
Après avoir cliqué sur OK, vous ne pouvez plus effectuer de modification sur la structure. Comme vous pouvez le constater en faisant un clic droit sur l'onglet d'une des feuilles, il est impossible d'ajouter, de modifier, de supprimer une feuille. Il n'est pas non plus possible d'afficher ou de masquer une feuille.



Pour ôter la protection, il faut se rendre sur l'onglet Révision et cliquer à nouveau sur le bouton Protéger le classeur qui est mis en surbrillance.



Pour terminer l'opération, saisissez le mot de passe et cliquez sur OK.

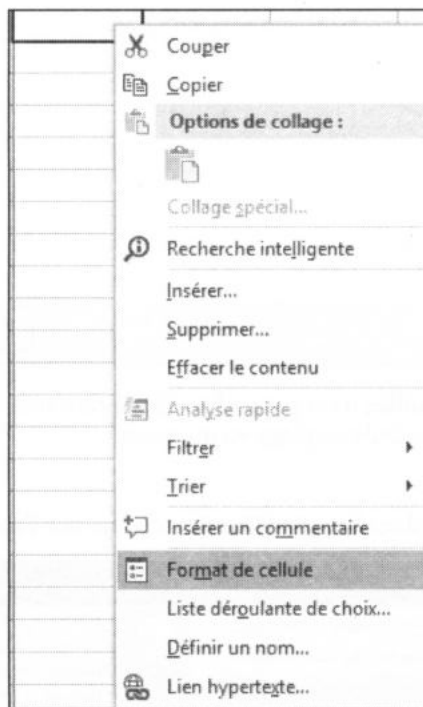


c. Protéger la feuille et ses cellules

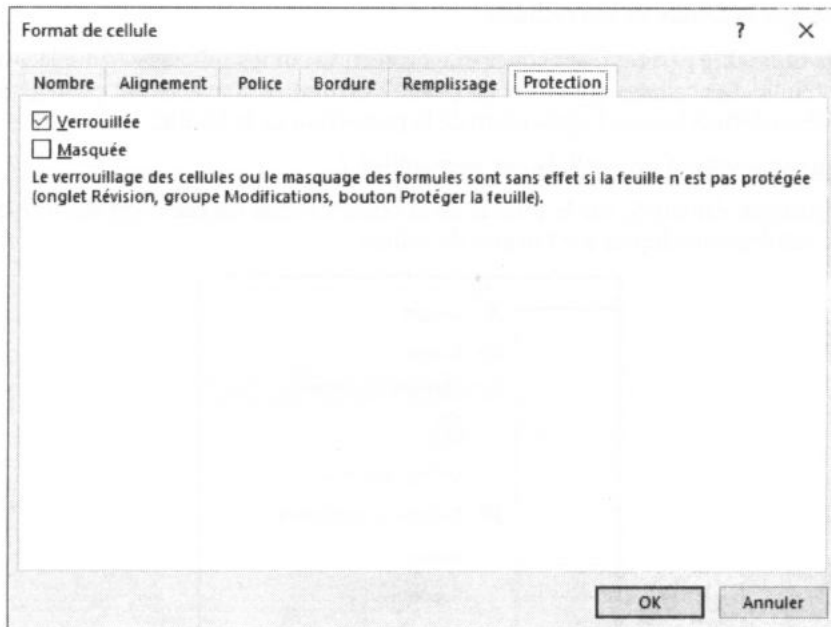
La fonctionnalité permet d'appliquer une protection sur les cellules verrouillées au sein d'une feuille. Les cellules qui ont la propriété **Verrouillée** active seront protégées selon les critères définis lors de l'application de la protection de la feuille.

Comment savoir si une cellule est verrouillée ?

L'information est située sur le format de la cellule. Faites un clic droit sur une ou plusieurs cellules puis cliquez sur **Format de cellule**.



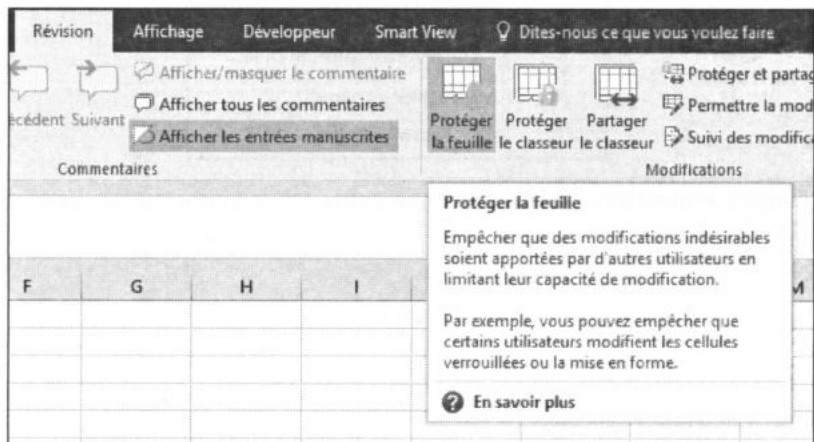
Dans l'onglet **Protection**, il est possible de constater si l'information **Verrouillée** est active. Ici, c'est bien le cas.



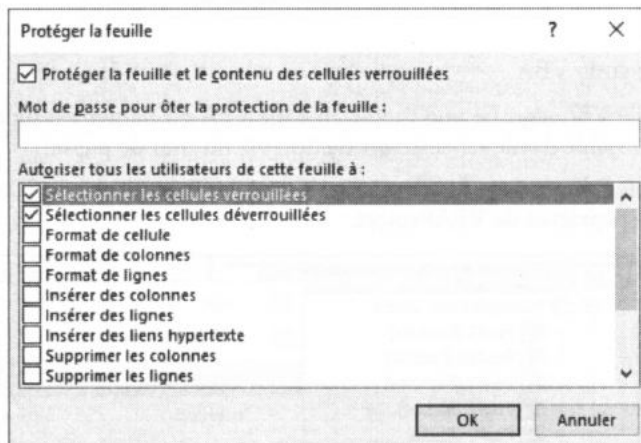
Si la case à cocher **Verrouillée** n'est pas cochée, l'application de la protection sur la feuille n'aura pas d'effet sur la cellule ou plage en question.

Protéger la feuille

Pour protéger la feuille, dans l'onglet **Révision**, cliquez sur **Protéger la feuille**.



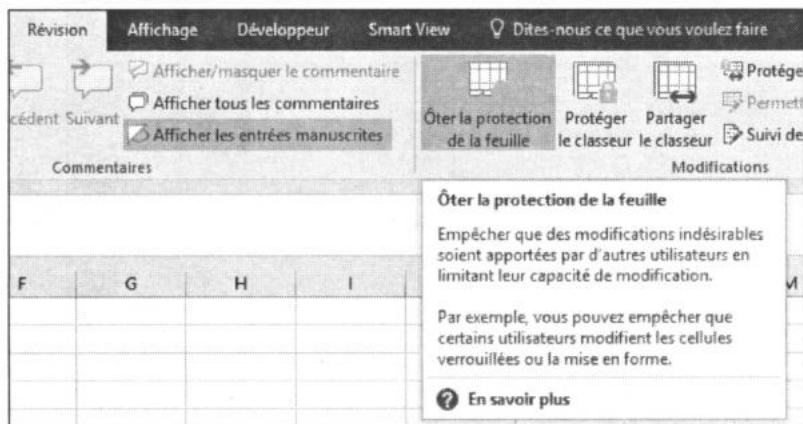
La fenêtre de paramétrage de la protection **Protéger la feuille** s'affiche. Les fonctionnalités cochées sont celles qui resteront accessibles après application de la protection de la feuille. Le mot de passe est facultatif, c'est-à-dire que vous n'êtes pas obligé de le saisir pour mettre en place la protection de la feuille et qu'aucun mot de passe ne vous sera demandé pour ôter la protection. En revanche, si vous saisissez un mot de passe dans la fenêtre ci-dessous, une autre fenêtre vous fera confirmer le mot de passe précédemment saisi.



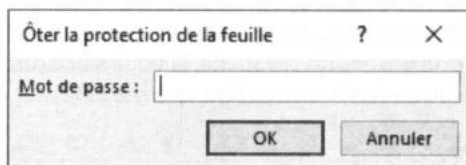
Après avoir cliqué sur OK la protection de la feuille est active.

Ôter la protection de la feuille

Pour ôter la protection de la feuille, allez dans l'onglet **Révision**, puis cliquez sur **Ôter la protection de la feuille**. Le bouton est situé là où se situait le bouton **Protéger la feuille** avant que celle-ci ne soit active.



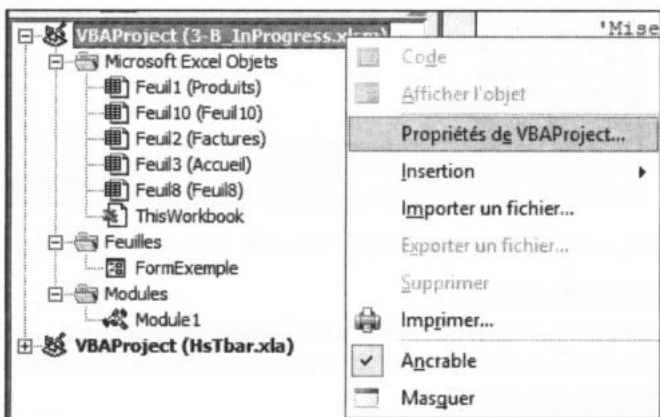
Dans le cas où un mot de passe avait été saisi, une fenêtre s'affiche pour entrer le mot de passe :



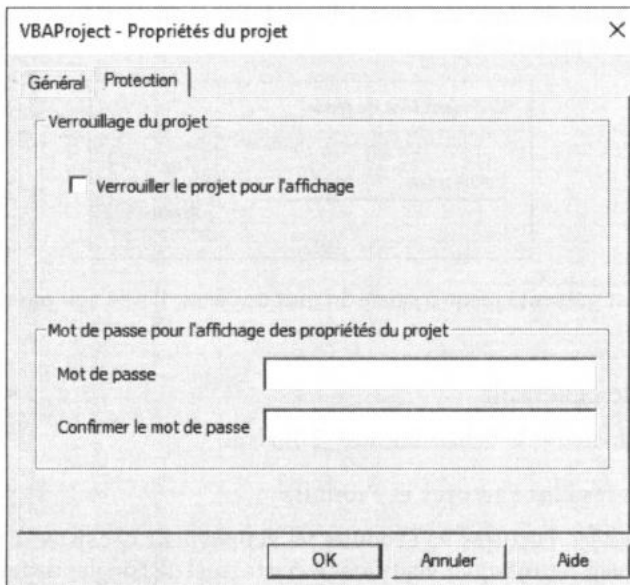
d. Protéger le code VBA

Protéger le code VBA signifie que l'accès au code VBA du fichier est bloqué aux utilisateurs. Il sera possible d'enlever cette protection via un mot de passe.

Sur Visual Basic Editor, dans l'explorateur de projet, faites un clic droit sur le document et cliquez sur Propriétés de VBAProject.



La fenêtre des propriétés s'affiche. Positionnez-vous sur l'onglet **Protection**.



VBAProject - Propriétés du projet

Général Protection

Verrouillage du projet

Verrouiller le projet pour l'affichage

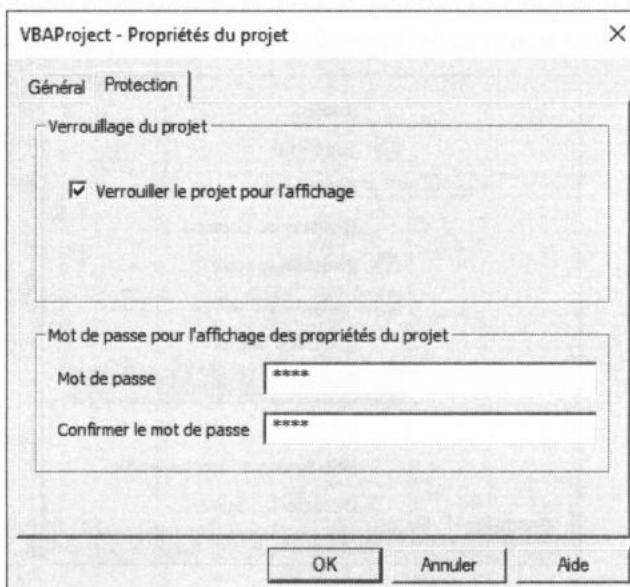
Mot de passe pour l'affichage des propriétés du projet

Mot de passe

Confirmer le mot de passe

OK Annuler Aide

Cochez **Verrouiller le projet pour l'affichage** et saisissez un mot de passe.



VBAProject - Propriétés du projet

Général Protection

Verrouillage du projet

Verrouiller le projet pour l'affichage

Mot de passe pour l'affichage des propriétés du projet

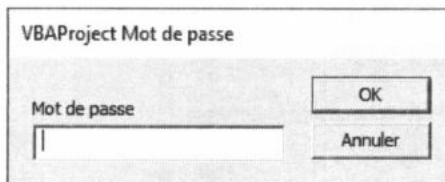
Mot de passe

Confirmer le mot de passe

OK Annuler Aide

Après avoir cliqué sur OK, votre projet VBA sera protégé.

Dès que vous vous rendrez sur Visual Basic Editor, une fenêtre vous permettra de déverrouiller cette protection.



La protection est présente jusqu'à saisie du mot de passe, il ne s'agit pas d'une protection permanente.

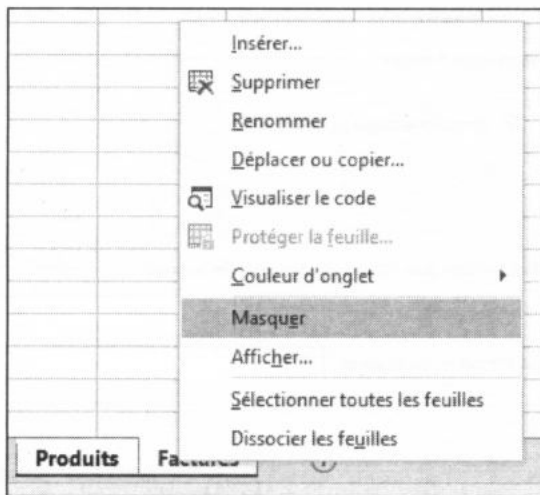
3. Création de l'exemple

❖ Tout d'abord, ouvrez le fichier Enoncé_3-B.xlsm.

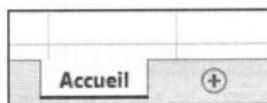
a. Masquer les feuilles Factures et Produits

Masquer les feuilles **Factures** et **Produits** va simplement consister à sélectionner les deux feuilles puis à les masquer via le menu contextuel de l'onglet de feuille.

- ❖ Cliquez sur la feuille **Factures**.
- ❖ Appuyez sur la touche **Ctrl** et cliquez sur l'onglet de la feuille **Produits**.
- ❖ Faites un clic droit au niveau de l'onglet de feuille **Produits**, puis cliquez sur **Masquer**.



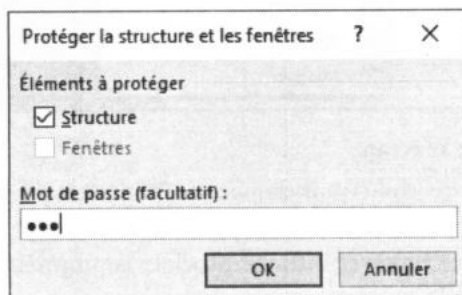
Seule la feuille **Accueil** est accessible comme vous pouvez le constater :



b. Protéger la structure du classeur

Protéger la structure du classeur permettra d'empêcher les utilisateurs du fichier d'afficher les feuilles masquées, d'ajouter de nouvelles feuilles et de modifier le nom de la feuille **Accueil**.

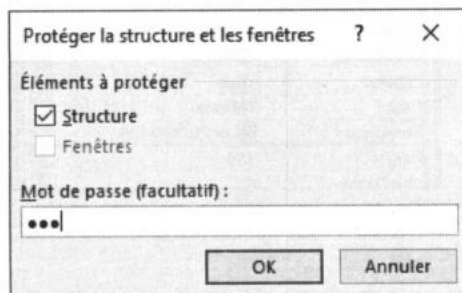
- ☞ Dans l'onglet **Révision**, cliquez sur le bouton **Protéger le classeur**.
- ☞ Dans la fenêtre de protection du classeur, saisissez le mot de passe **eni** et cliquez sur **OK**.



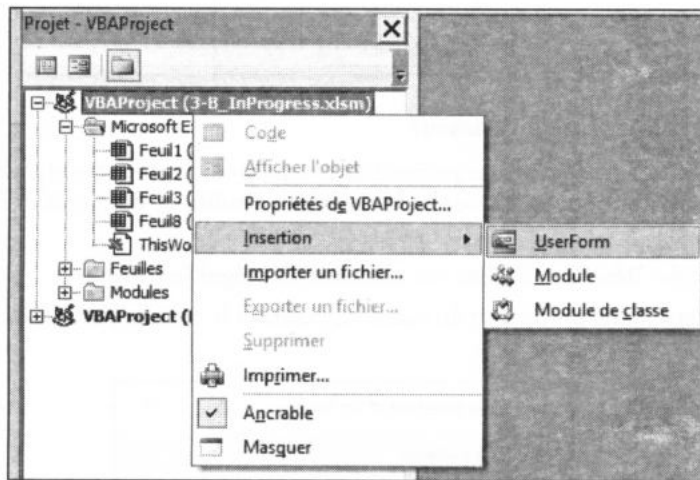
- ☞ Confirmez le mot de passe dans l'écran de confirmation en saisissant à nouveau **eni**. Votre classeur est désormais protégé, et sans mot de passe, il sera impossible pour un utilisateur de modifier sa structure.

c. Afficher les stocks via un formulaire

Cette fonctionnalité permet d'afficher un pop-up avec pour chaque produit le stock en cours. Au clic sur un bouton sur la feuille **Accueil** vous aurez la fenêtre suivante :



- ☞ Dans Visual Basic Editor, ajoutez une fenêtre en effectuant un clic droit sur le projet puis en cliquant sur **Insertion** puis sur **UserForm**.

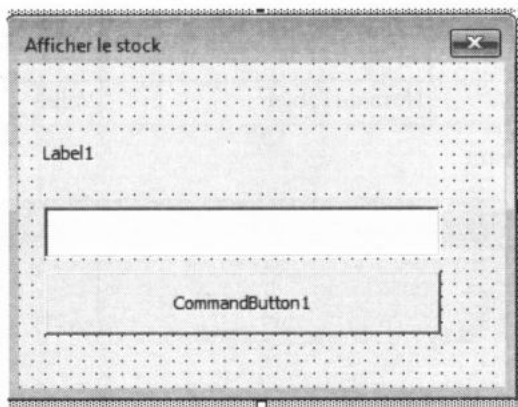


Le formulaire s'affiche à l'écran.

- ☞ Faites un clic droit sur celui-ci puis cliquez sur **Propriétés** pour modifier les propriétés du formulaire.
- ☞ Modifiez la propriété **Name** en **AfficherStock** et la propriété **Caption** en **Afficher le stock**.



- Sur le formulaire, ajoutez trois contrôles. De haut en bas : un Intitulé (Label), une Zone de texte saisissable (TextBox) et un Bouton de commande (CommandButton) pour que la fenêtre prenne l'apparence suivante :



- Modifiez les propriétés des différents objets.

Pour l'intitulé :

Propriété	Définition de la propriété	Valeur
Caption	Apparence visuelle de l'intitulé	Saisissez le mot de passe pour afficher les stocks

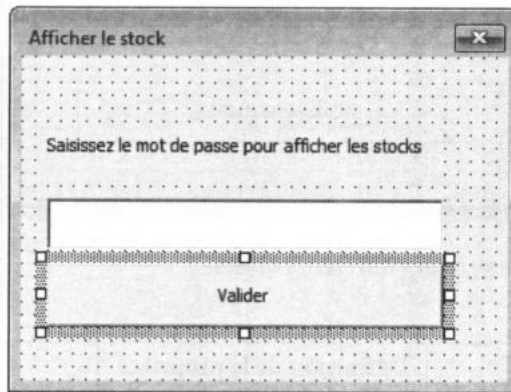
Pour la zone de texte :

Propriété	Définition de la propriété	Valeur
Text	Apparence visuelle de la zone de texte	Vide
PasswordChar	Cette propriété permet d'avoir un autre caractère que celui saisi par l'utilisateur dans la zone de texte. Ici chaque caractère saisi sera remplacé par une étoile	*
Name	Nom de l'objet	MDP

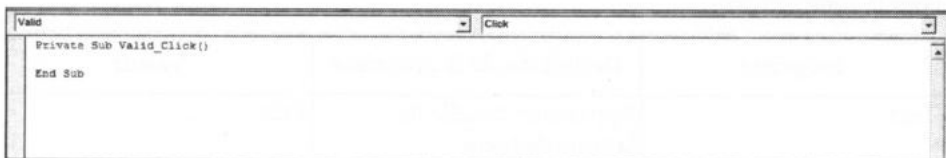
Pour le bouton de commande :

Propriété	Définition de la propriété	Valeur
Caption	Apparence visuelle du bouton de commande	Valider
Name	Nom de l'objet	Valid

Le résultat est le suivant :



- Double cliquez sur le bouton **Valid** pour accéder à l'événement **Click** du bouton **Valid**.



La procédure associée à l'événement clic sur le bouton permettra de vérifier si le mot de passe est correct avant d'afficher le pop-up avec le stock.

Pour vérifier le mot de passe il suffit de contrôler que le texte saisi dans le contrôle MDP est égal au mot de passe défini (notre mot de passe sera « enistock »)

- Dans la procédure `Valid_Click`, testez la valeur de la propriété `Text` du contrôle MDP :

```

Private Sub Valid_Click()
If MDP.Text = "enistock" Then
'affichage Popup
End If
End Sub

```

- ❖ Affichez le pop-up en réalisant une boucle sur la feuille **Produits**. La variable `ContenuString` de type `string` (chaîne de caractères) contiendra chaque ligne du stock.
- ❖ Réalisez une boucle pour parcourir chacune des lignes avec l'instruction `while...wend` avec une variable `Ligne` qui s'incrémente à chaque itération de la boucle. Parcourez à partir de la ligne 2 la feuille **Produits** jusqu'à trouver une ligne vide. À chaque ligne non vide, la variable `ContenuString` ajoute une portion de texte correspondant à un saut de ligne, le nom du produit, un caractère séparateur, le stock du produit.
- ❖ Terminez en ajoutant une boîte de dialogue de type `MsgBox` avec comme argument d'intitulé la variable `ContenuString`.

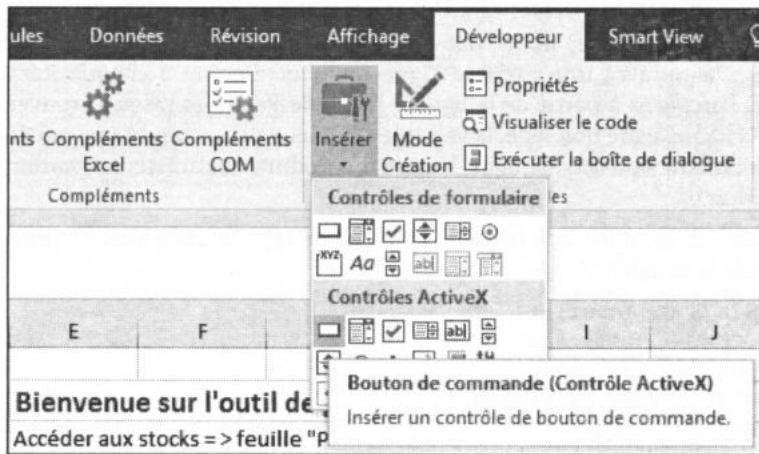
```
Private Sub Valid_Click()
If MDP.Text = "enistock" Then
'Création des variables
Dim ContenuString As String
Dim Ligne As Integer
'La variable ligne permet de parcourir la feuille Produits. La
première ligne de stock est la ligne 2.
Ligne = 2
'création d'une boucle pour tester si la cellule contient une
valeur.
While Sheets("Produits").Cells(Ligne, 1).Value <> ""
'Ajouter à la variable contenu string le nom du produit et le
stock en cours
ContenuString = ContenuString & Chr(13) &
Sheets("Produits").Cells(Ligne, 1).Value & " : " &
Sheets("Produits").Cells(Ligne, 3).Value
'Pour tester la ligne suivante dans la prochaine itération de
la boucle
Ligne = Ligne + 1
Wend
'Affichage du texte de la variable ContenuString dans une
MsgBox
MsgBox ContenuString
End If
End Sub
```

La dernière étape va consister à ajouter le bouton sur la feuille **Accueil** pour que l'utilisateur puisse cliquer dessus dès qu'il ouvre la fenêtre de saisie de mot de passe.

La manipulation est la suivante :

- ❖ Allez dans l'onglet **Développeur**, groupe **Contrôles** cliquez sur le bouton **Insérer**.

- ☞ Cliquez sur le **Bouton de commande** dans les **Contrôles ActiveX**.



- ☞ Dessinez l'objet sur la feuille.
- ☞ Faites un clic droit sur le bouton puis choisissez **Propriétés**.
- ☞ Changez la propriété **Caption**, saisissez **Afficher les stocks**.
- ☞ Faites à nouveau un clic droit sur le bouton et sélectionnez **Visualiser le code**.
- ☞ Modifiez l'événement lié au clic sur le bouton.



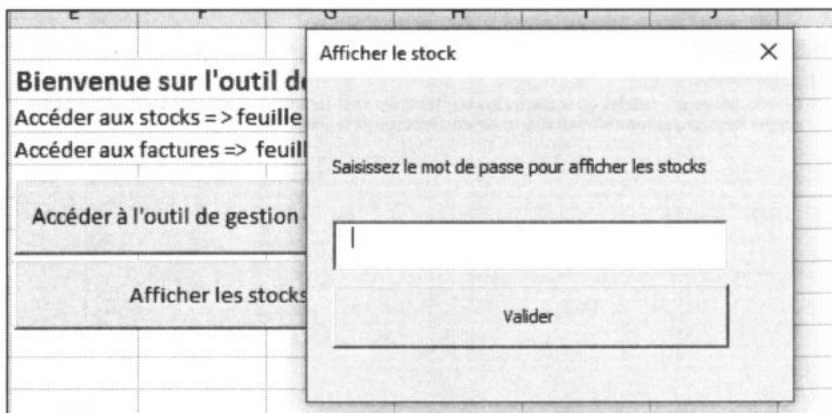
*Si le clic droit n'est pas possible, c'est que le **Mode Création** n'est pas activé. Cliquez sur le bouton **Mode Création** dans l'onglet **Développeur** pour passer en **Mode Création**.*

Lorsque vous cliquez sur **Visualiser le code**, vous êtes par défaut conduit vers le code sur l'événement du clic sur le bouton. Dans cette procédure :

- ☞ Affichez le formulaire **AfficherStock** avec la méthode **Show** sur le formulaire.

```
Private Sub CommandButton2_Click()
    AfficherStock.Show
End Sub
```

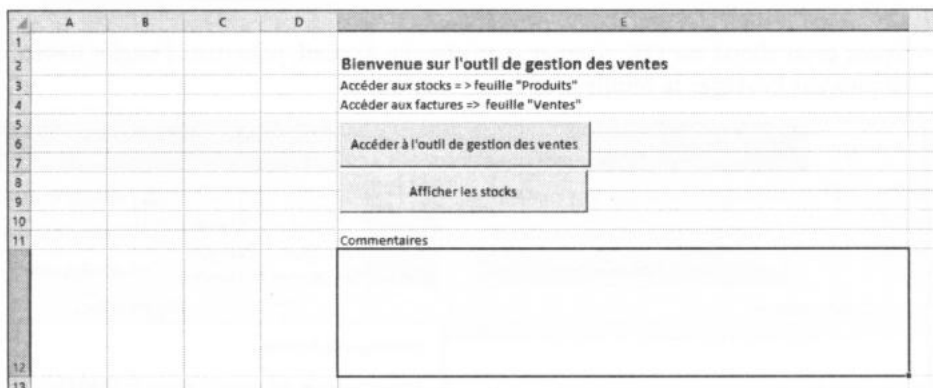
- ☞ Désactivez le **Mode Création**, cliquez sur le bouton **Afficher les stocks** et testez votre application.



d. Protéger les cellules de la feuille Accueil

Afin d'éviter toute modification dans la feuille **Accueil**, nous allons protéger les cellules de la feuille tout en laissant un champ commentaire accessible :

- ☞ Redimensionnez les cellules de la manière suivante :

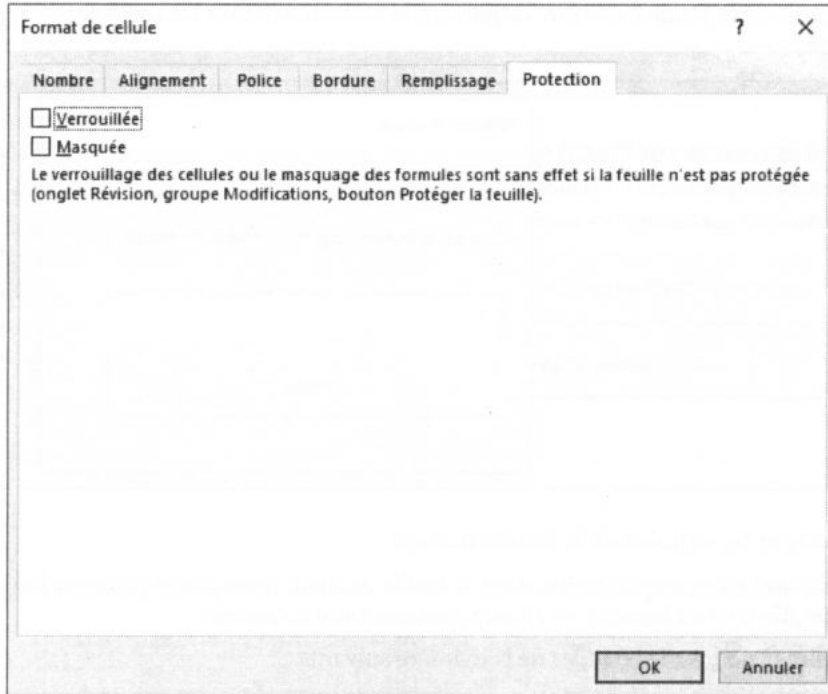


La cellule **E12** sera donc la seule modifiable dans la feuille.

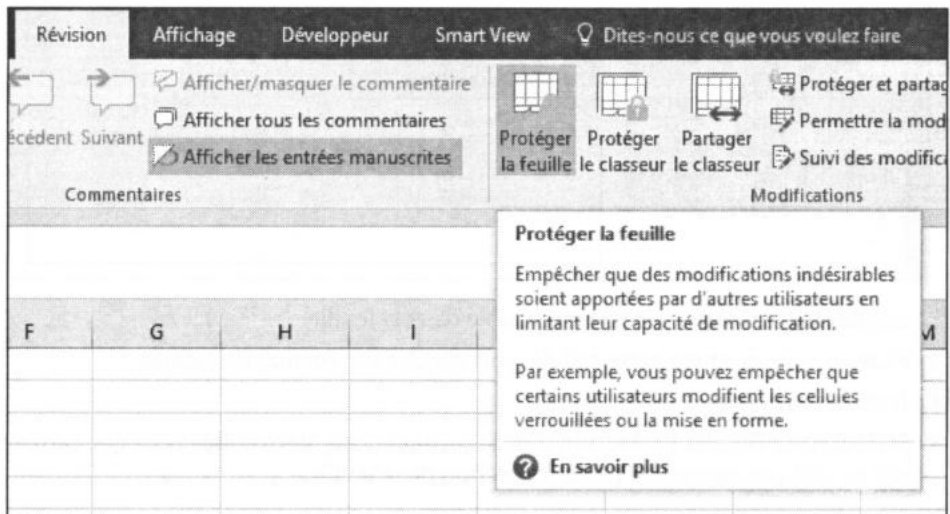
- ☞ Faites un clic droit sur cette cellule, puis cliquez sur **Format de cellule**.

La fenêtre **Format de cellule** apparaît.

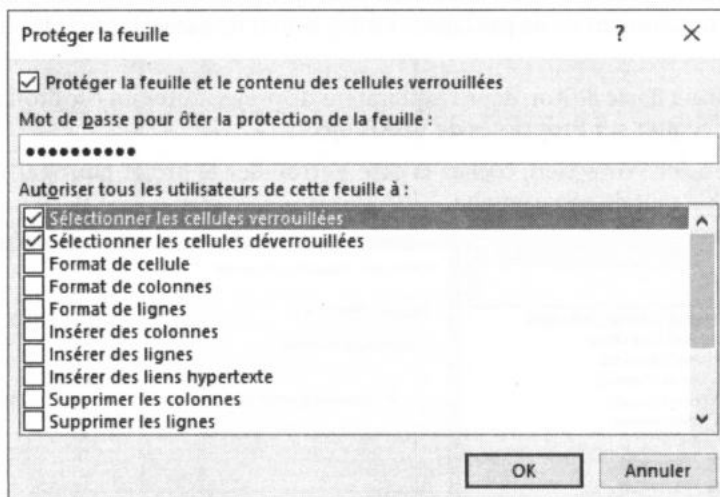
- ☞ Sélectionnez l'onglet **Protection** puis décochez la case **Verrouillée** pour que cette cellule ne soit pas concernée par la protection de la feuille.



- Après avoir cliqué sur OK, revenez sur la feuille Accueil, puis dans l'onglet Révision, cliquez sur Protéger la feuille.

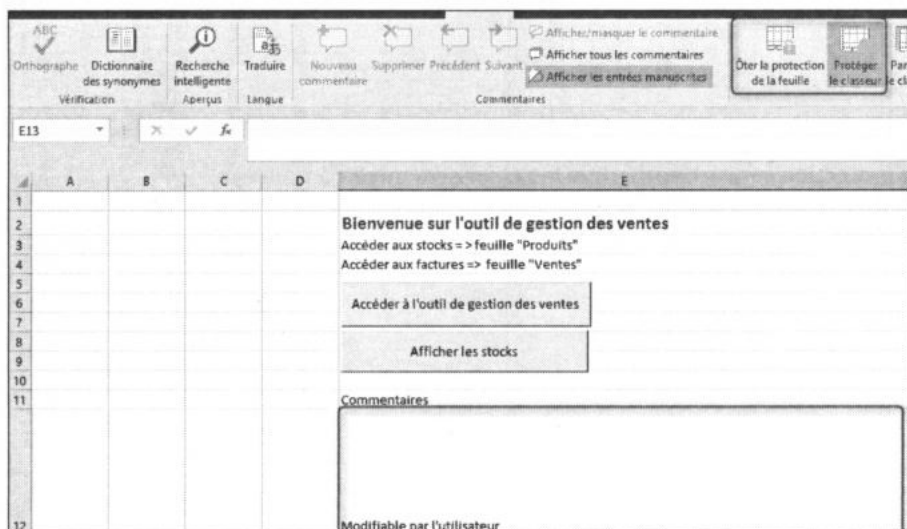


- ☞ Saisissez le mot de passe **enifeuille** sans toucher aux cases à cocher. La seule action possible sera de sélectionner les cellules verrouillées.



- ☞ Confirmez le mot de passe à l'écran suivant.

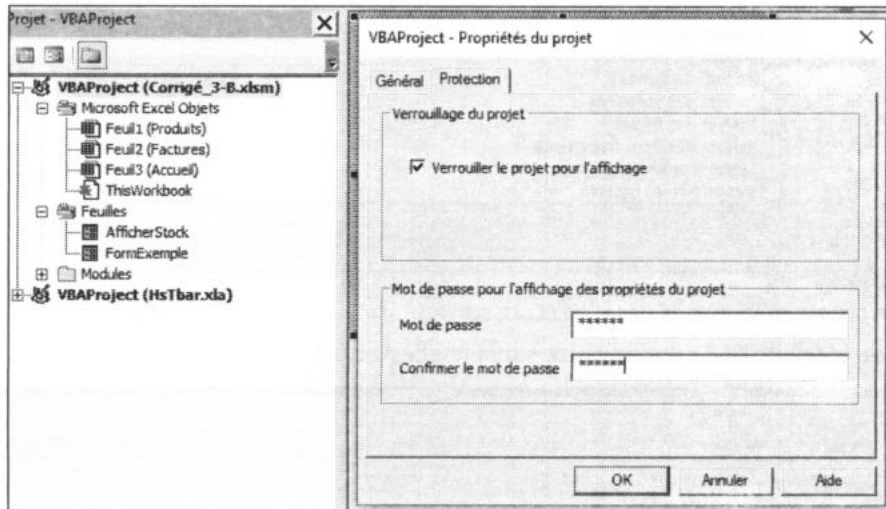
Seule la cellule E12 est modifiable sur la feuille Accueil.



e. Protéger le code VBA

L'objectif est de protéger votre code VBA pour que personne ne puisse y accéder. Cela permettra notamment de ne pas laisser visible le mot de passe permettant d'afficher les stocks.

- ❏ Dans **Visual Basic Editor**, dans l'explorateur de projet, faites un clic droit sur le document et cliquez sur **Propriétés de VBAProject**.
- ❏ Dans l'onglet **Protection**, cochez la case **Verrouiller le projet pour l'affichage**, puis saisissez le mot de passe **enivba**.



La protection sera active dès que le fichier sera ouvert à nouveau.

Chapitre 4

Gestion d'une campagne de test

- A. Création de tableaux et graphiques croisés dynamiques (TCD et GCD) 155
- B. Automatisation de la création d'un rapport PowerPoint 200

A. Création de tableaux et graphiques croisés dynamiques (TCD et GCD)

1. Description de l'exemple

a. Présentation de l'exemple

Dans le cadre d'un projet informatique de la société **SacEni** qui commercialise des sacs, nous travaillons sur la phase de recette de l'application portail client qui permet la vente en ligne de ces sacs. L'application **EniSac_App** se compose de trois parties : la partie Front, la partie Middle et la partie Back.



Dans les applications informatiques, on retrouve généralement les notions de Front-End, Middleware et Back-End : la partie Front-End porte sur l'interface de l'application ; la partie Middleware permet de gérer la relation entre les commandes liées à l'interface et les données ; la partie Back-End permet le stockage sur les bases de données.

Pendant cette phase de test, les testeurs tracent leurs actions dans des fichiers Excel. Il est également demandé par la direction de LCM d'avoir une vue de l'avancement global et de la situation de la recette.

Un peu de vocabulaire :

- ▶ **Test d'une fonctionnalité** : une fonctionnalité est couverte par un ensemble de cas de test. Ils couvrent l'ensemble des exigences de cette fonctionnalité. La couverture de la fonctionnalité doit être exhaustive.
- ▶ **Cas de test** : le cas de test correspond à un ensemble d'étapes composant un scénario à tester. Un test est :
 - ▶ OK si l'ensemble des étapes est déroulé et que toutes les étapes sont conformes à la description ;

- ▶ KO si l'ensemble des étapes est déroulé et qu'au moins une étape n'a pas été conforme à la description ;
- ▶ Bloqué : si à cause d'une anomalie, l'exécution n'a pas pu aller à la fin du scénario de test.
- ▶ **Anomalie** : une anomalie correspond à un cas rencontré qui n'est pas conforme à la description du cas de test.
- ▶ **Cycle de test** : un cycle de test correspond à l'exécution d'un ensemble de cas de test durant une période donnée. Il peut y avoir plusieurs cycles de test et donc plusieurs exécutions de chaque cas de test, notamment pour vérifier l'impact d'une correction éventuelle sur des tests connexes.

b. Présentation du fichier

Le fichier se décompose en cinq onglets :

- ▶ Feuille Test

La feuille **Test** récapitule l'ensemble des 60 tests de cette application avec leur état à l'instant de cet exemple.

Colonne	Libellé	Description
Colonne A	Nom	Il s'agit du nom du test, ici, les tests sont différenciés par un numéro. En théorie le nom du cas de test est suffisamment explicite pour décrire l'objet du test.
Colonne B	Priorité	La priorité est un critère déterminant pour qualifier l'importance d'un test par rapport aux autres. La priorité peut prendre les valeurs P1, P2, P3.
Colonne C	Statut du test	Il s'agit de connaître le statut du test au moment de la réalisation de l'extraction des données. Il peut être : <ul style="list-style-type: none"> ▶ OK ▶ KO ▶ Non commencé ▶ Non livré
Colonne D	Fonctionnalité	Il s'agit du sujet englobant le test.

► Feuille Exécution

La feuille **Exécution** contient toutes les exécutions de cas de test. Un test peut être exécuté plusieurs fois dans le cadre de différents cycles ou lorsque le cas de test est rejoué suite à une correction. Les colonnes sont les suivantes :

Colonne	Libellé	Description
Colonne A	ID Exécution	Identifiant unique de l'exécution de test. Il permet, par exemple, de faire la différence entre deux exécutions d'un même test.
Colonne B	Test	Nom du test exécuté.
Colonne C	Priorité	Priorité du test évoqué dans la colonne B (P1, P2, P3).
Colonne D	Statut de l'exécution	Situation du test au terme de son exécution. Il peut être : <ul style="list-style-type: none"> ► OK ► KO ► Bloqué
Colonne E	Fonctionnalité	Il s'agit du sujet englobant le test évoqué dans la colonne B.
Colonne F	Date d'exécution	Date de l'exécution du test.
Colonne G	Testeur	Personne qui a exécuté le test : ici Cécile ou Jean.

► Feuille Anomalies

La feuille **Anomalies** recense les différentes anomalies détectées durant l'exécution des tests.

Colonne	Libellé	Description
Colonne A	ID	Identifiant unique de l'anomalie.
Colonne B	Libellé	Libellé de l'anomalie ici composé de l'identifiant et du test. Dans de vrais cas, l'anomalie doit avoir un nom suffisamment explicite pour décrire l'objet de l'anomalie.
Colonne C	Date d'ouverture	Date de détection et/ou création de l'anomalie dans le fichier.

Colonne	Libellé	Description
Colonne D	Statut	État d'avancement de l'anomalie. Différentes étapes sont possibles : <ul style="list-style-type: none"> ▶ Créée : vient d'être créée mais non qualifiée ; ▶ En qualification : l'anomalie est avérée, mais n'a pas encore de projet attribué pour la correction ; ▶ En correction : l'anomalie a été affectée à un projet qui se charge de la correction ; ▶ À valider : la correction est terminée et livrée, le testeur doit repasser le cas pour terminer l'anomalie. ▶ Terminée : l'anomalie est soit corrigée, soit abandonnée.
Colonne E	Test associé	Test sur lequel l'anomalie a été détectée.
Colonne F	Date de clôture	Date à laquelle l'anomalie a été clôturée (pour les motifs vus ci-dessus).
Colonne G	Projet	Projet en charge de corriger l'anomalie détectée. Les projets de l'application sont les suivants : <ul style="list-style-type: none"> ▶ Front ; ▶ Middle ; ▶ Back ; ▶ Non défini : si l'anomalie n'a pas encore été qualifiée.
Colonne H	Testeur	Testeur qui a détecté l'anomalie.
Colonne I	Priorité	Priorité du ticket (P1, P2, P3).

▶ **Feuille Rapport**

La feuille **Rapport** est vierge initialement. Elle contiendra par la suite le rapport qui sera généré et exporté vers PowerPoint dans la deuxième partie de cet exemple.

▶ **Feuille TCD_GCD**

La feuille **TCD_GCD** contiendra les tableaux croisés dynamiques et les graphiques croisés dynamiques qui seront mis en place dans cet exemple.

c. Fonctionnalités

L'objectif de cet exemple est d'automatiser la création d'un rapport d'activité des tests sur l'application. Par conséquent, il faut réaliser différents tableaux croisés dynamiques pour récupérer l'information puis la transformer en graphiques croisés dynamiques pour la partie visuelle.

Voici les situations envisagées :

- ▶ Création d'un suivi hebdomadaire du stock de tickets ;
- ▶ Nombre d'anomalies par projet (et par priorité) ;
- ▶ Avancement des cas de test ;
- ▶ Revue des cycles de test ;
- ▶ Indicateur de situation des tests : nombre d'anomalies et pourcentage de tests OK.

2. Notions de cours

a. Créer un tableau croisé dynamique simple

Un **tableau croisé dynamique**, abrégé TCD, permet d'exposer des données en fonction d'axes définis par l'utilisateur. Les données sont agrégées selon les axes.

Un axe correspond à une définition permettant de qualifier les données. Par exemple : le sexe, le poste, le groupe... Les axes sont exprimés soit en colonne soit en ligne et leur contenu peut être filtré pour ne pas forcément afficher toutes les valeurs.

Les données sont agrégées c'est-à-dire qu'elles sont regroupées autour des valeurs de l'axe. Il est possible de les exprimer de plusieurs manières différentes : somme, moyenne, nombre (compter)... Les données sont forcément des valeurs numériques.

Exemple

Voici un exemple de transformation de tableau de données en TCD.

☞ Pour réaliser cet exemple, ouvrez le fichier **ExempleCours_Chapitre4.xlsm**, feuille TCD.

La source de données exprimée sous forme de tableau porte sur les notes d'élèves :

	A	B	C
1	Sexe	Note	Nom
2	Homme	9.5	Albert
3	Homme	12	Anthony
4	Femme	15	Aurélie
5	Femme	9.5	Béatrice
6	Homme	8.5	Benoit

	A	B	C
7	Femme	13	Bérangère

La transformation de ces données en TCD :

	Étiquettes de colonnes		
	Femme	Homme	Total général
Moyenne de Note	12.5	10	11.25

Ici l'axe correspond au sexe, et les données sont les notes dont la moyenne est réalisée en fonction de l'axe.

Manipulation

- ☞ Pour créer un TCD, sélectionnez la plage de données du tableau (en-tête comprise) soit la plage A1:C7 puis dans l'onglet Insérer cliquez sur Tableau croisé dynamique.

- ☞ Choisissez si vous voulez faire apparaître le TCD dans une nouvelle feuille ou une feuille existante.
- ☞ Validez la création du TCD.

Le TCD est créé sur la page. Vous pouvez manipuler les différents champs qui se trouvent dans la partie supérieure **Champs de tableau croisé dynamique** pour les positionner en tant qu'axe et données (en bas du volet) avec un glisser-déposer. Les axes sont appelés **Étiquettes de lignes** ou **Étiquettes de colonnes**. Les données sont appelées **Valeurs**. Deux nouveaux onglets apparaissent également dans le menu Excel **Outils de tableau croisé dynamique** : **Analyse** et **Création**.

☞ Faites glisser le champ **Sexe** dans la zone **Colonnes** et le champ **Note** dans la zone **Valeurs**.

C'est le nombre de notes qui est calculé.

☞ Pour obtenir la moyenne cliquez sur **Nombre de Note** dans la zone **Valeurs**, puis sur **Paramètres des champs de valeurs** et sélectionnez **Moyenne**.

Étiquettes de colonnes		Total général	
Femme	Homme		
Moyenne de Note	14	12	14,11111111



Une donnée peut-elle être un axe ? Si vous n'avez pas encore bien compris le fonctionnement des TCD, revenez sur cette remarque plus tard, elle vous déstabilisera plus qu'elle n'apportera de nouvelles perspectives. Pour les plus aguerris aux TCD, vous pourrez constater qu'une colonne comme les notes considérées comme des données peut aussi être un axe. Et le prénom peut servir de données. En effet, si la note est considérée comme un axe, alors il est possible de compter le nombre de personnes ayant obtenu chaque note.

Étiquettes de colonnes		Total général	
Femme	Homme		
13	1	1	1
15	1	1	1
18,5	1	1	1
9,5	1	1	2
Total général	3	3	6



Dans ce cas présent, les notes sont en colonnes et le nombre de personnes ayant la note se situe à côté car nous comptons le nombre de personnes ayant eu cette note.

b. Créer un tableau croisé dynamique avec l'assistant

L'Assistant Tableau croisé dynamique va permettre de consolider plusieurs plages de calcul au sein d'un même TCD.

Exemple

Voici un tableau contenant les ventes de rollers pour un magasin spécialisé sur les quatre trimestres (T1 à T4). Les ventes sont distinguées par peinture (en colonne) et par modèle de roller (en ligne).

Nous allons réaliser une petite manipulation pour comprendre les possibilités de l'Assistant TCD.

☞ Pour réaliser cet exemple, ouvrez le fichier ExempleCours_Chapitre4.xlsm, feuille Assistant_TCD.

Nous allons consolider les quatre plages (Plage T1 – A1:E4 ; Plage T2 – A5:E8 ; Plage T3 – A9:E12 ; Plage T4 – A13:E16) en une seule.

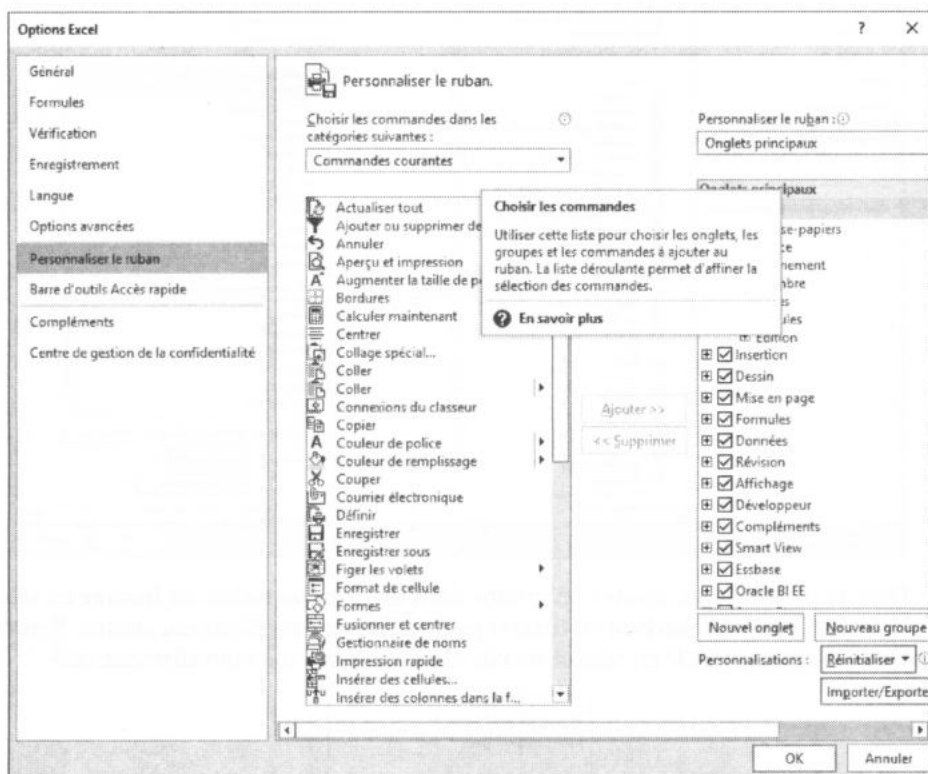
	A	B	C	D	E
1	Vente T1	35-37	38-40	41-43	44-46
2	Roller Freestyle	24	33	51	41
3	Roller Vitesse	14	21	25	30
4	Roller Street	19	24	32	28
5	Vente T2	35-37	38-40	41-43	44-46
6	Roller Freestyle	27	38	56	52
7	Roller Vitesse	16	24	30	38
8	Roller Street	21	29	38	33
9	Vente T3	35-37	38-40	41-43	44-46
10	Roller Freestyle	27	37	58	47
11	Roller Vitesse	16	24	29	33
12	Roller Street	22	26	35	31
13	Vente T4	35-37	38-40	41-43	44-46
14	Roller Freestyle	22	33	51	39
15	Roller Vitesse	13	20	23	29
16	Roller Street	20	25	34	28

Manipulation

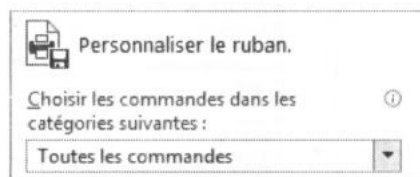
La première étape consiste à ajouter l'Assistant Tableau croisé dynamique dans Excel. Il s'agit d'ajouter une fonctionnalité dans le menu des fonctionnalités existantes. La description porte sur Excel 2016, toutefois, le processus est assez proche dans les autres versions d'Excel.

☞ Dans l'onglet Fichier, choisissez Options.

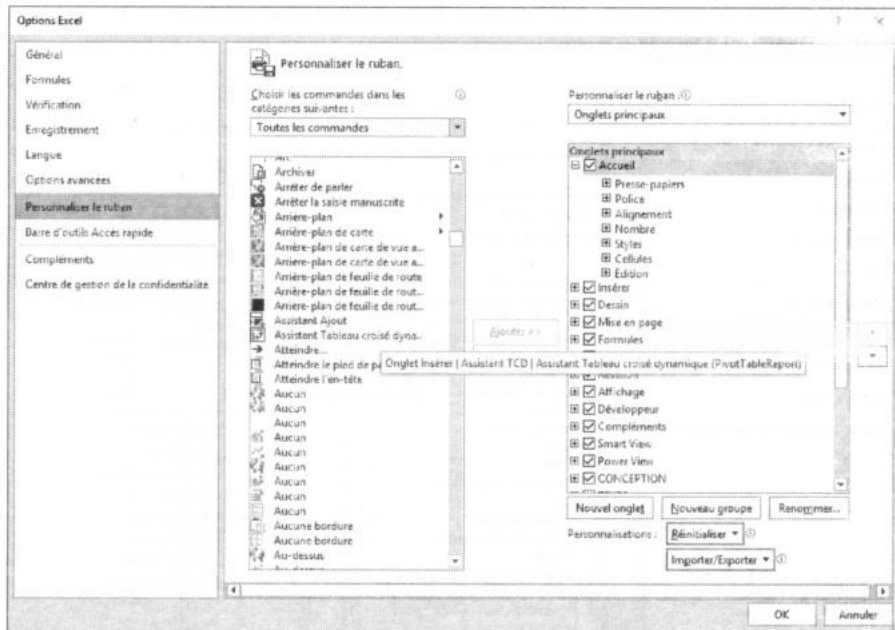
- ☞ Sur la gauche de la fenêtre, sélectionnez **Personnaliser le ruban**.
Le menu suivant s'affiche.



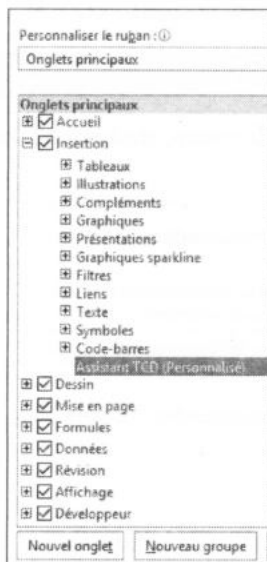
- ☞ Dans la zone **Personnaliser le ruban**, choisissez **Toutes les commandes** dans la liste déroulante de gauche.



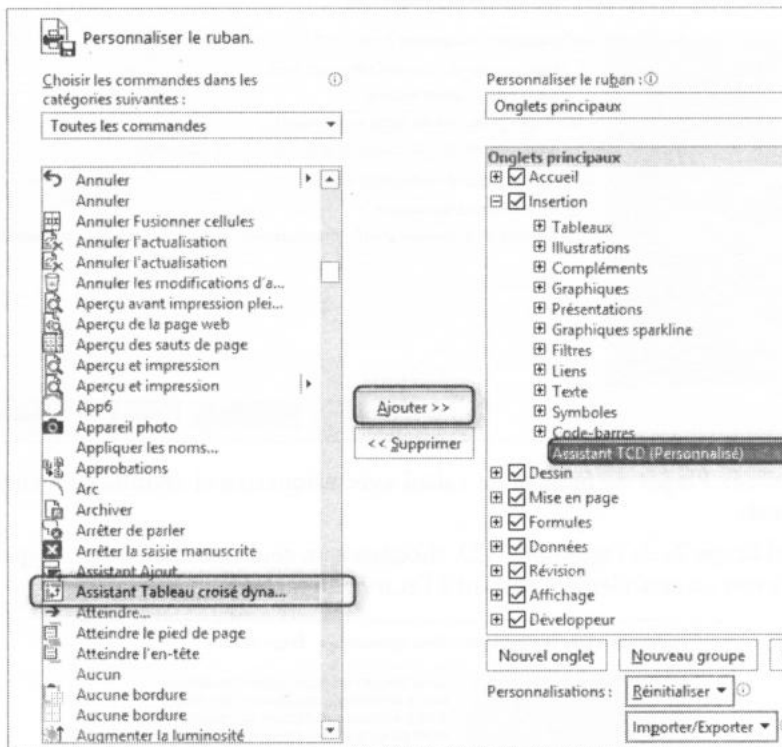
- ☞ Dans la longue liste des commandes disponibles, sélectionnez **Assistant Tableau croisé dynamique**.



- ☞ Dans la partie droite, ajoutez un groupe dans le menu **Insertion** ou **Insérer** en sélectionnant le menu **Insertion** ou **Insérer** puis en cliquant sur **Nouvel groupe**. Renommez-le **Assistant TCD** en faisant un clic droit sur le groupe nouvellement créé.



- Vous êtes désormais prêt à ajouter l'Assistant TCD dans le nouveau groupe Assistant TCD. Cliquez sur **Ajouter** pour finir l'opération.

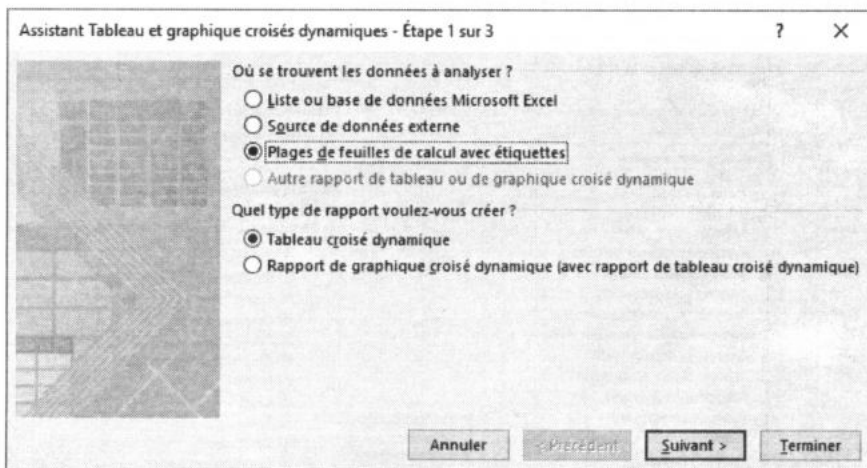


Voici le résultat final sur Excel :

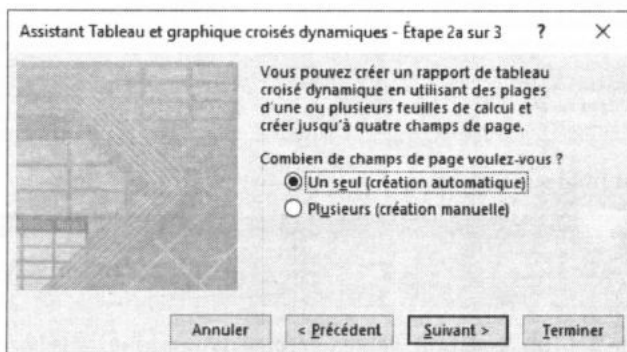


- Cliquez sur le bouton **Assistant Tableau croisé dynamique** dans le groupe personnalisé **Assistant TCD**.

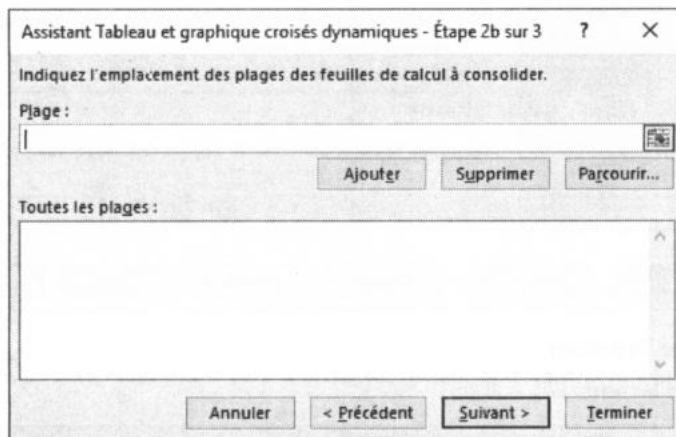
La fenêtre de l'Étape 1 de l'assistant TCD s'affiche.



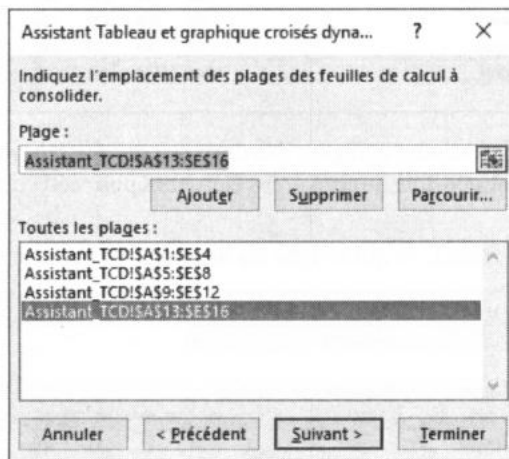
- ❖ Choisissez **Plages de feuilles de calcul avec étiquettes** et terminez en cliquant sur **Suivant**.
- ❖ Dans l'**Étape 2a** de l'assistant TCD, choisissez un **seul champ de page**, ce qui permettra d'avoir un seul élément filtrant à l'entrée du tableau.



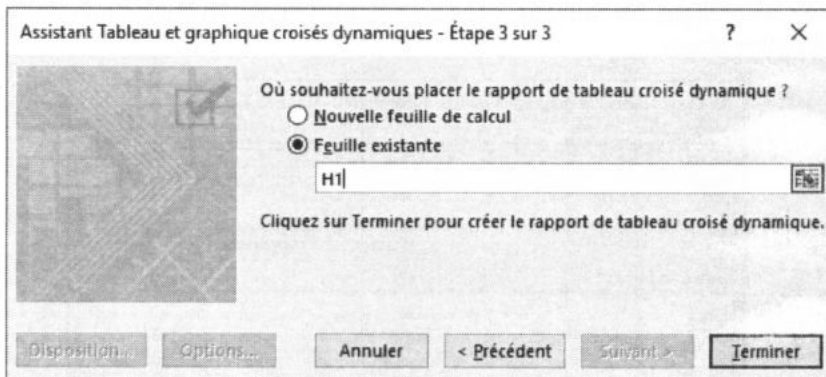
- ❏ Après avoir cliqué sur **Suivant**, vous arrivez sur l'**Étape 2b** qui permet d'ajouter les différentes pages.



- ❏ Sélectionnez la plage T1 qui correspond aux cellules A1:E4 puis cliquez sur **Ajouter**.
- ❏ Répétez cette opération pour les autres trimestres avec les plages A5:E8, A9:E12 et A13:E16 jusqu'à arriver au résultat suivant :



- ❏ Cliquez sur **Suivant**.
- ❏ Dans l'**Étape 3**, sélectionnez l'option **Feuille existante** puis choisissez la cellule H1 comme cellule de destination.



☞ Cliquez sur **Terminer**.

Le TCD s'affiche ainsi :

	H	I	J	K	L	M
Page1	(Tous)					
Somme de Valeur	Étiquettes de colonnes					
Étiquettes de lignes	35-37	38-40	41-43	44-46	Total général	
Roller Freestyle		100	141	216	179	636
Roller Street		82	104	139	120	445
Roller Vitesse		59	89	107	130	385
Total général		241	334	462	429	1466

Ce tableau contient les valeurs agrégées des données des quatre trimestres. Dans la liste déroulante **Page**, il est possible de choisir les éléments pour sélectionner un trimestre en particulier.

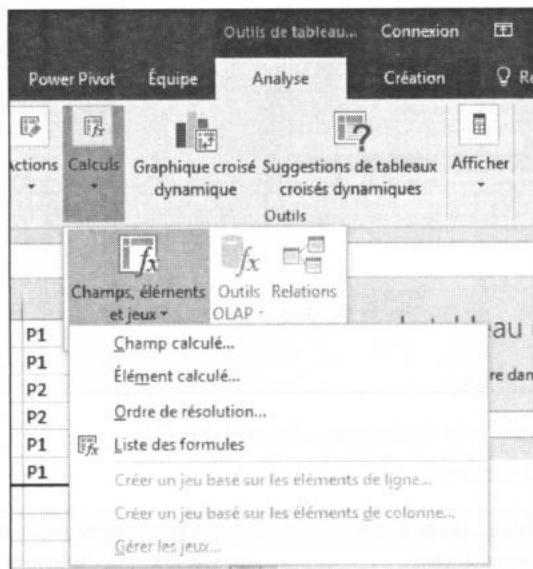
☞ Sélectionnez l'**Elément1**, ce qui affiche les données du Trimestre 1 :

	H	I	J	K	L	M
Page1	Elément1		.T			
Somme de Valeur	Étiquettes de colonnes					
Étiquettes de lignes	35-37	38-40	41-43	44-46	Total général	
Roller Freestyle		24	33	51	41	149
Roller Street		19	24	32	28	103
Roller Vitesse		14	21	25	30	90
Total général		57	78	108	99	342

Il est ensuite possible de naviguer facilement entre les différentes données du tableau.

c. Champs calculés et éléments calculés

Les calculs sont disponibles dans l'onglet Outils de tableau croisé dynamique - Analyse, plus particulièrement dans le groupe Calculs.



- ▶ Un **champ calculé** est une nouvelle donnée constituée à partir d'autres données déjà existantes.
- ▶ Un **élément calculé** est un nouvel axe constitué à partir d'axes existants.

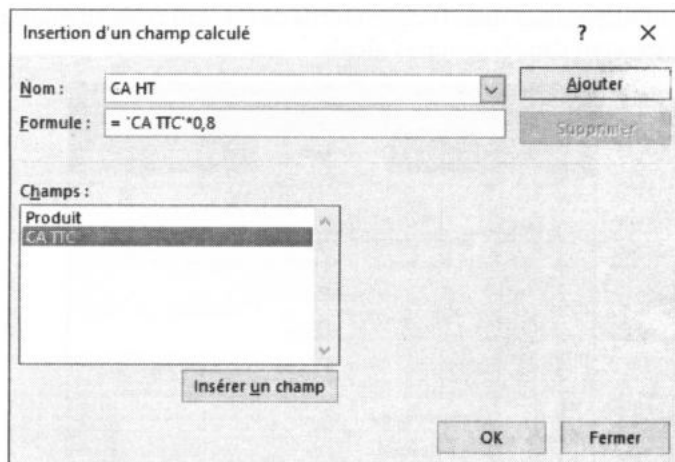
Exemple

- ☞ Dans le classeur **ExempleCours_Chapitre4.xlsm** feuille **ChampElementCalcule**, positionnez-vous sur la cellule D1.

	D	E
1	Étiquettes de lignes	Somme de CA TTC
2	Bague Argent	50 370,00 €
3	Bague Or	112 500,00 €
4	Bracelet Argent	7 575,00 €
5	Bracelet Or	21 900,00 €
6	Collier Argent	5 950,00 €
7	Collier Or	7 200,00 €
8	Total général	205 495,00 €

- ☞ Dans l'onglet **Outils de tableau croisé dynamique - Analyse** des outils TCD, cliquez sur **Calculs** puis sur le **champs, éléments et jeux**.
- ☞ Créez un **champ calculé**. La fenêtre **Insertion d'un champ calculé** apparaît.

- ☞ Créez une nouvelle formule qui a pour nom CA HT puis en Formule = 'CA TTC'*0,8.



- ☞ Cliquez sur **Ajouter** puis finissez en cliquant sur **OK**.

Un nouveau champ est créé :

Étiquettes de lignes	Somme de CA TTC	Somme de CA HT
Bague Argent	50 370,00 €	40 296,00 €
Bague Or	112 500,00 €	90 000,00 €
Bracelet Argent	7 575,00 €	6 060,00 €
Bracelet Or	21 900,00 €	17 520,00 €
Collier Argent	5 950,00 €	4 760,00 €
Collier Or	7 200,00 €	5 760,00 €
Total général	205 495,00 €	164 396,00 €

- ☞ Positionnez-vous à nouveau sur le TCD et plus particulièrement sur la cellule D1 puis créez cette fois-ci un **Élément calculé**.

La fenêtre suivante apparaît :

Insérer un élément calculé dans « Produit »

Nom :

Formule :

Champs :

- Produit
- CA TTC
- CA HT

Éléments :

- Bague Argent
- Bague Or
- Bracelet Argent
- Bracelet Or
- Collier Argent
- Collier Or

☞ Créez un champ Or correspondant à la somme des différents bijoux or.

Insérer un élément calculé dans « Produit »

Nom :

Formule :

Champs :

- Produit
- CA TTC
- CA HT

Éléments :

- Bague Argent
- Bague Or
- Bracelet Argent
- Bracelet Or
- Collier Argent
- Collier Or

Vous pouvez sélectionner les différents champs dans la zone Éléments.

- ☞ Terminez en cliquant sur **Ajouter** puis sur **OK** pour voir le champ apparaître dans votre TCD.

Étiquettes de lignes	Somme de CA TTC	Somme de CA HT
Bague Argent	50 370,00 €	40 296,00 €
Bague Or	112 500,00 €	90 000,00 €
Bracelet Argent	7 575,00 €	6 060,00 €
Bracelet Or	21 900,00 €	17 520,00 €
Collier Argent	5 950,00 €	4 760,00 €
Collier Or	7 200,00 €	5 760,00 €
Or	141 600,00 €	113 280,00 €
Total général	347 095,00 €	277 676,00 €

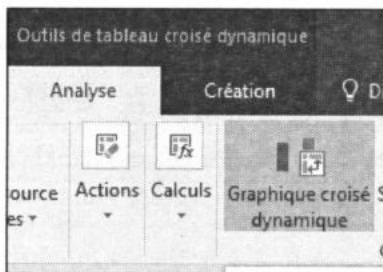
- ☞ Cliquez sur la liste déroulante **Étiquettes de lignes** puis désélectionnez **Bague Or**, **Bracelet Or** et **Collier Or**.

Étiquettes de lignes	Somme de CA TTC	Somme de CA HT
Bague Argent	50 370,00 €	40 296,00 €
Bracelet Argent	7 575,00 €	6 060,00 €
Collier Argent	5 950,00 €	4 760,00 €
Or	141 600,00 €	113 280,00 €
Total général	205 495,00 €	164 396,00 €

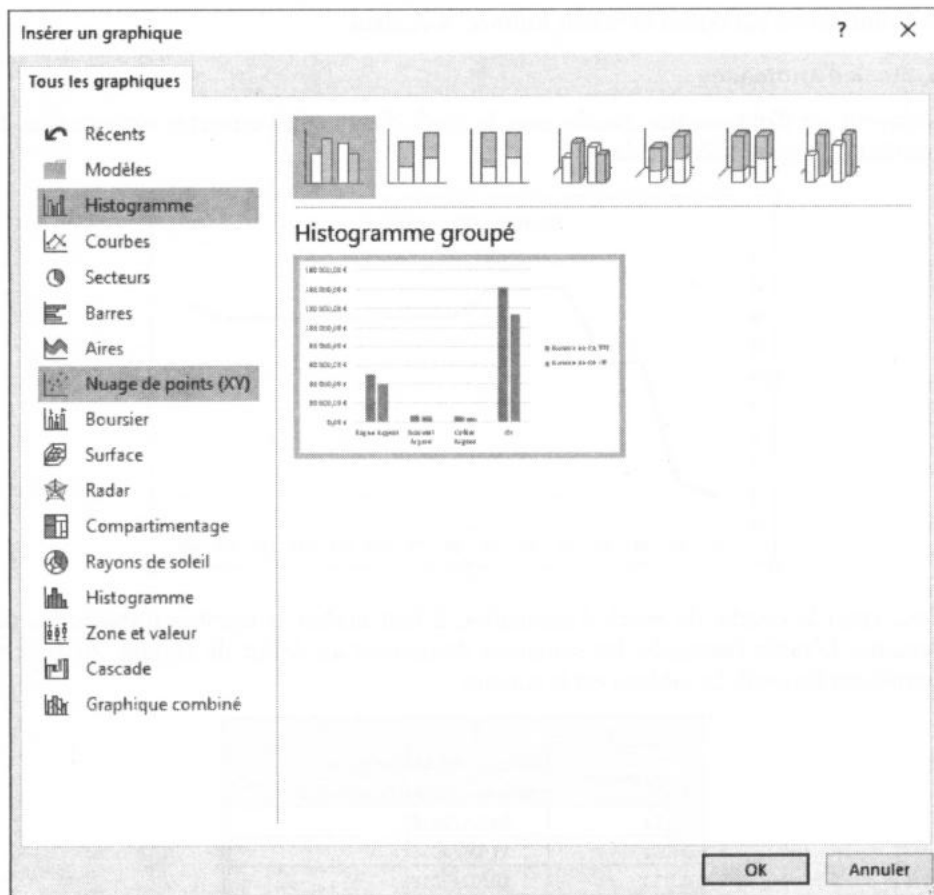
d. Créer un graphique croisé dynamique

Un graphique croisé dynamique est un graphique issu d'un tableau croisé dynamique.

- ☞ Pour le créer, positionnez-vous sur un tableau croisé dynamique, puis cliquez sur l'onglet **Outils de tableau croisé dynamique - Analyse** et cliquez sur **Graphique croisé dynamique**.



Une fenêtre **Insérer un graphique** apparaît.



☛ Choisissez votre type de graphique et terminez en cliquant sur OK.

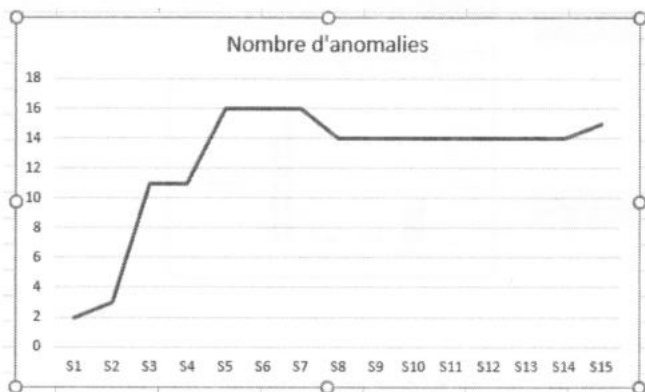
Le graphique apparaît alors sur la feuille en cours.

3. Réalisation de l'exemple

☞ Commencez par ouvrir le fichier `Enoncé_4-A.xlsm`.

a. Stock d'anomalies

L'objectif est d'obtenir une courbe avec le stock d'anomalies ouvertes par semaine. Le résultat doit ressembler à cela :



Pour créer la courbe du stock d'anomalies, il faut établir le nombre d'anomalies par semaine. D'après l'exemple, les semaines démarrent au début de l'année 2016 et se terminent fin avril. Le tableau est le suivant :

	A	B	C
1	Numéro	Date de début de la semaine	Nombre d'anomalies
2	S1	04/01/2016	
3	S2	11/01/2016	
4	S3	18/01/2016	
5	S4	25/01/2016	
6	S5	01/02/2016	
7	S6	08/02/2016	
8	S7	15/02/2016	
9	S8	22/02/2016	
10	S9	29/02/2016	
11	S10	07/03/2016	
12	S11	14/03/2016	
13	S12	21/03/2016	
14	S13	28/03/2016	
15	S14	04/04/2016	
16	S15	11/04/2016	

L'objectif est de remplir la colonne C avec le nombre d'anomalies présentes à la date du jour de début de la semaine contenue en colonne B.

Comment faire ?

Il est nécessaire de déterminer le nombre d'anomalies en cours à la date de début de la semaine. Pour cela, il faut comptabiliser les anomalies si elles valident les deux conditions suivantes :

- ▶ Les anomalies dont la date de création est antérieure ou égale à la date de début de la semaine ;
- ▶ Les anomalies dont la date de fermeture est supérieure ou égale à la date de début de la semaine ou non clôturée.

En sommant ces deux conditions, le résultat correspondra au nombre d'anomalies ouvertes à la date du jour de début de la semaine.

La formule comprend deux parties : anomalies clôturées et anomalies ouvertes. Il faut utiliser la formule NB.SI.ENS :

```
= NB.SI.ENS (date_ouverture ; "<="&date_debut_semaine ;
date_fermeture ; ">="&date_debut_semaine) ;
= NB.SI.ENS (date_ouverture ; "<="&date_debut_semaine ;
date_fermeture ; vide) ;
```

Les deux conditions sont sommées.

```
=NB.SI.ENS (Anomalies!$C$2:$C$27;"<="&B2;Anomalies!$F$2:$F$27;">="&B2
+NB.SI.ENS (Anomalies!$C$2:$C$27;"<="&B2;Anomalies!$F$2:$F$27;"")
```

- 👉 Appliquez cette formule à la plage C2:C16.

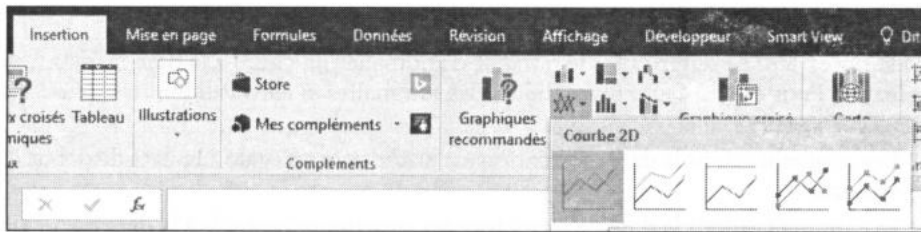


Attention, cette formule s'applique uniquement à la plage d'anomalies A1:F27 et ne fonctionne que pour les 26 anomalies détectées : une anomalie supplémentaire ne serait pas prise en compte. Pour pallier ce problème, il faut transformer le tableau contenant les anomalies en Tableau au sens Excel. Ainsi il sera possible de récupérer une colonne entière du tableau quelle que soit sa taille.

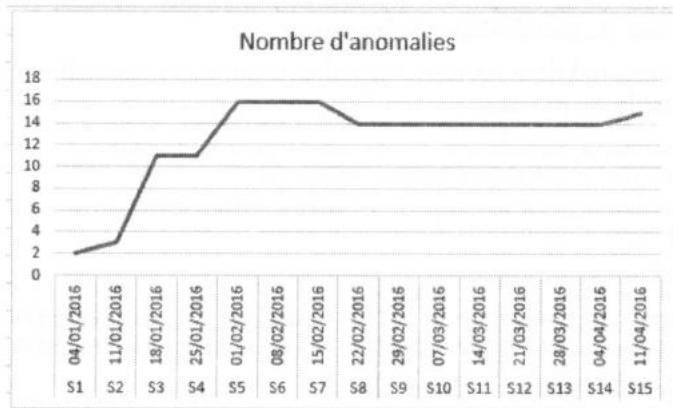
Ensuite, nous insérons le graphique :

- 👉 Sélectionnez la plage A1:C16.
- 👉 Dans l'onglet Insérer - groupe Graphiques cliquez sur l'icône Insérer un graphique en courbes ou en aires.

- La liste des courbes disponibles s'affiche, sélectionnez le graphique **Trait** dans la zone **Courbe 2D**.

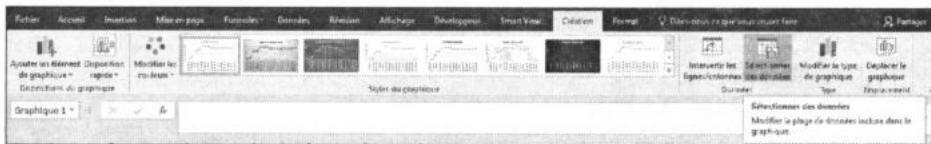


Le graphique s'affiche ainsi :

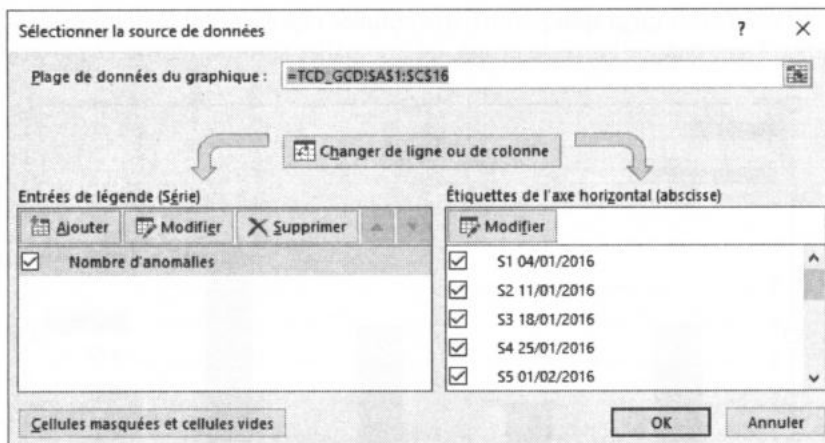



L'axe des abscisses contient la date et le numéro de semaine. Dans notre exemple, nous souhaitons afficher uniquement le numéro de semaine.

- Pour retirer la date, cliquez sur l'onglet **Outils de graphique - Création** - groupe **Données**, choisissez **Sélectionner des données**.



Une fenêtre s'affiche avec la possibilité de revoir la source de données :



☞ Modifiez la source en cliquant sur le bouton .

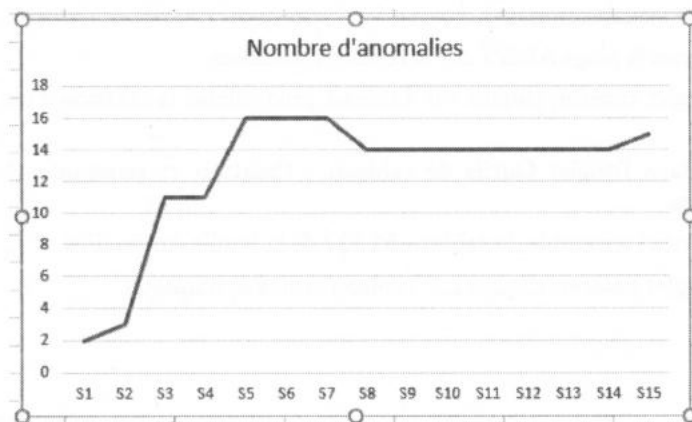
Nous allons donc sélectionner une plage de données différente en sélectionnant la plage des numéros de semaine (A1:A16) et la plage du nombre d'anomalies (C1:C16).

☞ Sélectionnez d'abord la plage A1:A16 puis tout en gardant la touche **Ctrl** appuyée sélectionnez la plage C1:C16 puis validez.

Le résultat est le suivant dans la zone **Plage des données du graphique** :

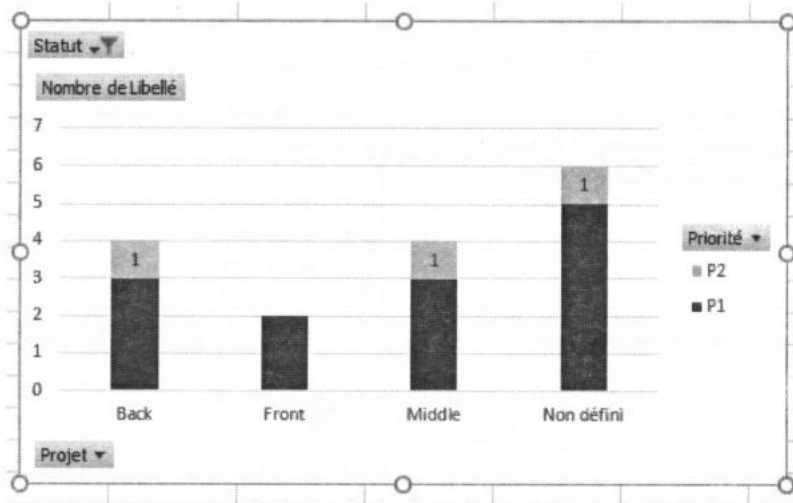
= 'TCD_GCD' ! \$A\$1 : \$A\$16 ; 'TCD_GCD' ! \$C\$1 : \$C\$16

Votre graphique s'affiche donc ainsi :



b. Nombre d'anomalies par projets (et par priorité)

L'objectif est d'avoir un graphique en barre empilée représentant le nombre d'anomalies par projet. Dans chaque barre de projet, les anomalies seront présentées par priorité.



Dans un premier temps, il faudra créer un tableau croisé dynamique basé sur la plage actuelle des anomalies sur la feuille **Anomalies** (A1:I27) pour pouvoir organiser les données afin d'afficher le graphique croisé dynamique ensuite.

L'intérêt d'avoir un tableau croisé dynamique réside dans la potentielle évolution de la plage de données. En effet, le tableau croisé dynamique s'adapte dynamiquement aux données sources, par conséquent il n'y a pas de risque d'avoir un décalage entre les données et la visualisation.

- ☞ Sélectionnez la plage A1:I27 sur la feuille **Anomalies**.
- ☞ Dans l'onglet **Insérer**, cliquez sur **Tableau**, puis validez la création d'un tableau avec en-tête.
- ☞ Cliquez dans l'onglet **Outils de tableau - Création** et renommez le tableau en **Anomalies**.
- ☞ Sélectionnez l'ensemble du tableau A1:I27 de la feuille **Anomalies**.
- ☞ Dans l'onglet **Insérer**, cliquez sur **Tableau croisé dynamique**.

- Sur la fenêtre de création du TCD, dans la partie Choisissez l'emplacement de votre rapport de tableau croisé dynamique, sélectionnez la cellule A25 de la feuille TCD_GCD.

- Cliquez sur OK.

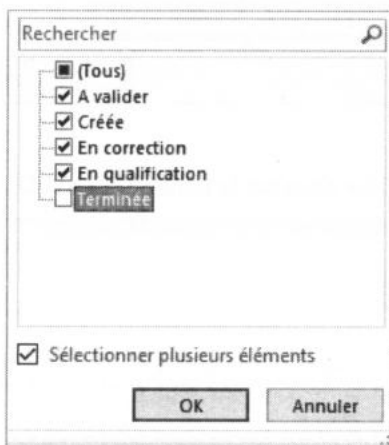
La structure du TCD s'affiche alors sur la feuille TCD_GCD :

- Glissez et déposez le champ **Projet** dans la zone **Lignes**.
- Glissez et déposez le champ **Priorité** dans la zone **Colonnes**.
- Enfin, glissez et déposez le champ **Libellé** dans la zone **Σ Valeurs**.

- ☞ Déplacez le champ **Statut** dans la zone **Filtres**. Cela permettra de filtrer le rapport en fonction des valeurs de statut choisies.

23	Statut	(Tous)				
24						
25	Nombre de Libellé	Étiquettes de colonnes				
26	Étiquettes de lignes	P1	P2	P3	Total général	
27	Back		4	3	1	8
28	Front		2	3		5
29	Middle		3	3	1	7
30	Non défini		5	1		6
31	Total général		14	10	2	26

- ☞ En B23, déroulez les statuts possibles puis cliquez sur **Sélectionner plusieurs éléments**.
- ☞ Décochez le statut **Terminée** : seules les anomalies en cours seront intégrées et calculées dans le TCD.



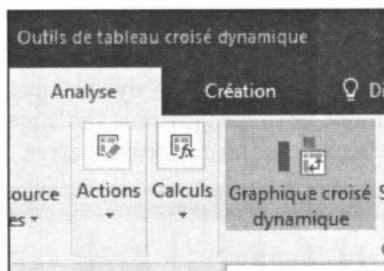
Filtrer le rapport au niveau de la zone **Filtre du Rapport** permet une meilleure optimisation du TCD. Filtrer au niveau de la page permet de filtrer la source de données et non les données à l'intérieur du TCD. Le filtre des données est plus long s'il est fait au sein du TCD que s'il avait été fait dans la source de données.

Le TCD s'affiche donc ainsi :

23	Statut	(Plusieurs éléments)		
24				
25	Nombre de Libellé	Étiquettes de colonnes		
26	Étiquettes de lignes	P1	P2	Total général
27	Back	3	1	4
28	Front	2		2
29	Middle	3	1	4
30	Non défini	5	1	6
31	Total général	13	3	16

La seconde partie consiste à afficher le graphique croisé dynamique :

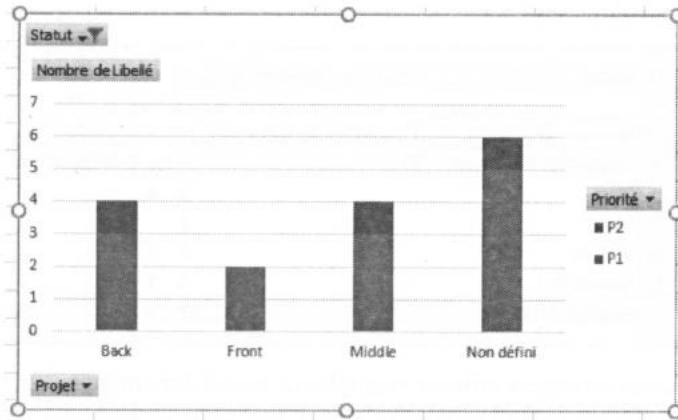
- ☞ Cliquez sur l'onglet Outils du tableau croisé dynamique - Analyse.



- ☞ Dans le groupe Outils, cliquez sur **Graphique croisé dynamique**, choisissez le graphique **Histogramme empilé** puis cliquez sur OK.

Le graphique apparaît, ainsi que les onglets groupés sous le nom Outils de Graphique croisé dynamique.

- ☞ Repositionnez le graphique pour le rendre plus visible.



Nous allons ajouter les étiquettes de données afin d'afficher les valeurs des données sur les différentes barres du graphique.

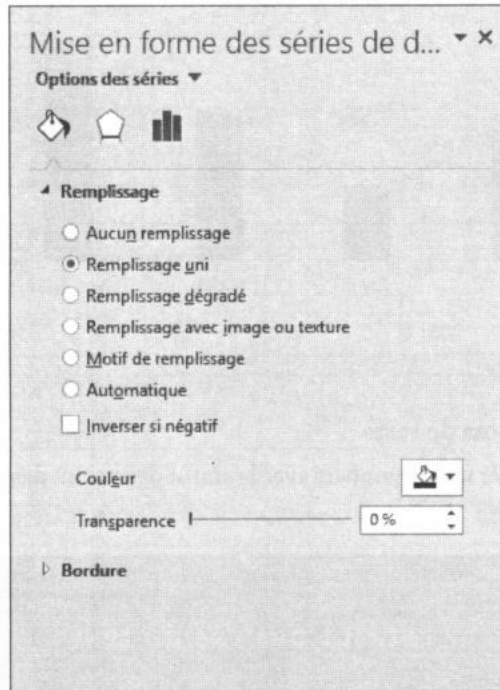
- ✎ Dans l'onglet **Outils de Graphique croisé dynamique - Création**, cliquez sur le bouton **Ajouter un élément de graphique** puis choisissez **Étiquettes de données et Au centre** pour afficher les valeurs au centre des barres.

The screenshot shows the Excel ribbon with the 'Ajouter un élément de graphique' menu open. The 'Au centre' option is selected. The background shows the same bar chart as in the previous figure, but with the 'Ajouter un élément de graphique' menu open over it.



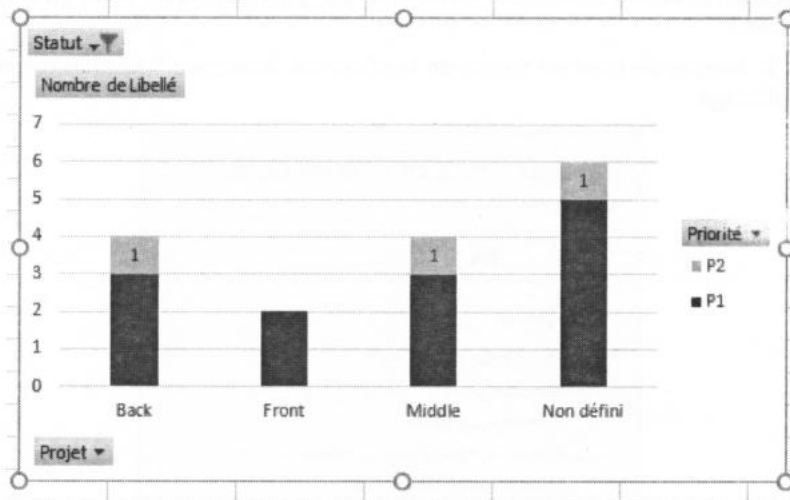
Dans les versions antérieures d'Excel, cette fonctionnalité se trouve (ainsi que toutes les fonctionnalités liées à la disposition) dans l'onglet **Disposition - Étiquettes de données**.

- ❏ Dans l'onglet **Outils de graphique croisé dynamique - Format**, dans le groupe **Sélection Active**, sélectionnez la série **Série "P1"** puis cliquez sur **Mise en forme de la sélection**.
- ❏ Dans la fenêtre de mise en forme de la sélection choisissez **Remplissage et ligne - Remplissage**.



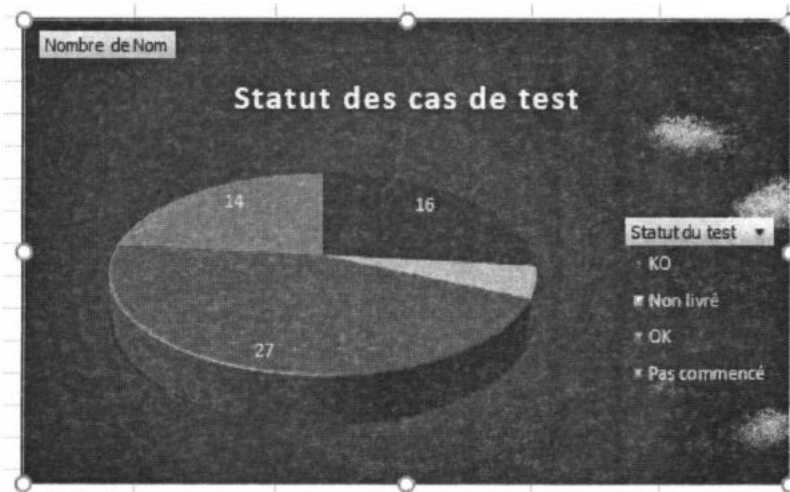
- ❏ Cochez **Remplissage uni** puis sélectionnez la couleur rouge pour la série P1.
- ❏ Appliquez la couleur orange pour la série P2.

Le résultat obtenu est le suivant :



c. Avancement des cas de tests

L'objectif est d'afficher un camembert avec le statut de chacun des cas de test. Le résultat doit ressembler à cela :



Pour cela :

- ❖ Sélectionnez la plage A1:D61 sur la feuille Test.
- ❖ Dans l'onglet **Insérer**, cliquez sur **Tableau**, puis validez la création d'un tableau avec en-tête.
- ❖ Cliquez dans l'onglet **Outils de tableau - Création** et renommez le tableau Test.
- ❖ Sélectionnez l'ensemble du tableau Test.
- ❖ Dans l'onglet **Insérer**, cliquez sur **Tableau croisé dynamique**.
- ❖ Positionnez le TCD sur la feuille TCD_GCD sur la cellule A40 :
 - ▶ Cochez l'option **Feuille de calcul existante**.
 - ▶ Positionnez-vous sur la cellule A40 de la feuille TCD_GCD.

Créer un tableau croisé dynamique ? X

Choisissez les données à analyser

Sélectionner un tableau ou une plage

Tableau/Plage : Test

Utiliser une source de données externes

Choisir la connexion...

Nom de la connexion :

Utiliser le modèle de données de ce classeur

Choisissez l'emplacement de votre rapport de tableau croisé dynamique

Nouvelle feuille de calcul

Feuille de calcul existante

Emplacement : TCD_GCD!\$A\$40

Indiquez si vous souhaitez analyser plusieurs tables

Ajouter ces données au modèle de données

OK Annuler

Après avoir cliqué sur OK, le socle du TCD est disponible et il est possible de glisser-déposer les champs via la partie droite.

- ❖ Positionnez le champ **Statut du test** dans la zone **Lignes**.
- ❖ Positionnez le champ **Nom** dans la zone Σ Valeurs.

Champs de tableau crois... x

Choisissez les champs à inclure dans le rapport :

Rechercher

Nom
 Priorité
 Statut du test
 Fonctionnalité

Plus de tableaux...

Faites glisser les champs dans les zones voulues ci-dessous:

Filtres
 Colonnes

Lignes
 Statut du test

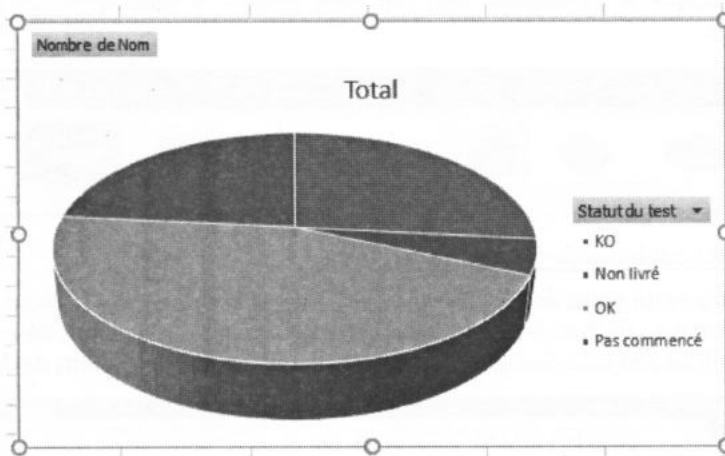
Valeurs
 Nombre de Nom

Différer la mise à jour de la disposition Mettre à jour

	A	B
40	Étiquettes de lignes	Nombre de Nom
41	KO	16
42	Non livré	3
43	OK	27
44	Pas commencé	14
45	Total général	60

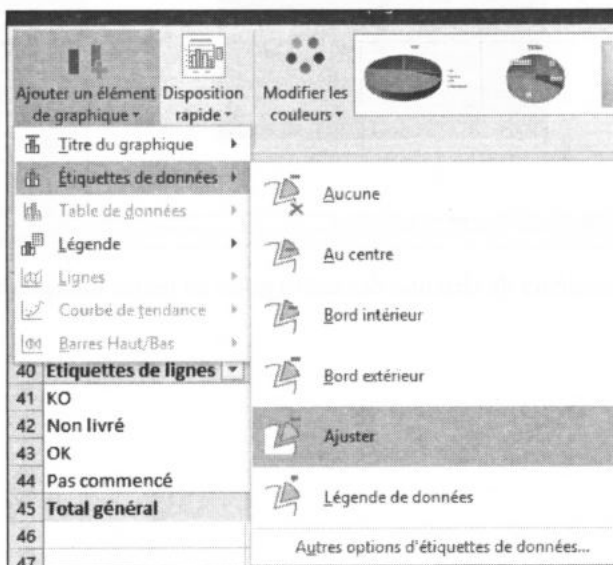
- ☞ Sélectionnez le TCD créé.
- ☞ Dans l'onglet Outils de tableau croisé dynamique - Analyse, cliquez sur Graphique croisé dynamique.
- ☞ Choisissez ici le graphique en Secteurs puis Secteur en 3D.

Le résultat est le suivant :



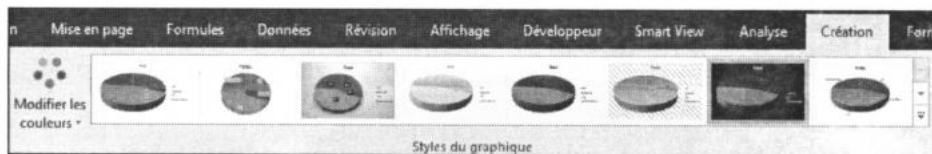
Nous allons ajouter ensuite les étiquettes de données pour avoir le nombre de chaque Statut de test.

- ☞ Dans l'onglet Outils de graphique croisé dynamique - Création, cliquez sur Ajouter un élément de graphique, puis choisissez Étiquettes de données - Ajuster.



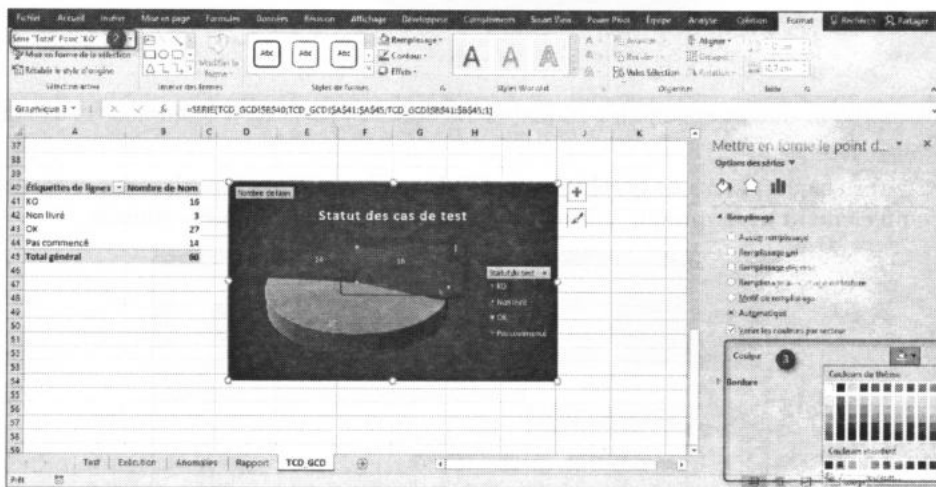
- ☞ Sélectionnez le titre et double cliquez sur celui-ci pour le modifier en Statut des cas de test.

- ☞ Dans l'onglet Outils de graphique croisé dynamique - Création, parcourez les Styles du graphique et choisissez celui qui vous semble le plus approprié parmi ceux proposés :



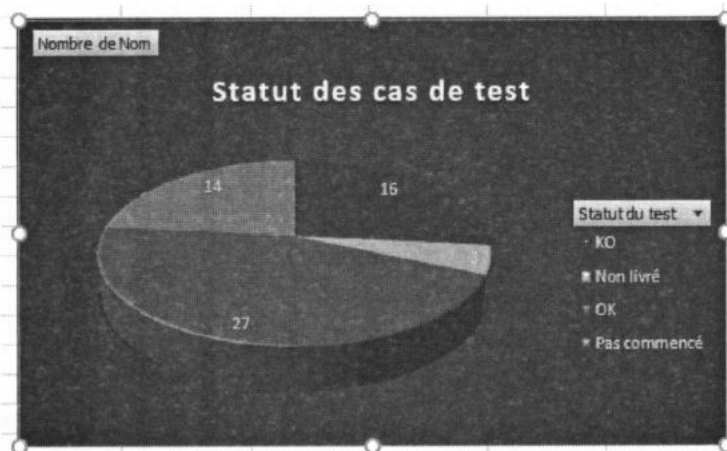
Modifier les couleurs de chaque série

- ☞ Sélectionnez un point de données directement sur le graphique (1). Vous pouvez vérifier la bonne sélection du point dans l'onglet Format dans le groupe Sélection active (2). Modifiez son remplissage dans le volet Mettre en forme le point de données (3).



- ☞ Changez les couleurs de chacune des séries grâce au menu Remplissage.

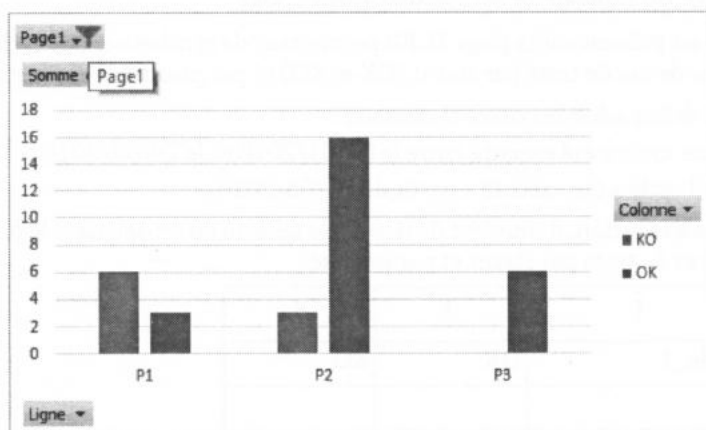
Voici le type de résultat que vous pouvez obtenir :



d. Revue des cycles de test

L'objectif est d'afficher un tableau consolidé des cycles de test où il est possible de voir le nombre de tests avec statut OK, le nombre de tests avec statut KO et le nombre de tests par campagne et par cycle.

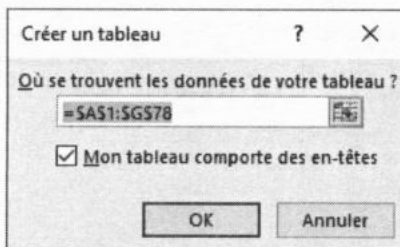
Le résultat attendu est le suivant :



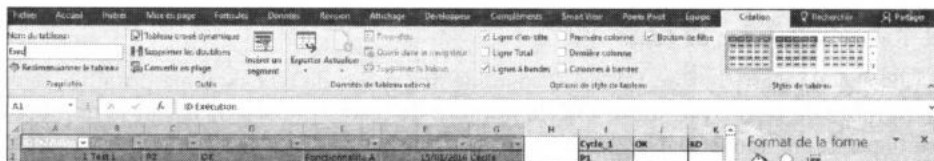
Pour réaliser cet exemple, nous allons tout d'abord créer un tableau à partir des données de la feuille Exécution, puis synthétiser les données du premier tableau pour créer un TCD avec les données synthétisées. Nous terminerons par la création d'un graphique croisé dynamique basé sur les données du TCD.

Création du tableau

- ❏ Sélectionnez la plage de données A1:G78 de la feuille **Exécution**.
- ❏ Cliquez sur l'onglet **Insérer** puis cliquez sur **Tableau**.
- ❏ Choisissez un tableau avec en-têtes :



- ❏ Dans l'onglet **Outils de tableau - Création - groupe Propriétés**, renommez le tableau en **Exec**.



Remplissage du tableau de récapitulatif des cycles

Un tableau est présent sur la plage I1:K8 permettant de synthétiser par cycle, le nombre d'exécutions de cas de tests par statut (OK et KO) et par priorité (P1, P2 et P3).

Le cycle est défini selon le critère ci-dessous :

- ▶ Le premier cycle a été exécuté entre le 01/01/2016 et le 29/02/2016 ;
- ▶ Le second cycle a été exécuté à partir du 01/03/2016.

Sur la feuille **Exécution**, il convient de remplir le tableau en renseignant le nombre d'exécutions de cas de tests par statut et par priorité :

	I	J	K
1	Cycle_1	OK	KO
2	P1		
3	P2		
4	P3		
5	Cycle_2	OK	KO
6	P1		

	I	J	K
7	P2		
8	P3		

Quelles formules doivent être utilisées ?

La formule va consister à compter le nombre de lignes avec les conditions suivantes :

- ▶ Nombre de lignes dont le statut est égal aux valeurs OK et KO qui correspondent respectivement aux cellules J1 et K1 pour le cycle 1 et aux cellules J5 et K5 pour le cycle 2.
- ▶ Nombre de lignes dont la priorité est égale aux valeurs P1, P2 et P3 qui correspondent respectivement aux cellules I2, I3 et I4 pour le cycle 1 et aux cellules I6, I7 et I8 pour le cycle 2.
- ▶ Nombre de lignes dont la date d'exécution est comprise entre le 01/01/2016 et le 29/02/2016 (ou supérieure ou égale au 01/03/2016).

Il faut donc utiliser la formule NB.SI.ENS qui permet de compter les lignes avec des conditions.

Manipulation

- ☞ Sélectionnez la plage J2:K4 qui correspond au cycle 1, appuyez sur la touche **F2** puis saisissez la formule suivante :

```
=NB.SI.ENS(Exec[Priorité];$I2;Exec[Statut de l'exécution];
J$1;Exec[Date d'exécution];">=01/01/2016";Exec[Date d'exécution];
"<=29/02/2016")
```

- ☞ Validez la formule en appuyant simultanément sur les touches **Ctrl** **↵**.
- ☞ Pour la plage J6:K8 qui correspond au cycle 2, appuyez sur la touche **F2** puis saisissez la formule suivante :

```
=NB.SI.ENS(Exec[Priorité];$I6;Exec[Statut de l'exécution];
J$1;Exec[Date d'exécution];">=01/03/2016")
```

- ☞ Appliquez la formule avec les touches **Ctrl** **↵**.

Le résultat est le suivant :

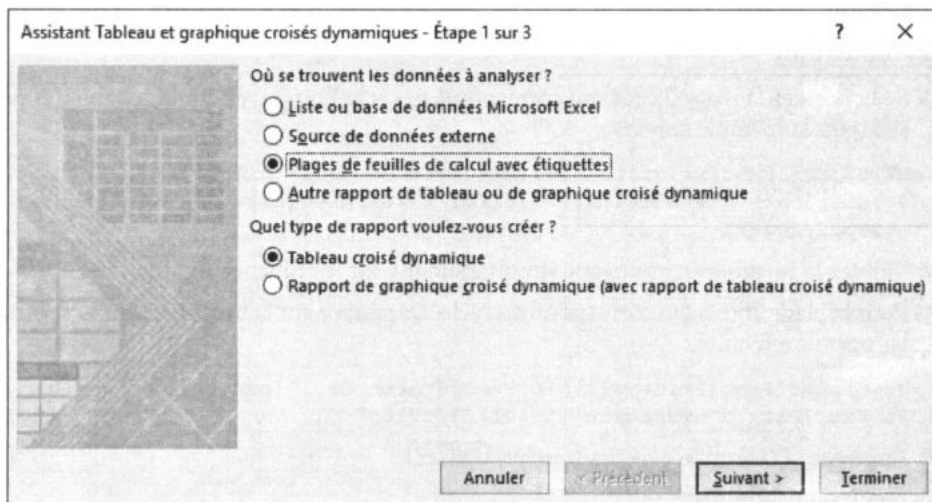
Cycle_1	OK	KO
P1	4	14
P2	13	6
P3	3	3

Cycle_2	OK	KO
P1	3	6
P2	16	3
P3	6	0

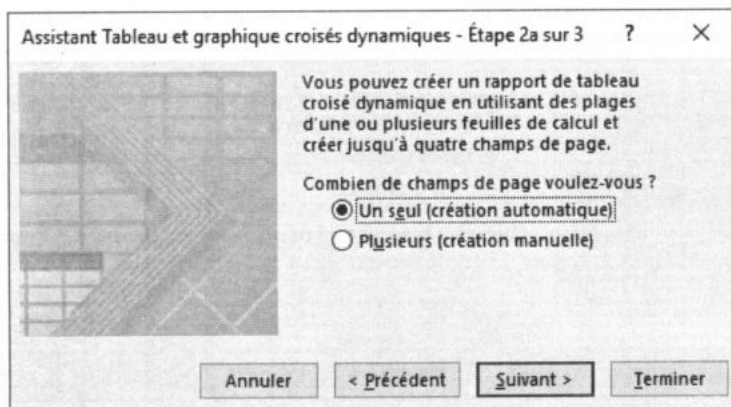
Création d'un tableau croisé dynamique avec page

Nous avons identifié deux plages de données correspondant aux deux cycles à analyser. Nous allons créer un TCD avec page pour consolider ces deux plages de données sur un même TCD.

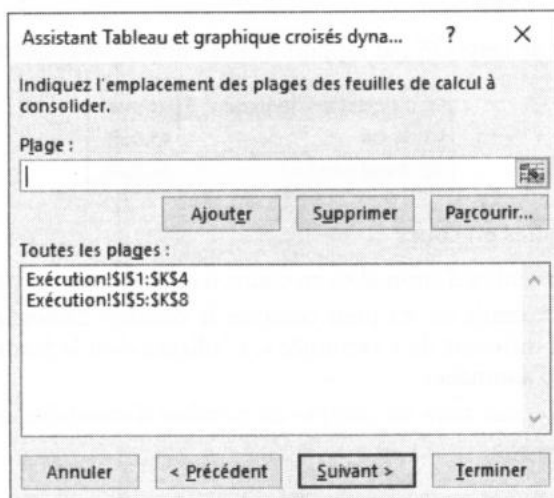
- Utilisez le tutoriel des notions de cours de ce chapitre pour afficher le bouton **Assistant Tableau croisé dynamique**.
- Lancez l'Assistant Tableau croisé dynamique.
- Dans la fenêtre de l'Étape 1 de l'assistant TCD, choisissez l'option **Plages de feuilles de calcul avec étiquettes**.



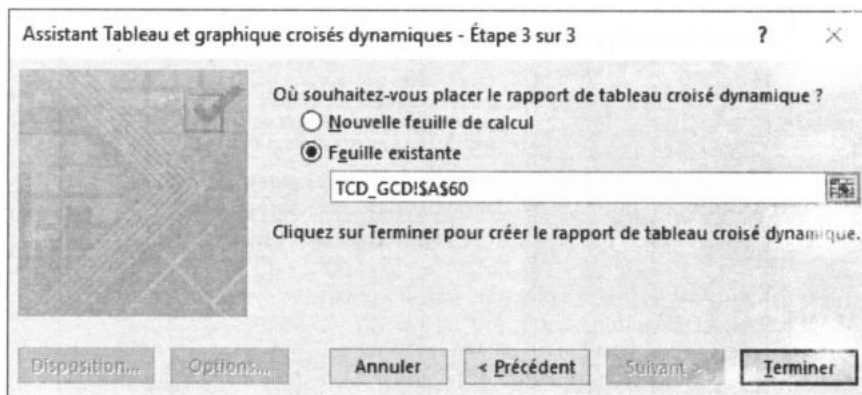
- ☞ Cliquez sur **Suivant**, puis dans l'**Étape 2a** de l'assistant, choisissez un seul champ de page avec création automatique.



- ☞ Après avoir cliqué sur **Suivant**, vous arrivez sur l'**Étape 2b** qui consiste à ajouter les différentes plages. Ici les plages à ajouter sont I1:K4 et I5:K8.
- ☞ Dans la zone plage, ajoutez la plage I1:K4 en saisissant manuellement ou en sélectionnant la plage.
- ☞ Cliquez sur **Ajouter** afin d'insérer la plage.
- ☞ Dans la zone **Plage**, ajoutez la plage I5:K8 en saisissant manuellement ou en sélectionnant la plage.
- ☞ Cliquez sur **Ajouter** afin d'insérer la plage.



- ☞ Cliquez sur **Suivant** pour afficher l'Étape 3. Choisissez comme cellule de destination la cellule A60 de la feuille TCD_GCD.



Le tableau s'affiche sur la cellule A60.

e. Indicateur de situation des tests

L'objectif est d'afficher deux indicateurs chiffrés :

- Le nombre d'anomalies en cours :

78	Statut	(Plusieurs éléments)	↕
79			
80	Nombre de Libellé		
81		16	

- Le pourcentage de tests OK par rapport à l'ensemble des cas de test :

90	Étiquettes de lignes	↕	Valeur
91	% OK		45,00%
92	Total général		45,00%

Nombre d'anomalies en cours

Pour connaître le nombre d'anomalies en cours, il existe plusieurs possibilités :

- Appliquer une formule **NB.SI** pour compter le nombre d'anomalies sur la colonne ayant un statut différent de « Terminée ». D'ailleurs c'est la formule utilisée dans le calcul du stock d'anomalies.
- Utiliser un TCD pour faire un compte du nombre d'anomalies avec un filtre sur le statut des anomalies.

Même si une formule Excel est simple, un TCD est plus facilement modifiable (exemple : ajout de condition au filtre, changement du libellé de statut), par conséquent, l'exemple sera basé sur un TCD.



Contrairement aux autres TCD dont la dimension varie, celui-ci ne comporte qu'une seule valeur, le nombre d'anomalies. Par conséquent, il sera plus facile d'en extraire les données.

Pour réaliser cet indicateur :

- ☞ Sélectionnez la cellule A80 de la feuille TCD_GCD.
- ☞ Dans l'onglet **Insérer**, cliquez sur **Tableau croisé dynamique**.

Comme nous n'avons pas sélectionné une source de données mais une cellule de destination, vous pouvez constater que l'emplacement de destination est rempli, mais pas la source de données.

- ☞ Dans **Tableau/Plage**, saisissez **Anomalies**.

Créer un tableau croisé dynamique

Choisissez les données à analyser

Sélectionner un tableau ou une plage

Tableau/Plage : Anomalies

Utiliser une source de données externes

Choisir la connexion...

Nom de la connexion :

Utiliser le modèle de données de ce classeur

Choisissez l'emplacement de votre rapport de tableau croisé dynamique

Nouvelle feuille de calcul

Feuille de calcul existante

Emplacement : TCD_GCD!\$A\$80

Indiquez si vous souhaitez analyser plusieurs tables

Ajouter ces données au modèle de données

OK Annuler

- ☞ Cliquez sur **OK**.

- Positionnez le champ **Libellé** dans la zone Σ Valeurs et le champ **Statut** dans la zone **Filtres** tel que ci-dessous :

Champs de tableau croisé d... ✕

Choisissez les champs à inclure dans le rapport : ⚙️

Rechercher 🔍

Date ouverture ticket
 Statut
 Test associé
 Date de cloture
 Projet
 Testeur

Faites glisser les champs dans les zones voulues ci-dessous:

📏 Filtres
 Statut ▼

📊 Colonnes

📄 Lignes

Σ Valeurs
 Nombre de Libellé ▼

Différer la mise à jour de la dispositi... Mettre à jour

- Filtrez le statut en retirant les anomalies terminées :

Rechercher 🔍

- (Tous)
- A valider
- Créée
- En correction
- En qualification
- Terminée

Sélectionner plusieurs éléments

Le résultat sera toujours présent dans la cellule A81 de la feuille TCD_GCD : il contient le nombre d'anomalies.

78	Statut	(Plusieurs éléments)
79		
80	Nombre de Libellé	
81		16

Proportion de tests OK

Le pourcentage de tests OK n'est pas présent directement dans les champs d'un TCD issu de la feuille Test. En revanche, il est possible de créer un élément calculé qui donnera le nombre de tests OK divisé par l'ensemble des tests.

- ☞ Sélectionnez la cellule A90 de la feuille TCD_GCD puis dans l'onglet Insérer, cliquez sur Tableau croisé dynamique.
- ☞ Dans Tableau/Plage, saisissez Test.

? X

Créer un tableau croisé dynamique

Choisissez les données à analyser

Sélectionner un tableau ou une plage

Tableau/Plage :

Utiliser une source de données externes

Nom de la connexion :

Utiliser le modèle de données de ce classeur

Choisissez l'emplacement de votre rapport de tableau croisé dynamique

Nouvelle feuille de calcul

Feuille de calcul existante

Emplacement :

Indiquez si vous souhaitez analyser plusieurs tables

Ajouter ces données au modèle de données

- ☞ Cliquez sur OK.

- ☞ Positionnez le champ **Statut du test** dans la zone **Lignes** et le champ **Nom** dans la zone **Σ Valeurs** pour obtenir le résultat suivant :

90	Étiquettes de lignes	Nombre de	Nom
91	KO	16	
92	Non livré	3	
93	OK	27	
94	Pas commencé	14	
95	Total général	60	

- ☞ Sélectionnez la plage A91:A94 puis dans l'onglet **Outils de tableau croisé dynamique - Analyse**, cliquez sur **Calculs** puis dans **Champs, éléments et jeux**, choisissez **Élément calculé**.

L'objectif est de créer un champ % OK qui calculera le pourcentage de tests OK sur l'ensemble des tests. La fenêtre s'affiche ainsi :

Insérer un élément calculé dans « Statut du test »

Nom :

Formule :

Champs :

- Nom
- Priorité
- Statut du test
- Fonctionnalité


Éléments :

- KO
- Non livré
- OK
- Pas commencé

- ☞ Dans le champ Nom, écrivez % OK et dans Formule saisissez = OK / (KO+ 'Non livré'+ OK+ 'Pas commencé').

- ☞ Terminez en cliquant sur Ajouter pour voir apparaître le champ dans le TCD.

Étiquettes de lignes	Nombre de	Nom
KO	16	
Non livré	3	
OK	27	
Pas commencé	14	
% OK	0,45	
Total général	60,45	

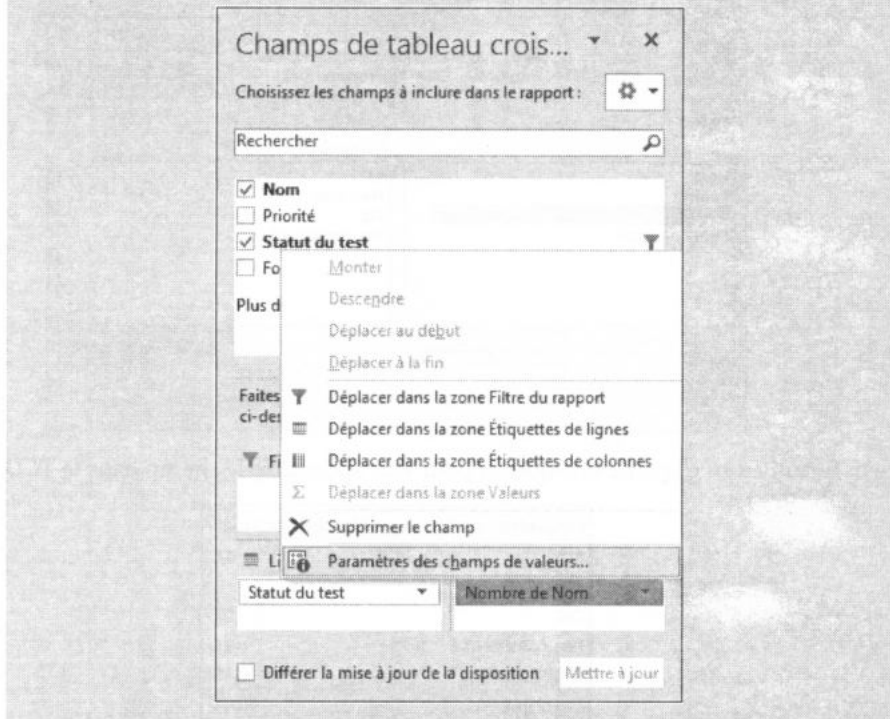
- ☞ Filtrez pour n'avoir que le champ % OK visible.
- ☞ Sélectionnez les cellules B91 et B92, cliquez sur l'icône  dans l'onglet Accueil - groupe Nombre.

Vous obtenez le résultat suivant :

90	Étiquettes de lignes	Valeur
91	% OK	45,00%
92	Total général	45,00%



Il est possible de modifier le format de la valeur affichée : faites un clic droit sur la valeur puis cliquez sur Paramètres des champs de valeur :



B. Automatisation de la création d'un rapport PowerPoint

1. Description de l'exemple

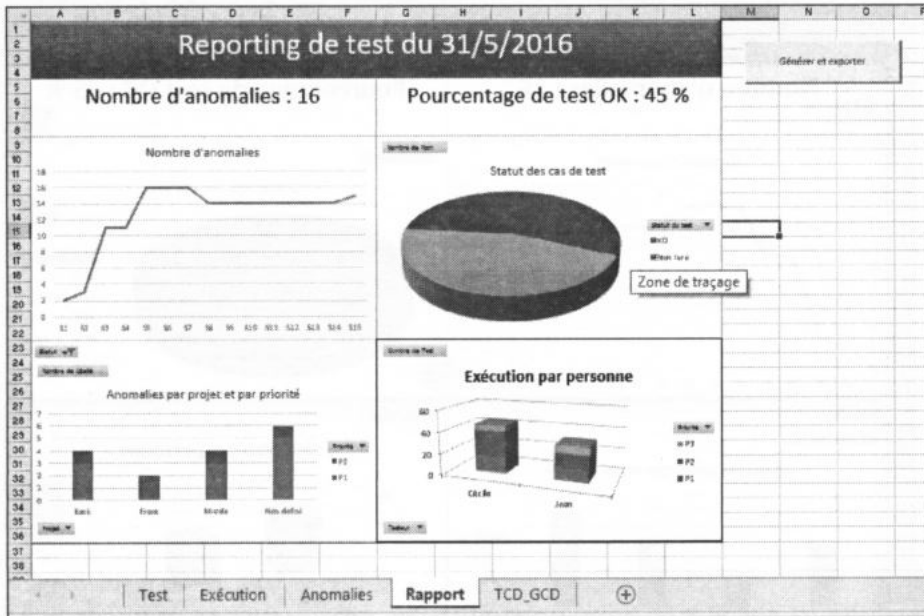
a. Présentation de l'exemple

L'objectif de cet exemple est d'automatiser la création d'un rapport PowerPoint à partir de différents graphiques et indicateurs réalisés dans le cadre de la première partie de l'exemple.

Comment faire ?

Le rapport va être mis en forme dans Excel à partir de la feuille **Rapport** : la mise en page sera réalisée au sein de la plage A1:L36 puis l'ensemble de cette plage sera copié dans un nouveau fichier PowerPoint.

Le résultat attendu est le suivant :

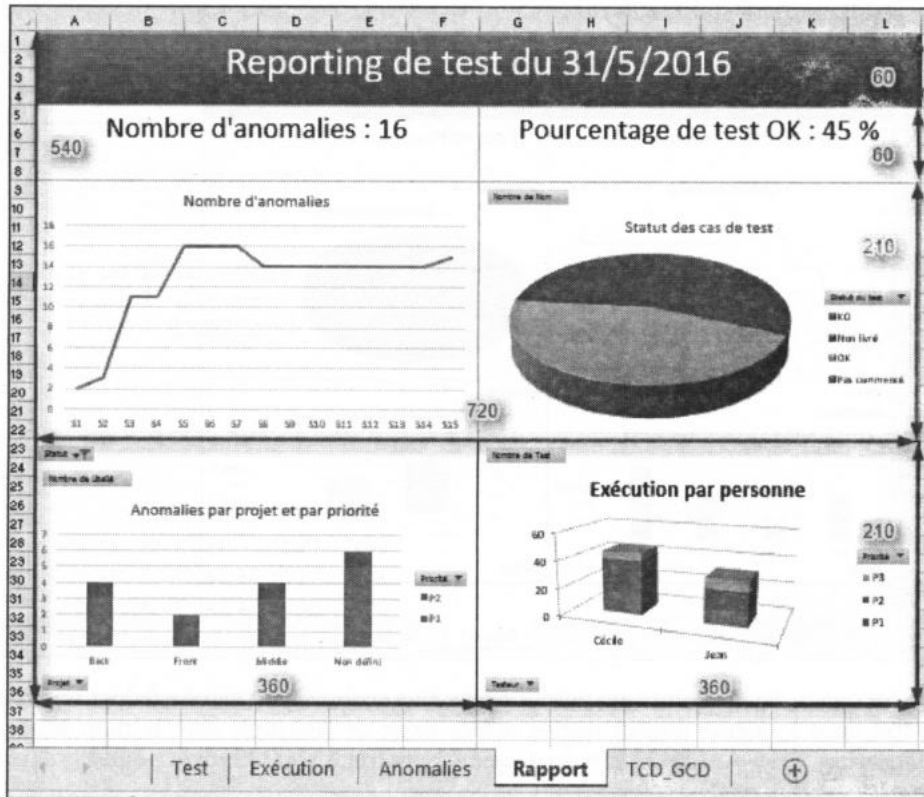


Le document PowerPoint sera présenté avec une seule diapositive. Le fichier sera enregistré dans le même dossier que le fichier Excel de l'exemple avec le nom suivant : **Reporting_Test_AAMMJJ** avec l'extension .pptx (ou AAMMJJ correspond à la date du jour).

b. Présentation du fichier

Le fichier utilisé **Enoncé_4-B.xlsm** correspond à la version obtenue à la fin de la première partie.

La feuille **Rapport** va permettre de créer le rapport au format Excel puis de le copier sur une diapositive PowerPoint. La feuille **Rapport** contient une plage prédéfinie A1:L36 qui correspond à la dimension habituelle d'une diapositive PowerPoint. La taille de ce rapport est 540 pixels en hauteur pour 720 en largeur. Le rapport sera créé via des graphiques et formes positionnés sur cette plage prédéfinie.



c. Fonctionnalités



Les fonctionnalités développées pour cet exemple sont :

- Actualiser les données des graphiques dans le cas où elles auraient été modifiées depuis la dernière génération du rapport puis les copier de la feuille TCD_GCD vers la feuille Rapport.
- Créer un nouveau graphique croisé dynamique portant sur le nombre d'exécutions de test par personne.
- Mettre en forme le titre du graphique et les indicateurs.
- Exporter le rapport Excel vers une diapositive PowerPoint.

2. Notions de cours


a. Enregistrement de macro

Cette fonctionnalité permet d'enregistrer les actions effectuées par les utilisateurs au sein d'une macro-procédure.

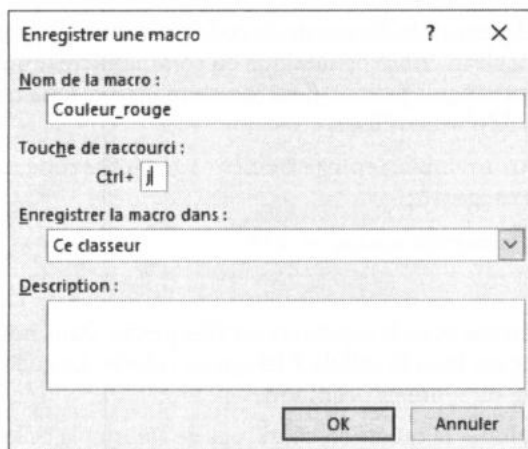
Cette fonctionnalité fonctionne comme un enregistreur, c'est-à-dire qu'à partir du moment où l'utilisateur décide d'activer l'enregistrement via le bouton  qui se situe dans la barre de statut en bas de l'application, toutes les actions sont consignées au sein d'une macro jusqu'à l'arrêt de l'enregistrement via le bouton  qui se situe également dans la barre de statut en bas de l'application.

Pour voir les avantages et inconvénients de ce type d'utilisation, nous allons faire un exemple qui permet de faire un remplissage automatique de la cellule active en rouge.

Exemple d'enregistrement de macro

Sur une feuille vierge, cliquez sur le bouton  pour démarrer l'enregistrement d'une macro. Une fenêtre s'affiche, choisissez **Couleur_rouge** en guise du nom de macro et en raccourci-clavier **Ctrl** j.

Ce raccourci-clavier signifie que lorsque l'utilisateur appuiera simultanément sur les touches **Ctrl** et j (en minuscule), la macro **Couleur_rouge** sera exécutée.



Enregistrer une macro

Nom de la macro :
Couleur_rouge


Touche de raccourci :
Ctrl+ j

Enregistrer la macro dans :
Ce classeur

Description :

OK Annuler

☞ Sélectionnez la cellule F10 et remplissez-la en rouge.

☞ Appuyez ensuite sur le bouton  pour stopper l'exécution de la macro.

- ☞ Testez la macro en remettant la cellule F10 en couleur transparente puis en appuyant simultanément sur les touches **Ctrl** j.

La macro est lancée et colorie à nouveau la cellule en cours.

- ☞ Vérifiez votre code en allant sur le **Visual Basic Editor** (touches **Alt** et **F11** en simultané).

Le code ressemble à cela :

```
Sub couleur_rouge()  
'  
' couleur_rouge Macro  
'  
' Touche de raccourci du clavier: Ctrl+j  
'  
    Range("F10").Select  
    With Selection.Interior  
        .Pattern = xlSolid  
        .PatternColorIndex = xlAutomatic  
        .Color = 255  
        .TintAndShade = 0  
        .PatternTintAndShade = 0  
    End With  
End Sub
```

Avantages

Cette fonctionnalité permet de découvrir du code sans avoir à l'apprendre. Par exemple, comment créer un tableau croisé dynamique ou comment créer un graphique. La génération du code automatique permet donc de connaître comment se structurent certaines classes ou quels objets utiliser.

Dans le cadre de notre exemple, l'enregistrement a généré ce code concernant le remplissage de la cellule en rouge :

```
Range(Cellule).Interior.Color = 255
```

Inconvénients

L'enregistrement comme vous le constatez est très précis. Dans notre exemple à chaque exécution du code, c'est bien la cellule F10 qui se colorie. Le code ne s'adapte pas à la cellule sur laquelle nous sommes positionné.

L'objectif était de colorier la cellule en cours, pas de colorier la cellule F10.

L'enregistrement génère des codes complexes alors que seul le code concernant la couleur est utile. En effet il peut se limiter à :

```
Selection.Interior.Color = 255
```

Conclusion

L'utilisation de cette fonctionnalité permet de découvrir de nouvelles possibilités, notamment pour démarrer sur un domaine peu maîtrisé. Toutefois, il ne se substitue pas à la connaissance de Visual Basic for Application puisque le code produit est « lourd » et figé.

b. Créer un tableau croisé dynamique avec VBA

Créer un TCD via VBA signifie que toutes les actions amenant à la création d'un TCD sont contenues au sein d'une macro. La création d'un TCD se décompose en trois parties :

- ▶ Création du TCD : il s'agit de la création de l'objet sôcle qui permettra de lier les données au TCD.
- ▶ Ajout de champ de valeur au TCD.
- ▶ Ajout de champ d'axe au TCD.

Création du TCD

Les données d'un TCD sont contenues dans un objet `PivotCache`. La syntaxe VBA utilisée va permettre de créer de nouvelles données en cache sur l'objet `PivotCache` grâce à la méthode `Create` puis de créer un tableau croisé dynamique à partir de ces données avec la méthode `CreatePivotTable`.

Structure du code :

```
Workbook.PivotCaches.Create(Arguments).CreatePivotTable Arguments
PivotCaches.Create
```

Cette méthode permet la création d'une source de données associée à un classeur.

Cette méthode contient deux arguments principaux :

- ▶ Type de source de données (obligatoire) : l'argument `SourceType` peut être issu d'un tableau consolidé (`xlConsolidation`), d'une plage Excel (`xlDatabase`) ou externe (`xlExternal`).
- ▶ Source de données (facultatif) : l'argument `SourceData` contient la référence à la source de données.

Par exemple :

```
ThisWorkbook.PivotCaches.Create(SourceType:=xlDatabase,
SourceData:="MesDonnees") CreatePivotTable
```

Cette méthode permet la création d'un tableau croisé dynamique à partir d'une source de données.

Elle contient trois arguments principaux :

- ▶ Zone de destination (obligatoire), cellule où sera positionné le TCD ;
- ▶ Nom du TCD (facultatif) : nom libre ;

- Type de TCD (facultatif) : différents codes existent pour déterminer l'apparence visuelle d'un TCD.

Par exemple :

```
ThisWorkbook.PivotCaches.Create(SourceType:=xlDatabase,
SourceData:="MesDonnees").CreatePivotTable
TableDestination:="Feuill1!R1C1", TableName:="NotreTCD",
DefaultVersion:= xlPivotTableVersion14
```



L'étape de création des données en cache peut être dissociée de l'étape de la création du tableau croisé dynamique.

Ajout des champs de valeur

Ajouter un champ de valeur signifie que le champ est ajouté comme donnée du TCD.

Un champ de type `DataField` va être intégré au TCD en cours grâce à la méthode `AddDataField`.

La méthode `AddDataField` aura pour argument :

- Le champ à ajouter en valeur (obligatoire) ;
- Le libellé du champ dans le TCD (facultatif) ;
- Le type d'agrégation du champ (facultatif).

```
Worksheet.PivotTables("NotreTCD").AddDataField
Worksheet.PivotTables("NotreTCD").PivotFields("ChampDonnees"),
"Compte de données", xlCount
```

Ajout des champs d'axe

Ajouter un champ d'axe signifie que l'on va ajouter un champ comme axe du TCD.

En termes techniques, il ne s'agit pas d'un ajout mais plutôt de positionner le champ dans le tableau en ligne ou en colonne. En effet, le champ (`PivotField`) est déjà présent dans la source de données du `PivotCache`.

Ajout d'une ligne :

```
Worksheet.PivotTables("NotreTCD").PivotFields("ChampAxeLigne").
Orientation = xlRowField
```

Ajout d'une colonne :

```
Worksheet.PivotTables("NotreTCD").PivotFields("ChampAxeColonne")
.Orientation = xlColumnField
```

Filtre de champ

Un champ (PivotField) peut posséder des filtres. La méthode AddFilter permet d'ajouter des filtres avec comme argument :

Type de filtre (obligatoire) : il s'agit de la condition à appliquer en tant que filtre.

Valeur (Value1 et Value2) : ces champs permettent de saisir la valeur du filtre. Par défaut, une seule valeur (value1) est utilisée, mais dans certains cas une deuxième valeur (value2) peut être utilisée : pour un filtre entre deux valeurs.

```
Worksheet.PivotTables("NotreTCD").PivotFields("ChampAxeColonne")
.AddFilter FilterType := xlCaptionEquals Value1 := "Test"
```

c. Créer un graphique avec VBA

La création d'un graphique consiste à créer une forme (Shape) de type graphique (Chart).

Il est important de dissocier les objets Shape et Chart car ils n'ont pas les mêmes méthodes et par conséquent.

La méthode Shapes.AddChart permet de créer un graphique vide.

Il est possible de renseigner sa position et son type à cette étape-là, toutefois ce n'est pas indispensable.

Il est pertinent de sélectionner avec la méthode Select l'objet nouvellement créé pour l'utiliser avec ActiveChart. ActiveChart est le graphique en cours sélectionné. C'est un objet possédant les mêmes caractéristiques qu'un objet Chart.

Une fois le graphique créé et sélectionné, il est plus simple à manipuler pour y ajouter des informations via ses propriétés.

```
Worksheet.Shapes.AddChart.Select
'Ajout de la source de données
ActiveChart.SetSourceData Source:=Range("MesDonnees")
'Graphique Histogramme Empilé
ActiveChart.ChartType = xlColumnStacked
'Changement du nom de la forme contenant le graphique (utilisation du
parent)
ActiveChart.Parent.Name = "MonGraphique"
```

Une fois le graphique créé, il est possible d'ajouter des éléments (titres, étiquettes, courbes, axes) comme par exemple, l'ajout d'un titre :

```
'Ajout du titre au graphique
ActiveChart.SetElement (msoElementChartTitleAboveChart)
ActiveChart.ChartTitle.Text = "Titre du graphique"
```



La méthode `Shapes.AddChart2` existe également, celle-ci permet de créer un graphique avec un nouvel habillage stylistique par défaut (titre par défaut).

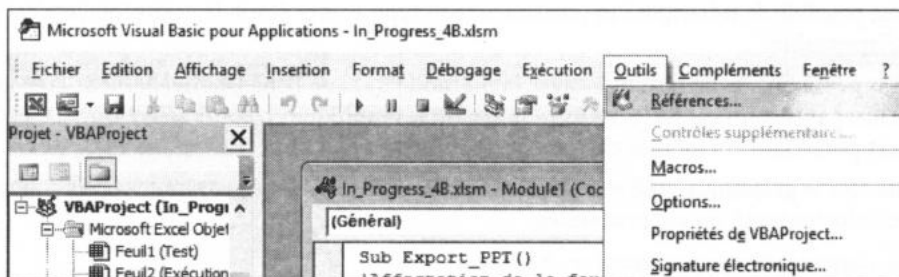
d. Manipuler PowerPoint

Ajouter la bibliothèque

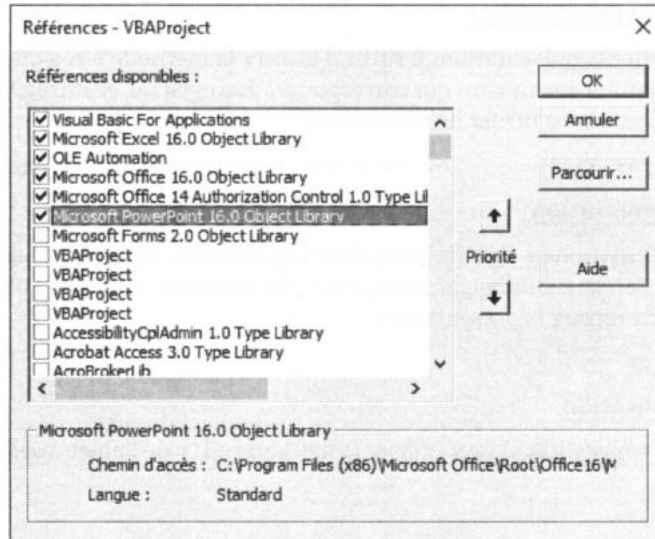
Le prérequis principal pour manipuler un objet PowerPoint au sein de VBA pour Excel est d'ajouter la bibliothèque PowerPoint au fichier Excel.

Une bibliothèque (traduction de *library*, parfois appelée librairie), est un ensemble de fonctions permettant d'ajouter des possibilités à une application. Par exemple la bibliothèque PowerPoint pour VBA permettra de manipuler PowerPoint via Excel VBA. Cela ajoutera donc des classes et méthodes à notre application Excel VBA.

- ☞ Pour ajouter cette bibliothèque, allez dans Visual Basic Editor (touches **Alt** et **F11** simultanément ou onglet **Développeur**).
- ☞ Dans le menu **Outils**, cliquez sur **Références**.



- ☞ Choisissez la référence à **Microsoft PowerPoint X.0 Object Library**.



Attention la version de la bibliothèque dépend de votre version de Microsoft Office. Une fois la bibliothèque ajoutée, il est possible de "manipuler" PowerPoint.

Manipuler l'objet application

Pour manipuler l'objet application PowerPoint, il est nécessaire de créer un objet de type PowerPoint Application.

```
Dim PptApp As Variant
Set PptApp = CreateObject("Powerpoint.Application")
```

Une fois l'objet PowerPoint Application créé, il est possible de créer une présentation et de l'ajouter à notre objet application en cours.

```
'Création de la présentation PowerPoint
Dim PptDoc As PowerPoint.Presentation
Set PptDoc = PptApp.Presentations.Add
```

Enfin, ajouter une diapositive consiste à utiliser la méthode Add de la classe Slides qui a comme parent l'objet Presentation.

```
'Ajouter une Slide
PptDoc.Slides.Add Index:=1, Layout:=ppLayoutBlank
```

L'Index correspond ici à la position et le Layout au format type de la diapositive : ici diapositive vierge et blanche.

Sauvegarder la présentation

Pour sauvegarder la présentation, il suffit d'utiliser la méthode `Save` ou `SaveAs` (`FileName`) de l'objet `Presentation` qui correspond à **Enregistrer** et **Enregistrer Sous** avec en argument le nom du fichier de destination.

```
| PptDoc.SaveAs FileName:=ThisWorkbook.Path & MonFichier.pptx"
```

Fermer la présentation

Fermer la présentation ne signifie pas quitter l'application, il s'agit juste de fermer la présentation tout en laissant l'application active. La méthode `Close` de l'objet `Presentation` permet de fermer la présentation.

```
| PptDoc.Close
```

Quitter l'application

Pour quitter l'application, il faut utiliser la méthode `Quit` de l'objet `Application`.

```
| PptApp.Quit
```

3. Réalisation de l'exemple


a. Actualiser et copier les graphiques

L'objectif de cette partie est de commencer à créer le rapport en copiant les graphiques (Nombre d'anomalies, Anomalies par projet et priorité, Statut des cas de test) créés dans la partie précédente sur la feuille `TCD_GCD`, puis en les mettant sur la feuille `Rapport`.

☞ Pour commencer, ouvrez le fichier `Enoncé_4-B.xlsm`.

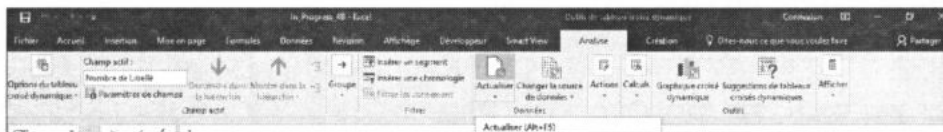
Manipulation à réaliser

Pour copier-coller les graphiques dans la feuille `Rapport`, nous allons enregistrer la manipulation réalisée sous Excel puis l'adapter pour que les graphiques soient positionnés au bon emplacement dans le rapport.

☞ Cliquez sur le bouton  pour lancer l'enregistrement des actions au sein d'une macro-procédure.

La fenêtre d'enregistrement s'affiche.

- ❏ Saisissez le nom de la macro MAJ_Copie_GCD puis cliquez sur OK.
- ❏ Sur la feuille TCD_GCD, sélectionnez le graphique Nombre d'anomalies. Copiez-le, puis collez-le sur la feuille Rapport. La position du graphique sur le rapport est approximative, mais elle sera revue par la suite.
- ❏ Sur la feuille TCD_GCD, sélectionnez le tableau croisé dynamique Anomalies par projet et par priorité.
- ❏ Cliquez sur l'onglet Outils de tableau croisé dynamique - Analyse, puis cliquez sur le bouton Actualiser comme ci-dessous.

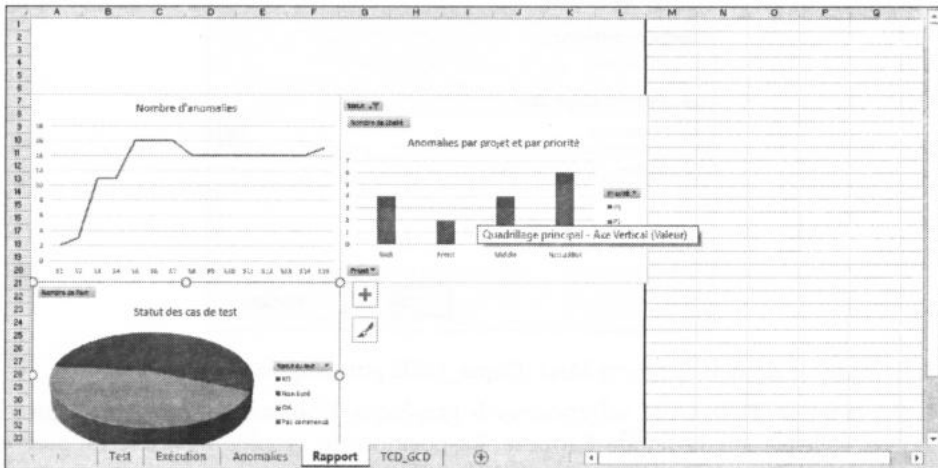



Les données sont situées dans le cache c'est-à-dire qu'elles sont stockées temporairement au sein de l'application. Cela permet d'éviter de faire de nombreux appels à la source de données et donc de gagner en temps d'affichage.

- ❏ Toujours sur la feuille TCD_GCD, sélectionnez le graphique croisé dynamique Anomalies par projet et par priorité. Copiez-le puis collez-le sur la feuille Rapport. La position du graphique sur le rapport est approximative, mais elle sera revue par la suite.
- ❏ Revenez sur la feuille TCD_GCD puis sélectionnez le tableau croisé dynamique Statut des cas de test et actualisez-le.

- ☞ Sélectionnez le graphique croisé dynamique **Statut des cas de test**. Copiez-le puis collez-le sur la feuille **Rapport**. La position du graphique sur le rapport est approximative, mais elle sera revue par la suite.

La feuille **Rapport** contient donc les trois derniers graphiques récemment actualisés.



- ☞ Cliquez sur le bouton  pour stopper l'enregistrement.

Une macro est générée MAJ_Copie_GCD dans le Module2. Chaque macro sera différente puisqu'elle est générée par les actions utilisateurs.

Par exemple, le code généré est le suivant :

```
Sub MAJ_Copie_GCD()
'
' MAJ_Copie_GCD Macro
'
'
'
ActiveSheet.ChartObjects("NbAno").Activate
ActiveChart.ChartArea.Copy
Sheets("Rapport").Select
ActiveSheet.Paste
ActiveSheet.ChartObjects("NbAno").Activate
ActiveSheet.Shapes("NbAno").IncrementLeft -1140
ActiveSheet.Shapes("NbAno").IncrementTop -148.9285826772
Sheets("TCD_GCD").Select
ActiveWindow.SmallScroll Down:=18
ActiveSheet.ChartObjects("AnoProjetPriorite").Activate
ActiveChart.PivotLayout.PivotTable.PivotCache.Refresh
ActiveChart.ChartArea.Copy
```

```

Sheets("Rapport").Select
Range("J12").Select
ActiveSheet.Paste
ActiveSheet.ChartObjects("AnoProjetPriorite").Activate
ActiveSheet.Shapes("AnoProjetPriorite").IncrementLeft -172.5
ActiveSheet.Shapes("AnoProjetPriorite").IncrementTop -60
Sheets("TCD_GCD").Select
ActiveWindow.SmallScroll Down:=21
ActiveSheet.ChartObjects("StatutCasTest").Activate
ActiveChart.ChartArea.Copy
Sheets("Rapport").Select
Range("E28").Select
ActiveSheet.Paste
ActiveSheet.ChartObjects("StatutCasTest").Activate
ActiveSheet.Shapes("StatutCasTest").IncrementLeft -236.7856692913
ActiveSheet.Shapes("StatutCasTest").IncrementTop -80.3571653543
ActiveWindow.SmallScroll Down:=3

```

End Sub

- Retirez les éléments superflus (positionnement, scroll, sélection de cellule), pour obtenir un code portant uniquement sur le copié-collé des graphiques ainsi que le rafraîchissement des données.

```

Sub MAJ_Copie_GCD()
  Sheets("TCD_GCD").Select
  ActiveSheet.ChartObjects("NbAno").Activate
  ActiveChart.ChartArea.Copy
  Sheets("Rapport").Select
  ActiveSheet.Paste
  Sheets("TCD_GCD").Select
  ActiveSheet.ChartObjects("AnoProjetPriorite").Activate
  ActiveChart.PivotLayout.PivotTable.PivotCache.Refresh
  ActiveChart.ChartArea.Copy
  Sheets("Rapport").Select
  Range("J12").Select
  ActiveSheet.Paste
  Sheets("TCD_GCD").Select
  ActiveSheet.ChartObjects("StatutCasTest").Activate
  ActiveChart.ChartArea.Copy
  Sheets("Rapport").Select
  Range("E28").Select
  ActiveSheet.Paste

```

End Sub

La taille et la position des différents graphiques sont présentés dans le tableau suivant :

Graphiques	Left (Gauche)	Top (Haut)	Height (Hauteur)	Width (Largeur)
NbAno	0	120	210	360
AnoProjetPriorite	0	330	210	360
StatutCasTest	360	120	210	360

- ☞ Utilisez les propriétés **Left**, **Top**, **Height** et **Width** de l'objet **Shape** pour définir la taille et la position des graphiques. Saisissez les lignes suivantes un peu avant la fin de la procédure pour procéder à la mise en forme des graphiques sur la feuille **Rapport**.

```
ActiveSheet.Shapes("NbAno").Left = 0
ActiveSheet.Shapes("NbAno").Top = 120
ActiveSheet.Shapes("NbAno").Height = 210
ActiveSheet.Shapes("NbAno").Width = 360

ActiveSheet.Shapes("AnoProjetPriorite").Left = 0
ActiveSheet.Shapes("AnoProjetPriorite").Top = 330
ActiveSheet.Shapes("AnoProjetPriorite").Height = 210
ActiveSheet.Shapes("AnoProjetPriorite").Width = 360

ActiveSheet.Shapes("StatutCasTest").Left = 360
ActiveSheet.Shapes("StatutCasTest").Top = 120
ActiveSheet.Shapes("StatutCasTest").Height = 210
ActiveSheet.Shapes("StatutCasTest").Width = 360
```

b. Nombre de tests par personne

L'objectif de cette procédure est de créer un tableau croisé dynamique permettant de connaître le nombre de cas de test exécutés par les deux testeurs. Cela permettra de créer un graphique qui sera positionné sur le rapport à exporter.

Voici les différentes étapes à créer dans la procédure `Créer_TCD` :

- ▶ Dans le cas où vous avez déjà lancé la procédure, il faut pouvoir supprimer le précédent TCD ;
- ▶ Créer le TCD ;
- ▶ Ajouter un champ de données ;
- ▶ Ajouter les axes ;
- ▶ Créer le graphique ;
- ▶ Positionner le graphique sur la feuille **Rapport**.

Voici la manipulation à réaliser :

- ❖ Créez la procédure `Creer_TCD`.

```
Sub Creer_TCD()
End Sub
```

- ❖ Supprimez l'ancien TCD qui était positionné sur la plage `A100:E104` dans la feuille `TCD_GCD`.

```
'Supprimer l'ancien TCD
Sheets("TCD_GCD").Activate
Range("A100:E104").Select
Selection.Clear
```

- ❖ Créez les données en cache avec la méthode `PivotCaches.Create` qui a pour argument la source de données de la plage `Exec`. À partir de ce cache, créez le TCD. La plage de destination est la cellule `A100` (exprimée dans ce contexte par sa dénomination `R100C1` pour ligne (Row) 100, colonne (Column) 1) de la feuille `TCD_GCD`.

```
'Création du TCD
ActiveWorkbook.PivotCaches.Create(SourceType:=xlDatabase,
SourceData:= "Exec", Version:=6).CreatePivotTable
TableDestination:="TCD_GCD!R100C1", _
TableName:="TCD_ExecParPersonne", DefaultVersion:=6
```

- ❖ Ajoutez le champ `Test` en tant que valeur du TCD `TCD_ExecParPersonne`.

```
'Ajouter le champ Test en Valeurs
ActiveSheet.PivotTables("TCD_ExecParPersonne").AddDataField
ActiveSheet.PivotTables("TCD_ExecParPersonne").PivotFields("Test"),
"Nombre de Test", xlCount
```

- ❖ Positionnez le champ `Testeur` en ligne, le champ `Priorité` en colonne.

```
'Ajouter le champ Testeur en Lignes
With
ActiveSheet.PivotTables("TCD_ExecParPersonne").PivotFields("Testeur")
.Orientation = xlRowField
.Position = 1
End With
'Ajouter le champ Priorité en Colonnes
With
ActiveSheet.PivotTables("TCD_ExecParPersonne").PivotFields("Priorité")
.Orientation = xlColumnField
.Position = 1
End With
```

- ☞ Accédez à la feuille **Rapport** puis créez un graphique.

```
'Sélection de la feuille Rapport
Sheets("Rapport").Activate
'Création puis sélection du graphique
ActiveSheet.Shapes.AddChart.Select
```

- ☞ Ajoutez en guise de source de données le TCD TCD_ExecParPersonne.

```
'Ajout de la source de données correspondant au TCD
TCD_ExecParPersonne
ActiveChart.SetSourceData Source:=Range("TCD_GCD!$A$100:$E$104")
```

- ☞ Déterminez le style du graphique avec un histogramme en barres empilées 3D.

```
'Graphique Histogramme 3D empilé
ActiveChart.ChartType = xl3DColumnStacked
```

Dans les graphiques imbriqués, le nom de l'objet est porté par la forme parente qui englobe le graphique.

- ☞ Donnez un nom à la forme parente du graphique.

```
'Changement du nom de la forme parente du graphique
ActiveChart.Parent.Name = "GCD_ExecParPersonne"
```

- ☞ Ajoutez le titre **Exécution par personne**.

```
'Ajout du titre au graphique
ActiveChart.SetElement (msoElementChartTitleAboveChart)
ActiveChart.ChartTitle.Text = "Exécution par personne"
```

- ☞ Remplissez les axes et le contour sur fond noir en mettant leur propriété couleur à RGB(0,0,0).



Chaque couleur est définie par une combinaison d'une échelle allant de 0 à 255 de Rouge (Red), Vert (Green) et Bleu (Blue). RGB (0,0,0) correspond au noir, RGB (255,255,255) correspond au blanc.

```
'Mise à jour de la couleur de l'axe
ActiveChart.Axes(xlCategory).TickLabels.Font.Color = RGB(0, 0, 0)
'Ajout d'un contour et mise à jour de sa couleur
With ActiveSheet.Shapes("GCD_ExecParPersonne").Line
.Visible = msoTrue
.ForeColor.RGB = RGB(0, 0, 0)
.Transparency = 0
End With
```

- ☞ Terminez en positionnant le graphique sur le rapport : gauche 360, haut 330, et définissant sa taille : largeur 360, hauteur 210.

```
'Définition de la position et de la taille du graphique
ActiveSheet.Shapes("GCD_ExecParPersonne").Top = 330
ActiveSheet.Shapes("GCD_ExecParPersonne").Left = 360
ActiveSheet.Shapes("GCD_ExecParPersonne").Height = 210
ActiveSheet.Shapes("GCD_ExecParPersonne").Width = 360
```

Le résultat de la procédure est le suivant :

```
Sub Creer_TCD()
'Supprimer l'ancien TCD
Sheets("TCD_GCD").Activate
Range("A100:E104").Select
Selection.Clear
'Création du TCD
ActiveWorkbook.PivotCaches.Create(SourceType:=xlDatabase,
SourceData:="Exec", Version:=6).CreatePivotTable
TableDestination:="TCD_GCD!R100C1", _
TableName:="TCD_ExecParPersonne", DefaultVersion:=6
'Ajouter le champ Test en Valeurs
ActiveSheet.PivotTables("TCD_ExecParPersonne").AddDataField
ActiveSheet.PivotTables("TCD_ExecParPersonne").PivotFields("Test"),
"Nombre de Test" _
, xlCount
'Ajouter le champ Testeur en Lignes
With
ActiveSheet.PivotTables("TCD_ExecParPersonne").PivotFields("Testeur")
.Orientation = xlRowField
.Position = 1
End With
'Ajouter le champ Priorité en Colonnes
With
ActiveSheet.PivotTables("TCD_ExecParPersonne").PivotFields("Priorité")
.Orientation = xlColumnField
.Position = 1
End With
'Sélection de la feuille Rapport
Sheets("Rapport").Activate
'Création puis sélection du graphique
ActiveSheet.Shapes.AddChart.Select
'Ajout de la source de données correspondant au TCD TCD_ExecParPersonne
ActiveChart.SetSourceData Source:=Range("TCD_GCD!$A$100:$E$104")
'Graphique Histogramme 3D Empilé
ActiveChart.ChartType = xl3DColumnStacked
'Changement du nom de la forme parente du graphique
ActiveChart.Parent.Name = "GCD_ExecParPersonne"
'Ajout du titre au graphique
ActiveChart.SetElement (msoElementChartTitleAboveChart)
ActiveChart.ChartTitle.Text = "Exécution par personne"
'Mise à jour de la couleur de l'axe
```

```

ActiveChart.Axes(xlCategory).TickLabels.Font.Color = RGB(0, 0, 0)
'Ajout d'un contour et mise à jour de sa couleur
With ActiveSheet.Shapes("GCD_ExecParPersonne").Line
.Visible = msoTrue
.ForeColor.RGB = RGB(0, 0, 0)
.Transparency = 0
End With
'Définition de la position et de la taille du graphique
ActiveSheet.Shapes("GCD_ExecParPersonne").Top = 330
ActiveSheet.Shapes("GCD_ExecParPersonne").Left = 360
ActiveSheet.Shapes("GCD_ExecParPersonne").Height = 210
ActiveSheet.Shapes("GCD_ExecParPersonne").Width = 360
End Sub

```

c. Mise en forme du rapport

L'objectif de cette mise en forme est de créer une procédure permettant d'ajouter les zones de texte sur le rapport. Les zones de texte sont le titre du graphique contenant la date du jour, le nombre d'anomalies en cours et le pourcentage de tests OK.


Comment procéder ?

Nous allons utiliser l'enregistrement de macro pour générer le code associé à la création d'une forme (shape) de type zone de texte (Text Box) avec un style prédéfini.

Une fois le code généré, il est possible de l'adapter pour générer les trois zones de texte.

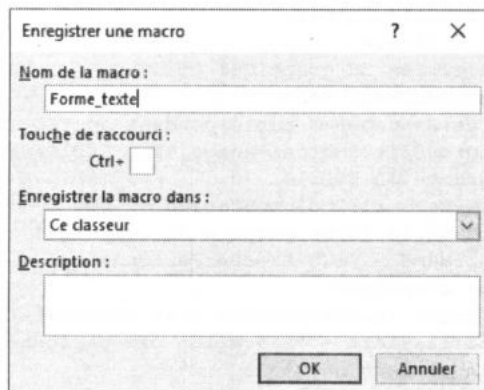
Enregistrer la création d'une zone de texte

Pour commencer, il convient d'enregistrer la création de la zone de texte avec un style prédéfini bleu accentué.

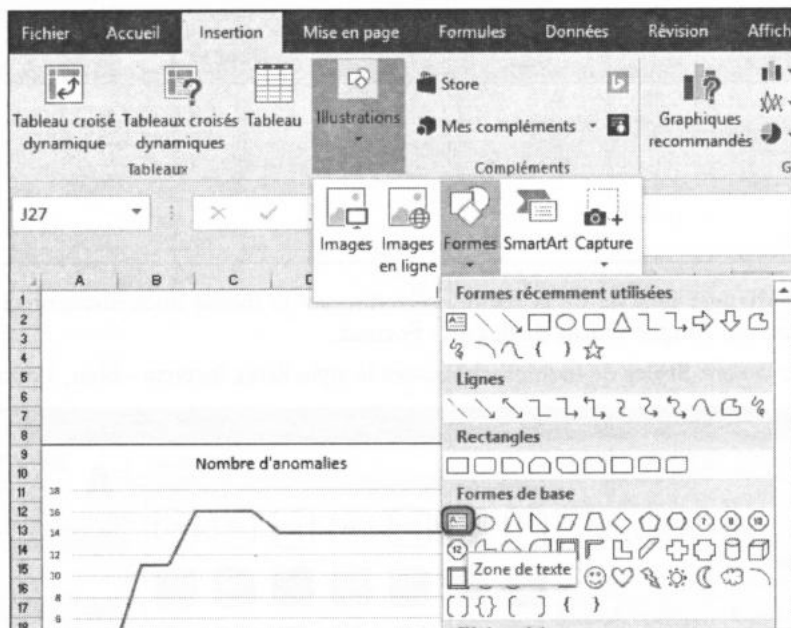
❖ Démarrez l'enregistrement en cliquant sur le bouton .

La fenêtre d'enregistrement de la macro apparaît.

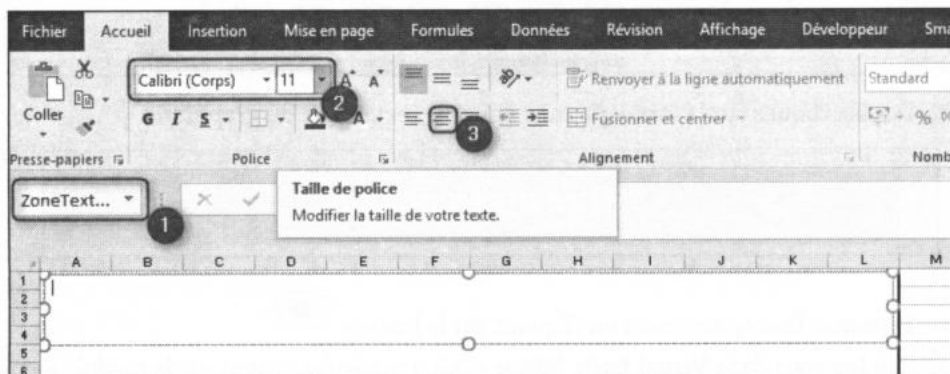
❖ Saisissez `Forme_texte` pour le Nom de la macro.



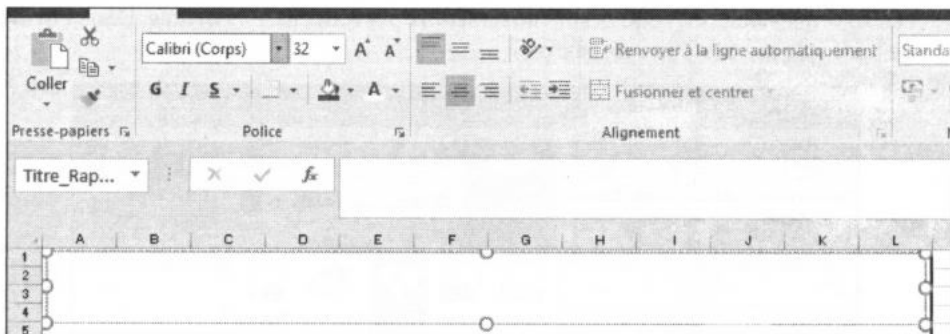
- ❖ Après avoir cliqué sur le bouton OK, positionnez-vous ensuite sur la feuille **Rapport**.
- ❖ Dans l'onglet **Insérer**, cliquez sur **Illustrations** puis choisissez **Formes**. Dans **Formes de base**, cliquez sur **Zone de texte**.



- ❖ Dessinez la forme approximativement sur la zone de rapport en titre (voir schéma). La taille et la position de la forme seront reprises dans le code généré.
- ❖ Modifiez dans un premier temps le nom du graphique (1) en **Titre_Rapport**, puis la taille de la police (2) en 32, puis modifiez l'alignement du texte (3) en **Centrer**.



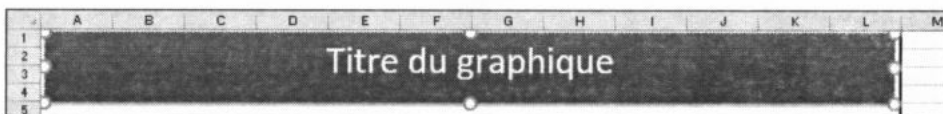
Voici le résultat :




- ❏ Pour changer le style de la forme, sélectionnez la forme nouvellement créée puis cliquez sur l'onglet Outils de dessin - Format.
- ❏ Dans la zone Styles de formes, choisissez le style Effet intense - Bleu, 1 accentué.



- ❏ Double cliquez sur la forme pour en éditer le texte Titre du graphique.



- ❏ Terminez l'enregistrement en cliquant sur le bouton .
- ❏ Rendez-vous dans Visual Basic Editor et plus particulièrement sur le module contenant la procédure `Forme_Texte`.

- ☞ Supprimez le code superflu pour obtenir une procédure contenant les informations suivantes :

```
Sub Forme_Texte()
    'Création de la zone de texte : les informations liées à la
    position ne sont pas importantes pour le moment.
    'L'instruction Select permet de sélectionner la textbox créée.

    ActiveSheet.Shapes.AddTextbox(msoTextOrientationHorizontal,
    4.2857480315, _
        3.2142519685, 710.3570866142, 57.8571653543).Select
    'Attribution du nom du graphique
    Selection.ShapeRange(1).Name = "Titre_Rapport"
    'Changement de la taille du contenu
    Selection.ShapeRange(1).TextFrame2.TextRange.Font.Size = 32
    'Alignement au centre
    Selection.ShapeRange(1).TextFrame2.TextRange.ParagraphFormat.
    Alignment = _            msoAlignCenter
    'Application du style Effet Intense - 1 Bleu accentué
    Selection.ShapeRange(1).ShapeStyle = msoShapeStylePreset37
    'Modification du contenu texte
    Selection.ShapeRange(1).TextFrame2.TextRange.Characters.Text = _
        "Titre du graphique"
    'Application de la couleur blanche pour le texte
    Selection.ShapeRange(1).TextFrame2.TextRange.Characters(1,
    18).Font.Fill.ForeColor.ObjectThemeColor = msoThemeColorLight1
End Sub
```

Analyser le code généré

Le code généré ferait bondir de nombreux programmeurs ayant une vision rigoureuse de la programmation car ce morceau de code n'est pas très "propre"... Mais pour vous, l'objectif est d'apprendre et l'enregistrement est un très bon moyen.

- ▶ Tout d'abord `AddTextbox` est une méthode `Worksheet.Shapes` qui permet de créer une forme de type Zone de texte (`TextBox`). Cette méthode nécessite d'avoir l'orientation, la position et la taille.

```
AddTextbox(Orientation, Left, Top, Width, Height)
```

- ▶ L'objet `ShapeRange` est compliqué à utiliser car il s'agit d'une plage de forme. L'instruction `Selection.ShapeRange` dans notre contexte correspond à notre zone de texte puisque seule notre zone de texte est dans la sélection, s'il y a plusieurs shapes dans la sélection, cela peut porter à confusion dans le code. Il semble donc plus approprié d'utiliser l'objet `Shapes` qui a comme objet parent `Worksheet`. Par conséquent `Worksheet.Shapes("NomForme")` permet d'identifier précisément la forme souhaitée.
- ▶ Le style prédéfini est contenu dans la propriété `ShapeStyle` de l'objet `Shape`.

- Pour éditer le style du texte, il faut utiliser les propriétés contenues au sein de `Shapes.TextFrame2.TextRange`

Après avoir épuré le code, le voici opérationnel pour la génération du titre du rapport avec la date :

```
Sub Forme_Texte()
'Selection de la feuille Rapport
Worksheets("Rapport").Activate
'Création de la zone de texte
'Position gauche=0, haut=0, taille largeur : 720 hauteur : 60
ActiveSheet.Shapes.AddTextbox(msoTextOrientationHorizontal, 0, _
    0, 720, 60).Select
'Attribution du nom du graphique
Selection.ShapeRange.Name = "Titre_Rapport"
'Changement de la taille du contenu
ActiveSheet.Shapes("Titre_Rapport").TextFrame2.TextRange.Font.Size = 32
'Alignement au centre
ActiveSheet.Shapes("Titre_Rapport").TextFrame2.TextRange.ParagraphFormat.
    Alignment = msoAlignCenter
'Modification du contenu texte avec information de la date
ActiveSheet.Shapes("Titre_Rapport").TextFrame2.TextRange.Characters.Text
    = "Reporting de test du " & Day(Now) & "/" & Month(Now) & "/" & Year(Now)
'Application du style Bleu accentué
ActiveSheet.Shapes("Titre_Rapport").ShapeStyle = msoShapeStylePreset37
'Application de la couleur blanche pour le texte
ActiveSheet.Shapes("Titre_Rapport").TextFrame2.TextRange.Characters.
    Font.Fill.ForeColor.ObjectThemeColor = msoThemeColorLight1
End Sub
```

Ajout des zones de texte pour le nombre d'anomalies et le pourcentage de tests OK

Maintenant que le code est en place pour le titre, il suffit de reprendre le même code avec quelques adaptations.

- ☞ Copiez-collez le code créé pour le titre du rapport puis faites les modifications suivantes pour créer la zone de texte affichant le nombre d'anomalies :
 - La taille de la police sera de 24 au lieu de 32 ;
 - La position sera : gauche 0, haut 60 (en dessous du titre) et la taille sera largeur 360, hauteur 60 ;
 - Le nombre d'anomalies récupère la valeur de cellule A81 de la feuille TCD_GCD ;
 - Pas de style appliqué.

Le code obtenu est le suivant :

```
'Création de la zone de texte
ActiveSheet.Shapes.AddTextbox(msoTextOrientationHorizontal, 0, _
    60, 360, 60).Select
'Attribution du nom du graphique
```

```

Selection.ShapeRange.Name = "Nb_Ano"
'Changement de la taille du contenu
ActiveSheet.Shapes("Nb_Ano").TextFrame2.TextRange.Font.Size = 24
'Alignement au centre
ActiveSheet.Shapes("Nb_Ano").TextFrame2.TextRange.ParagraphFormat.
Alignment = msoAlignCenter
'Modification du contenu texte avec la cellule A81
ActiveSheet.Shapes("Nb_Ano").TextFrame2.TextRange.Characters.Text =
"Nombre d'anomalies : " & Sheets("TCD_GCD").Cells(81, 1).Value

```

☞ Procédez de même pour la zone de texte affichant le nombre de test réussis :

- ▶ La taille de la police sera de 24 au lieu de 32 ;
- ▶ La position sera : gauche 360 (à côté du nombre d'anomalies, haut 60 (en dessous du titre) et la taille sera largeur 360, hauteur 60 ;
- ▶ Le pourcentage de test OK récupère la valeur de cellule B91 de la feuille TCD_GCD qui contient le pourcentage de test OK qu'il faudra multiplier par 100 et ajouter le signe %.

Vous obtenez le code suivant :

```

'Code après la création de zone de texte correspondant au nombre
d'anomalie
'Création de la zone de texte
ActiveSheet.Shapes.AddTextbox(msoTextOrientationHorizontal, 360, _
    60, 360, 60).Select
'Attribution du nom du graphique
Selection.ShapeRange.Name = "TestOK"
'Changement de la taille du contenu
ActiveSheet.Shapes("TestOK").TextFrame2.TextRange.Font.Size = 24
'Alignement au centre
ActiveSheet.Shapes("TestOK").TextFrame2.TextRange.ParagraphFormat.
Alignment = msoAlignCenter
'Modification du contenu texte avec la cellule B91
ActiveSheet.Shapes("TestOK").TextFrame2.TextRange.Characters.Text =
"Pourcentage de test OK : " & Sheets("TCD_GCD").Cells(91, 2).Value
* 100 & " %"
Fin de la procédure

```

d. Création du rapport PowerPoint

L'objectif de cette procédure est de créer une présentation PowerPoint et de copier l'ensemble des éléments du rapport construit sur la feuille Excel Rapport puis de les coller sur la présentation PowerPoint.

☞ Créez une procédure **Export_PPT**.

```

Sub Export_PPT()
End Sub

```

- ☞ Au sein de cette procédure, créez une variable de type `Worksheet` qui contient la feuille **Rapport**. Cela permettra de faciliter la copie des formes (`Shapes`) vers la présentation PowerPoint.

```
'Affectation de la feuille à la variable WS
Dim WS As Excel.Worksheet
Set WS = ThisWorkbook.Sheets("Rapport")
```

- ☞ Créez ensuite l'application PowerPoint puis une nouvelle présentation.

```
'Création d'un objet application PowerPoint
Dim PptApp As Variant
Set PptApp = CreateObject("Powerpoint.Application")
'Création de la présentation PowerPoint
Dim PptDoc As PowerPoint.Presentation
Set PptDoc = PptApp.Presentations.Add
```

- ☞ Assurez-vous que la présentation est au format 4/3 car le rapport défini dans la feuille Excel a été réalisé au format 4/3. Pour cela, utilisez la propriété `SlideSize` de `PageSetup`.

```
'Format de la présentation en mode 4/3
PptDoc.PageSetup.SlideSize = ppSlideSizeOnScreen
```

- ☞ Positionnez un bloc `With PptDoc ... End With` afin de ne pas réécrire de nombreuses fois ce bout de code.

```
With PptDoc
End With
```

- ☞ Créez une diapositive vierge dans la présentation.

```
'--- Ajout d'une Slide
.Slides.Add Index:=1, Layout:=ppLayoutBlank
```

- ☞ Utilisez des variables `Haut`, `Gauche`, `Largeur`, `Hauteur` de type nombre décimal pour stocker les positions et tailles des différentes formes (`Shapes`) à copier dans la diapositive PowerPoint.

```
Dim Haut, Gauche, Largeur, Hauteur As Double
```

- ☞ Faites une boucle de type `For ... Next` pour parcourir l'ensemble des formes contenues sur la feuille `WS` qui correspond à la feuille **Rapport**. Il sera possible ensuite d'identifier chaque forme avec son numéro d'index `WS.Shapes(Index)` où `Index` est l'élément variable de la boucle.

```
For I = 1 To WS.Shapes.Count
Next
```

- ☞ Au sein de la boucle, affectez les valeurs aux variables `Haut` et `Gauche`.

```
Haut = WS.Shapes(I).Top
Gauche = WS.Shapes(I).Left
```

Seules les formes ayant une hauteur inférieure à 540 et une valeur gauche inférieure à 720 doivent être copiées. Les autres sont en dehors du rapport.

Par exemple les boutons qui seront situés à la droite du rapport sont considérés comme des formes mais ne doivent pas être copiés dans la diapositive.

```
If Haut < 540 And Gauche < 720 Then
End If
```

À l'intérieur de l'instruction conditionnelle `If` :

- ❏ Affectez les valeurs aux variables **Largeur** et **Longueur**.
- ❏ Copiez la forme de la feuille **Rapport** du fichier Excel.
- ❏ Collez-la sur la diapositive **PowerPoint**.

La diapositive est identifiée avec l'index 1 puisqu'elle est seule sur la présentation. La forme sera identifiée sur la diapositive avec son index qui correspond au nombre de formes sur la diapositive.

- ❏ Pour éviter tout problème dans la mise en forme, il est recommandé de copier-coller les éléments en mode Image avec un collage spécial (`PasteSpecial`) avec l'argument `ppPasteMetafilePicture` qui permet de les transformer en image.
- ❏ Pour éviter également de bloquer la proportionnalité de la forme, attribuez la valeur `msoFalse` à la propriété `LockAspectRatio`.
- ❏ Modifiez la taille et la position de la forme nouvellement créée dans la diapositive **PowerPoint**.
- ❏ Fermez l'instruction conditionnelle et la boucle.

```
For I = 1 To WS.Shapes.Count
Haut = WS.Shapes(I).Top
Gauche = WS.Shapes(I).Left
If Haut < 540 And Gauche < 720 Then
    Largeur = WS.Shapes(I).Width
    Hauteur = WS.Shapes(I).Height
    WS.Shapes(I).Copy
    .Slides(1).Shapes.PasteSpecial (ppPasteMetafilePicture)
    .Slides(1).Shapes(.Slides(1).Shapes.Count).LockAspectRatio =
msoFalse
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Top = Haut
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Left = Gauche
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Width = Largeur
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Height = Hauteur
End If
Next
```

La procédure est la suivante :

```

Sub Export_PPT()
'Affectation de la feuille à la variable WS
Dim WS As Excel.Worksheet
Set WS = ThisWorkbook.Sheets("Rapport")
'Création d'un objet application PowerPoint
Dim PptApp As Variant
Set PptApp = CreateObject("Powerpoint.Application")
'Création de la présentation PowerPoint
Dim PptDoc As PowerPoint.Presentation
Set PptDoc = PptApp.Presentations.Add
'Format de la présentation en mode 4:3
PptDoc.PageSetup.SlideSize = ppSlideSizeOnScreen

With PptDoc
'--- Ajouter une slide
.Slides.Add Index:=1, Layout:=ppLayoutBlank
Dim Haut, Gauche, Largeur, Hauteur As Double
For I = 1 To WS.Shapes.Count
Haut = WS.Shapes(I).Top
Gauche = WS.Shapes(I).Left
If Haut < 540 And Gauche < 720 Then
    Largeur = WS.Shapes(I).Width
    Hauteur = WS.Shapes(I).Height
    WS.Shapes(I).Copy
    .Slides(1).Shapes.PasteSpecial (ppPasteMetafilePicture)
    .Slides(1).Shapes(.Slides(1).Shapes.Count).LockAspectRatio =
msoFalse
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Top = Haut
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Left = Gauche
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Width = Largeur
    .Slides(1).Shapes(.Slides(1).Shapes.Count).Height = Hauteur
End If
Next

.SaveAs Filename:=ThisWorkbook.Path & "\" & "Reporting_Test_"
& Year(Now) & Month(Now) & Day(Now) & ".pptx"
'fermeture de la présentation
.Close
End With
'fermeture du PowerPoint
PptApp.Quit
End Sub

```


e. Finalisation

L'objectif de cette dernière procédure est d'exécuter l'ensemble des actions décrites jusque-là et de l'associer à un bouton à créer sur la feuille **Rapport**.

Création d'une procédure pour exécuter l'ensemble des procédures créées

Cette procédure doit :

- ▶ Supprimer tous les éléments du rapport actuel ;
- ▶ Lancer les différentes procédures.

Commencez par créer la procédure :

```
Sub ProcedureComplete
End Sub
```

Pour supprimer l'ensemble des formes du rapport, il faut faire une boucle sur l'ensemble des formes de la feuille **Rapport**. Toutefois, il faut veiller à ne pas supprimer les formes en dehors de la zone du rapport.

Pour cela, au sein de la boucle il faut créer une instruction conditionnelle If permettant de ne pas supprimer les formes dont la hauteur est supérieure à 540 en hauteur ou 720 en largeur.

Pour supprimer la forme en cours, utilisez la méthode Delete.

```
Sheets("Rapport").Select
For Each SH In ActiveSheet.Shapes
If SH.Top < 540 And SH.Left < 720 then
SH.Delete
End If
Next
```

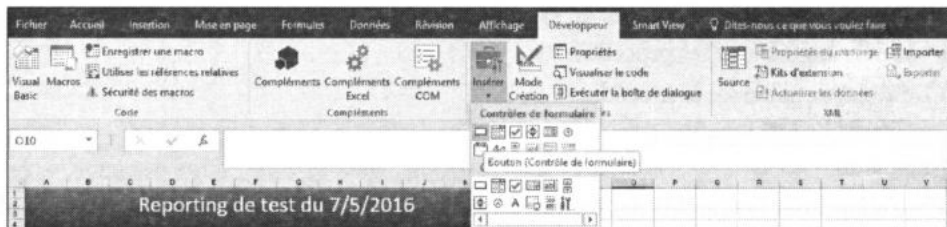
- ☞ Faites ensuite appel aux différentes procédures avec l'instruction Call :
 - ▶ Procédure pour actualiser et copier-coller les graphiques ;
 - ▶ Procédure pour créer le TCD sur le nombre de tests exécutés par testeur ;
 - ▶ Procédure pour créer les formes de texte ;
 - ▶ Procédure pour exporter les formes sous PowerPoint.

```
Call MAJ_Copie_GCD
Call Creer_TCD
Call Forme_Texte
Call Export_PPT
MsgBox "Opération terminée."
```

Création du bouton d'action

L'objectif est de créer un bouton déclenchant la procédure `ProcedureComplete`.

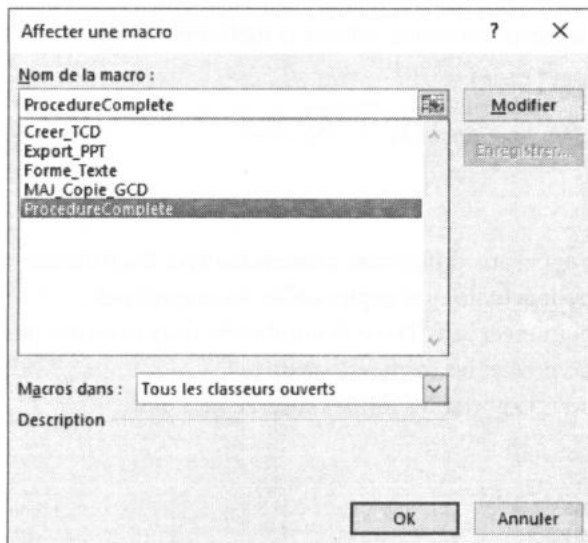
- ☞ Positionnez-vous sur la feuille **Rapport**.
- ☞ Dans l'onglet **Développeur**, cliquez sur **Insérer** puis sur **Bouton** dans **Contrôles de formulaire**.



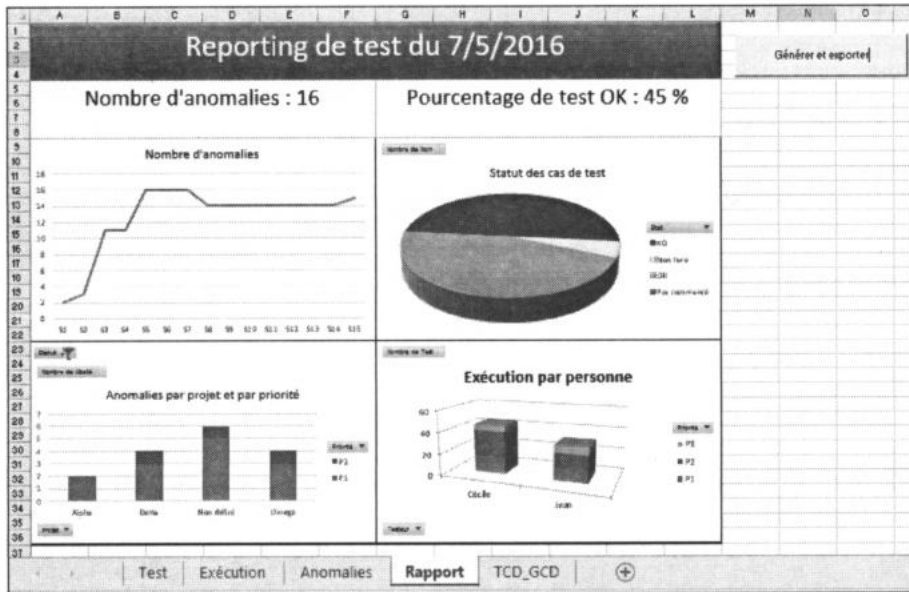
- ☞ Positionnez le bouton à la droite du rapport.

Une fois que le bouton est créé, la fenêtre d'affectation de la macro apparaît.

- ☞ Affectez la macro `ProcedureComplete` au bouton créé.



☞ Terminez en cliquant sur OK puis renommez le bouton Générer et exporter.



Chapitre 5

Gestion des employés

A. Calcul de la durée et du planning.....	233
B. Gestion des présences - Outil d'administration.....	254

A. Calcul de la durée et du planning

1. Description de l'exemple

a. Présentation de l'exemple

L'objectif de cet exemple est de réaliser le planning du projet informatique de portail client de la société **SacEni** avec les ressources en main d'œuvre à disposition. Un ensemble de tâches doit être réalisé pour délivrer le projet.

Les ressources

Chaque ressource a un poste : manager, concepteur, développeur ou testeur.

Chaque ressource a indiqué sa disponibilité sur la période allant du 01/04/2016 au 30/06/2016.

Les tâches

Chaque tâche est considérée comme exécutée lorsque le nombre de jours par poste a été consommé. Elles débutent lorsque les tâches précédentes ont été réalisées et se terminent lorsque les ressources nécessaires ont été affectées.

Objectif

L'objectif est de déterminer la date au plus tôt pour finir le projet en fonction de la disponibilité des ressources. Il sera demandé également d'avoir une représentation graphique de cette date de fin.

b. Présentation du fichier

Le fichier **Enoncé_5-A.xlsm** se compose de deux feuilles : **Planning** et **Taches**.

Feuille Planning

La feuille Planning contient la liste des ressources participant au projet. Chaque ressource a son poste (colonne B) et indique ses disponibilités pour chaque jour en indiquant une croix (X) dans la cellule correspondant à la date (à partir de la colonne C) :

	A	B	C	D	E	F	G	H	I
			01/04/2016	02/04/2016	03/04/2016	04/04/2016	05/04/2016	06/04/2016	07/04/2016
3	Ressources	Poste							
4	Ressource 1	Manager	X			X	X	X	X
5	Ressource 2	Manager							
6	Ressource 3	Concepteur						X	X
7	Ressource 4	Concepteur				X			
8	Ressource 5	Concepteur							
9	Ressource 6	Développeur				X			
10	Ressource 7	Développeur							

La plage de cellules des disponibilités des ressources C4:CO26 se nomme **Planning**.

La plage de cellules des jours fériés située en CQ4:CQ6 se nomme **Férier**.

Feuille Taches

La feuille Taches contient la liste des tâches à réaliser avec :

- ▶ Le nombre de jours par poste requis (colonnes B à E) ;
- ▶ Les tâches antérieures à réaliser avant de commencer la tâche en question (colonne F) ;
- ▶ La date de début (colonne G) ;
- ▶ La durée (colonne H) ;
- ▶ La date de fin (colonne I) ;
- ▶ Le planning sous forme de diagramme de Gantt (à partir de la colonne J).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
3		Manager	Concepteur	Développeur	Testeur	Tache antérieure	Date de début minimal	Durée	Date de fin		01/04/2016	02/04/2016	03/04/2016	04/04/2016	05/04/2016
4	Tache 1	4	12	6	0		01/04/2016		01/04/2016						
5	Tache 2	6	13	4	1	Tache 1									
6	Tache 3	4	2	20	5	Tache 1									
7	Tache 4	12	10	36	12	Tache 3									
8	Tache 5	6	2	31	20	Tache 4									
9	Tache 6	3	3	21	21	Tache 4									
10	Tache 7	3	0	10	24	Tache 6, Tache 5									
11	Tache 8	2	0	8	21	Tache 7									
12															

La zone du diagramme de Gantt J4:CV11 se nomme **Tâches**.

Visual Basic

Dans la partie Visual Basic du projet, un module a été créé et contient une fonction permettant de calculer si le jour fait partie de la liste des jours fériés :

```
EstFerie(date)
```

c. Fonctionnalités

Les fonctionnalités proposées dans cet exemple sont les suivantes :

- ▶ Calculer la durée de chaque tâche ;
- ▶ Établir un diagramme de Gantt.

2. Notions de cours

a. Formules de date

Les formules de date permettent de faire des opérations sur les dates : ajouter/compter des jours, trouver la fin de mois...

Calculer le nombre de jours ouvrés entre deux dates

La fonction NB.JOURS.OUVRES permet de calculer le nombre de jours ouvrés entre deux dates.

La syntaxe est la suivante :

```
=NB.JOURS.OUVRES(date_debut ; date_fin ; [jours_feries])
```

- ▶ Date_debut : correspond à la date de début de la série.
- ▶ Date_Fin : correspond à la date de fin de la série.
- ▶ Jours_Feries : correspond soit à un nombre de jours fériés, soit à une plage contenant les jours fériés. L'argument est facultatif.

Exemple :

Calcul du nombre de jours ouvrés entre le 15 décembre 2015 et le 31 décembre 2015. L'argument est le jour férié du 25 décembre 2015.

Résultat : Le nombre de jours ouvrés est 12.

The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E
1	15/12/2015	31/12/2015	12		
2	25/12/2015				
3					
4					

The formula bar at the top shows the formula: `=NB.JOURS.OUVRES(A1;B1;A2)`

Additionner un nombre de jours ouvrés à une date

La fonction SERIE.JOUR.OUVRE permet de renvoyer un nombre qui représente une date correspondant à une date (date de début) à laquelle est ajouté ou soustrait le nombre de jours ouvrés spécifiés.

La syntaxe est la suivante :

```
=SERIE.JOUR.OUVRE(date_debut ; nb_jours ; [jours_feries])
```

- ▶ Date_debut : correspond à la date de base au calcul.
- ▶ Nb_jours : nombre de jours à ajouter à la date de base.
- ▶ Jours_Feries : correspond soit à un nombre de jours fériés, soit à une plage contenant les jours fériés. L'argument est facultatif.

Exemple : ajouter 25 jours ouvrés à la date du 15 décembre 2015.

Les jours fériés sont le 25 décembre 2015 et le 1er janvier 2016. Il est possible d'afficher un jour férié qui n'est pas dans la plage (par exemple ajouter le 14 juillet, n'empêcherait pas de calculer la date du 21 janvier 2016).

	A	B	C	D	E	F
1	15/12/2015	25	21/01/2016			
2	25/12/2015					
3	01/01/2016					

Calcul de la fin de mois

La formule FIN.MOIS permet de connaître le dernier jour du mois d'une date donnée. Il est également possible d'ajouter un nombre de mois à ajouter.

La syntaxe de la formule est la suivante :

```
=FIN.MOIS(date ; nb_mois)
```

- ▶ Date : correspond à la date de début du calcul ;
- ▶ Nb_mois : correspond au nombre de mois à ajouter à la date pour calculer la fin de mois.

Exemple :

	A	B	C	D
1	Date	Consigne	Résultat	Formule utilisée
2	15/12/2015	Fin de mois	31/12/2015	=FIN.MOIS(A2;0)
3	15/12/2015	60 jours fin de mois	29/02/2016	=FIN.MOIS(A3+60;0)
4	15/12/2015	Fin de mois dans 3 mois	31/03/2016	=FIN.MOIS(A4;3)
5	15/12/2015	1er jour mois suivant	01/01/2016	=FIN.MOIS(A5;0)+1

b. Mise en forme conditionnelle avancée

La mise en forme conditionnelle peut prendre des formes plus complexes que celles vues jusque-là. En effet, il est possible de baser l'affichage de la mise en forme conditionnelle sur des formules conditionnelles.

L'exemple étudié ci-dessous permet de surligner de jaune et de mettre en gras les chiffres pairs :

	A	B	C
1	2	3	8
2	4	1	5
3	9	6	7

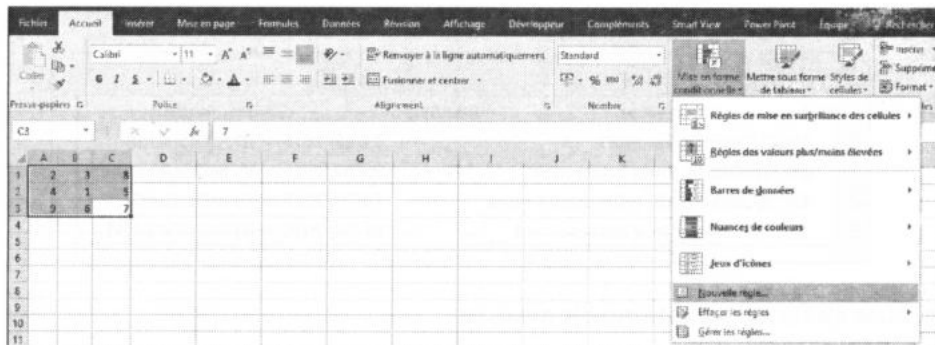
Formule conditionnelle pour mise en forme

La formule conditionnelle aura pour résultat VRAI ou FAUX. Lorsque le résultat de la formule est VRAI, la mise en forme conditionnelle est appliquée, si le résultat de la formule est FAUX, la mise en forme conditionnelle n'est pas appliquée.

La formule doit être unique pour toute la plage : sur une plage A1:C3, elle doit s'adapter aussi bien à A1 qu'à C3.

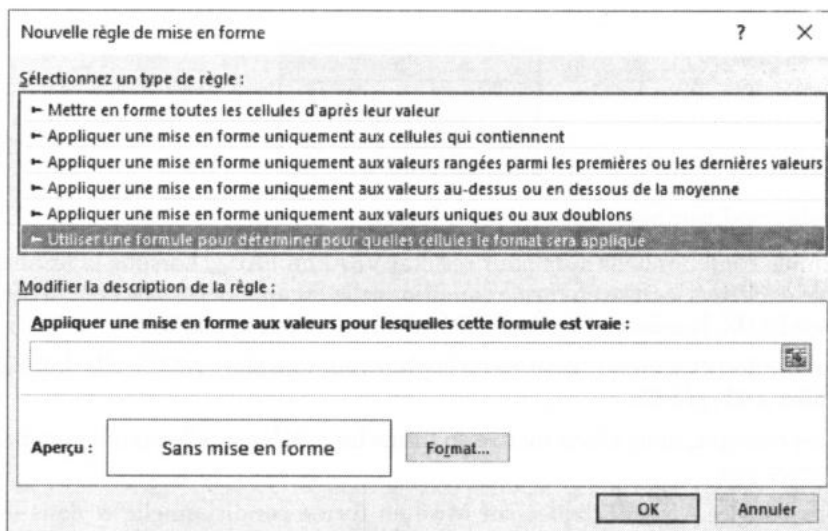
Dans cet exemple, nous allons mettre en forme les cellules si celles-ci ont comme valeur un nombre pair.

- ☞ Dans l'onglet Accueil, cliquez sur **Mise en forme conditionnelle** et dans le menu cliquez sur **Nouvelle règle...**



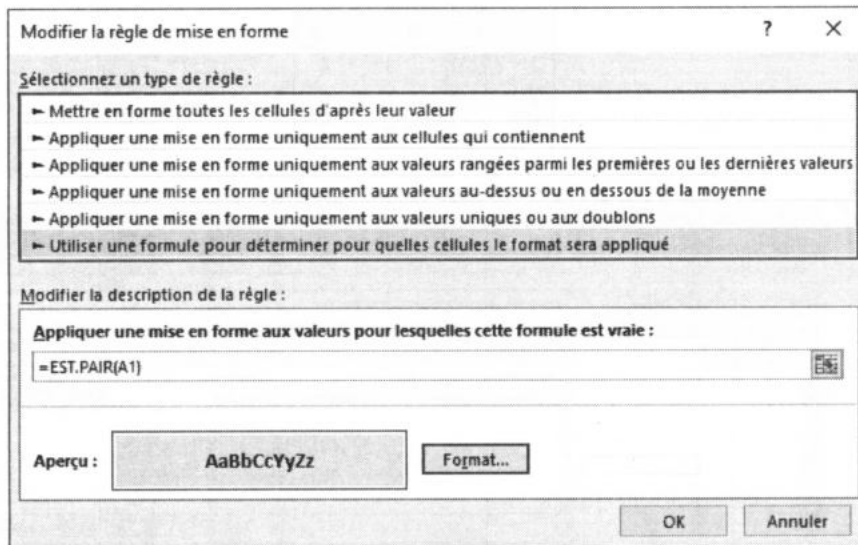
La fenêtre Nouvelle règle de mise en forme s'affiche.

- ☞ Cliquez sur Utiliser une formule pour déterminer pour quelles cellules le format sera appliqué :



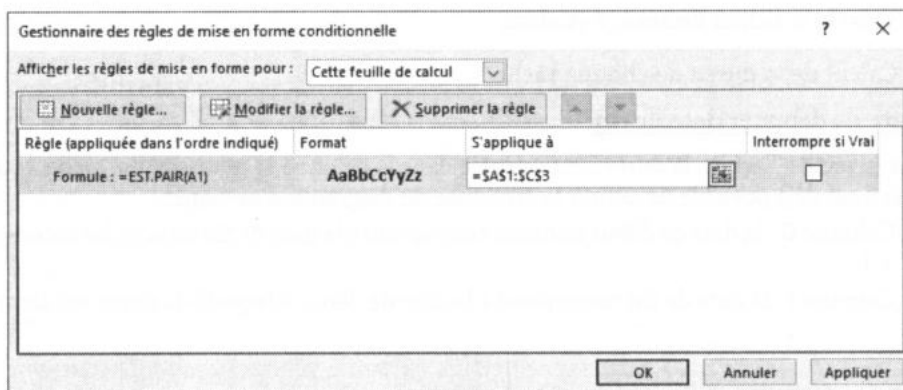
Dans la zone Appliquer une mise en forme aux valeurs pour lesquelles cette formule est vraie, saisissez la formule que vous souhaitez appliquer :
=EST.PAIR(A1)

☞ Ajoutez une mise en forme : ici style gras et fond jaune.



☞ Cliquez sur OK.

☞ Cliquez sur Mise en forme conditionnelle puis sur Gérer les règles pour accéder à la fenêtre Gestionnaire des règles de mise en forme conditionnelle. Dans la zone S'applique à, sélectionnez la plage A1:C3.



Le résultat est le suivant :

	A	B	C
1	2	3	8
2	4	1	5
3	9	6	7



Si dans la formule `=EST.PAIR(A1)`, nous figeons la colonne (`=EST.PAIR($A1)`), alors nous obtiendrons le résultat suivant :

Dans ce cas, les cellules B1 et B2 deviennent jaunes car les cellules A1 et A2 sont paires (ce ne sont plus les cellules B1 et B2 qui sont testées mais A1 et A2).

3. Réalisation de l'exemple

📁 Ouvrez le fichier `Enoncé_5-A.xlsm`.

a. Calcul de la durée de chaque tâche

Date de début et date de fin

Sur la feuille `Taches`, la durée sera calculée dans la colonne H pour chaque tâche, toutefois il est déjà possible de définir la structure du diagramme de Gantt.

- ▶ Colonne G : la date de début minimal correspond à la date de fin de la tâche antérieure + 1.
- ▶ Colonne I : la date de fin correspond à la date de début à laquelle la durée est ajoutée.

- Appuyez sur la touche **F2** pour éditer la formule de la cellule I4.
- Saisissez dans la cellule `=SERIE.JOUR.OUVRE(G4;H4;Férié)` puis appuyez simultanément sur les touches **Ctrl** **↵** pour appliquer la formule à l'ensemble de la plage.

	F	G	H	I	J	K	L	M
1								
2								
3	Tache antérieure	Date de début minimal	Durée	Date de fin	01/04/2016	02/04/2016	03/04/2016	04/04/2016
4		01/04/2016		01/04/2016				
5	Tache 1	02/04/2016		02/04/2016				
6	Tache 1	02/04/2016		02/04/2016				
7	Tache 3	03/04/2016		03/04/2016				
8	Tache 4	04/04/2016		04/04/2016				
9	Tache 4	04/04/2016		04/04/2016				
10	Tache 6, Tache 5	05/04/2016		05/04/2016				
11	Tache 7	06/04/2016		06/04/2016				
12								

Calculer la durée nécessaire

Tout d'abord, il est nécessaire de récupérer le nombre de ressources disponibles par métier et par jour. La formule `NB.SI.ENS` permettra de compter le nombre de ressources ayant indiqué leur disponibilité par poste.

- Dans la feuille **Planning**, saisissez les différents postes en dessous du tableau (plage B27:B30) :

	A	B
26	Ressource 23	Testeur
27		Manager
28		Concepteur
29		Développeur
30		Testeur

- Sélectionnez la plage entière C27:C30.
- La formule va être un `NB.SI.ENS` avec deux critères : que le poste de la ressource en B4:B26 soit égal au poste dans la plage B27:B30 et que la cellule de la plage soit égale à X : `=NB.SI.ENS(B4:B26;$B27:C$4:C$26;"X")`
- Appuyez sur les touches **Ctrl** **↵** simultanément pour appliquer la formule à l'ensemble de la plage.

Imbrication des boucles

Dans l'exemple, il faut parcourir les données disponibles dans la feuille **Planning** pour calculer la durée nécessaire pour l'accomplissement d'une tâche. Il y a 8 tâches, 4 postes, 90 jours, par conséquent, il convient d'imbriquer trois boucles :

► Boucle sur les tâches

La boucle sur les tâches va permettre de se déplacer de la ligne 4 à la ligne 11 sur la feuille **Taches**. À chaque itération de la boucle, il faut réinitialiser les valeurs attendues pour chaque tâche : nombre de jours nécessaires pour chacun des postes, date de début de la tâche.

► Boucle sur les postes

Pour chacune des tâches, le nombre de ressources disponibles par jour est à récupérer. Par conséquent, il faut se déplacer de la ligne 27 à la ligne 30 sur la feuille **Planning**.

► Boucle sur les dates

Une fois les tâches et les postes définis, il faut parcourir toutes les dates du projet dans la feuille **Planning**. De ce fait, il faut se déplacer de la colonne 3 à la colonne 93 de la feuille **Planning**.

► Structure

```
For Ligne = 4 To 11
    For Poste = 27 To 30
        For Colonne = 3 To 93
            Next
        Next
    Next
```

☞ À l'intérieur de la boucle sur les tâches, initialisez les valeurs des variables pour la suite du calcul :

```
'initialiser les valeurs
DureeMax = 0
'stocker dans la variable le nombre de jours par poste
nécessaire pour réaliser la tâche
NbManager = FTaches.Cells(Ligne, 2).Value
NbConcepteur = FTaches.Cells(Ligne, 3).Value
NbDev = FTaches.Cells(Ligne, 4).Value
NbTest = FTaches.Cells(Ligne, 5).Value
'Date à partir de laquelle il faut rechercher des ressources
disponibles
DateDebut = CDate(FTaches.Cells(Ligne, 7).Value)
```

☞ Dans la boucle **Poste**, une valeur attendue est définie pour chaque poste correspondant au nombre. Il convient également d'initialiser la durée relative pour chaque poste, c'est-à-dire la durée totale nécessaire pour consommer toutes les ressources requises pour un poste.


```
'Initialisation de la durée maximale relative
DureeRelative = 0
'Initialisation de la valeur attendue
Select Case Poste
Case 27
ValeurAttendue = NbManager
Case 28
ValeurAttendue = NbConcepteur
Case 29
ValeurAttendue = NbDev
Case 30
ValeurAttendue = NbTest
End Select
```

Dans la boucle sur les jours, il convient de parcourir l'ensemble des dates pour décompter le nombre de ressources disponibles pour le poste. Avant toute chose, il est nécessaire de contrôler si la date respecte les critères suivants :

- ▶ N'est pas un weekend : utilisation de la formule `Weekday (Date, param_debut_semaine)`. Si `param_debut_semaine` est égal à 2, cela signifie que le numéro du jour sera calculé à partir du lundi. Les samedi et dimanche prendront donc respectivement les valeurs 6 et 7.
- ▶ N'est pas un jour férié : utilisation de la fonction `EstFerie` déjà disponible dans le module. La fonction renvoie `VRAI` si c'est un jour férié, `FAUX` sinon
- ▶ N'est pas inférieure à la date de début : si la date de début de la tâche n'est pas encore effective, il n'est pas possible de récupérer le nombre de ressources disponibles ce jour-là.

🔗 Saisissez le code suivant :

```
For Colonne = 3 To 93
    Dim DateEnCours As Date
    DateEnCours = CDate(FPlanning.Cells(3, Colonne).Value)
    'contrôler si la date de début est passée
    If DateEnCours >= DateDebut And Weekday(DateEnCours, 2)
< 6 Or EstFerie(CStr(DateEnCours)) Then
        End If
    Next
```

Si la date respecte les conditions, il faut déduire de la valeur attendue (`ValeurAttendue`), les ressources disponibles pour le poste et la date en cours (`DateEnCours`).

Si la valeur attendue est négative ou nulle, il faut calculer la durée avec la fonction `WorksheetFunction.NetworkDays(debut, fin, jours_ferie)` qui correspond à la fonction `NB.JOURS.OUVRES` en VBA.

Si cette durée relative (`DureeRelative`) calculée pour ce poste est supérieure à la durée maximale actuelle (`DureeMax`), la durée maximale actuelle (`DureeMax`) sera égale à la durée relative (`DureeRelative`).

Une fois que la durée maximale est obtenue pour un poste, le programme quitte la boucle avec l'instruction `Exit For`.

```
ValeurAttendue = ValeurAttendue - FPlanning.Cells(Poste,
Colonne).Value
    If ValeurAttendue <= 0 Then
        DureeRelative = WorksheetFunction.NetworkDays
(DateDebut, DateEnCours, Range("Férié"))
        If DureeRelative > DureeMax Then
            DureeMax = DureeRelative
        End If
    Exit For
End If
```

Avant de passer à la nouvelle tâche (avant le `Next` de la boucle sur les lignes de tâches), mettez à jour la durée comme étant la valeur `DureeMax`.

```
FTaches.Cells(Ligne, 8).Value = DureeMax
```

Le résultat de la procédure est le suivant :

```
Sub CalculerDuree()
'Déclaration des variables feuilles et affectation de la valeur
Dim FTaches, FPlanning As Worksheet
Set FTaches = ActiveWorkbook.Sheets("Taches")
Set FPlanning = ActiveWorkbook.Sheets("Planning")
'Déclaration des variables de compteur pour chaque poste
Dim NbManager, NbConcepteur, NbDev, NbTest As Integer
'DureeMax sera la variable qui stockera la durée maximale pour les 4
postes, la durée relative stockera la durée maximale pour un poste
Dim DureeMax, DureeRelative As Integer
'Valeur attendue
Dim ValeurAttendue As Integer
'Date de début de la tâche
Dim DateDebut As Date
'Parcourir les 8 tâches qui vont de la ligne 4 à la ligne 11
For Ligne = 4 To 11
    'Initialiser les valeurs
    DureeMax = 0
    'Stocker dans la variable le nombre de jours par poste
nécessaire pour réaliser la tâche
    NbManager = FTaches.Cells(Ligne, 2).Value
    NbConcepteur = FTaches.Cells(Ligne, 3).Value
    NbDev = FTaches.Cells(Ligne, 4).Value
    NbTest = FTaches.Cells(Ligne, 5).Value
    'Date à partir de laquelle il faut rechercher des ressources
disponibles
    DateDebut = CDate(FTaches.Cells(Ligne, 7).Value)
    For Poste = 27 To 30
```

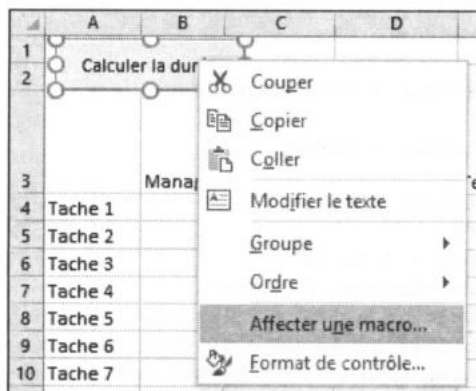
```

'Initialisation de la durée maximale relative
DureeRelative = 0
'Initialisation de la valeur attendue
Select Case Poste
Case 27
ValeurAttendue = NbManager
Case 28
ValeurAttendue = NbConcepteur
Case 29
ValeurAttendue = NbDev
Case 30
ValeurAttendue = NbTest
End Select
For Colonne = 3 To 93
    Dim DateEnCours As Date
    DateEnCours = CDate(FPlanning.Cells(3, Colonne).Value)
    'Contrôler si la date de début est passée
    If DateEnCours >= DateDebut And Weekday(DateEnCours, 2)
< 6 Or EstFerie(CStr(DateEnCours)) Then
        'Decompte des ressources par rapport à la valeur
attendue
        ValeurAttendue = ValeurAttendue - FPlanning.Cells
(Poste, Colonne).Value
        'si la valeur attendue est inférieure ou égale à 0,
on calcule la durée relative pour faire la tâche
        If ValeurAttendue <= 0 Then
            DureeRelative = WorksheetFunction.NetworkDays
(DateDebut, DateEnCours, Range("Férieré"))
            'si la durée relative est supérieure à la durée
maximum jusque-là, mettre à jour la durée maximum.
            If DureeRelative > DureeMax Then
                DureeMax = DureeRelative
            End If
        End If
    End For
End If
Next
Next
'Mise à jour de la durée
FTaches.Cells(Ligne, 8).Value = DureeMax
Next
End Sub

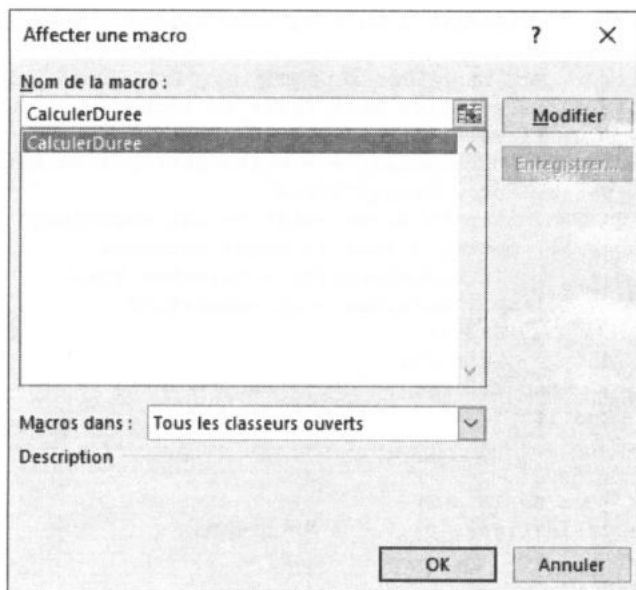
```

Relier le code au bouton

- Sur la feuille Taches, faites un clic droit sur le bouton puis cliquez sur **Affecter une macro...**



- Choisissez la macro `CalculerDuree` dans la liste des macros disponibles.



Cliquez sur OK.

- Cliquez sur le bouton **Calculer la durée**.

La procédure écrite ci-dessus s'exécute et les durées sont calculées.

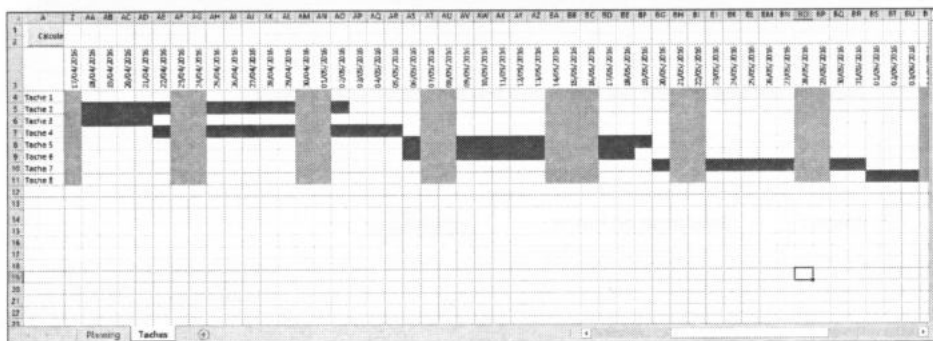
Date de début minimal	Durée	Date de fin
01/04/2016	9	14/04/2016
15/04/2016	11	02/05/2016
15/04/2016	4	21/04/2016
22/04/2016	9	05/05/2016
06/05/2016	8	19/05/2016
06/05/2016	7	18/05/2016
20/05/2016	7	31/05/2016
01/06/2016	7	10/06/2016

b. Mise en forme du diagramme de Gantt

Une fois les durées calculées, il faut afficher le diagramme de Gantt sur la feuille Taches de manière automatisée avec la mise en forme conditionnelle.

Voici le résultat attendu :

- ▶ Mise en forme pour un week-end ou jour férié ;
- ▶ Mise en forme pour les jours où une tâche se déroule.



Mise en forme pour un week-end ou jour férié

- ☞ Sélectionnez la plage Taches.



- ☞ Dans l'onglet Accueil, cliquez sur Mise en forme conditionnelle puis choisissez Nouvelle règle...

La formule conditionnelle doit respecter l'une des conditions suivantes pour avoir une mise en forme :

- ▶ Être un week-end : `JOURSEM(jour ; 2) > 5`
- ▶ Être un jour férié : `EstFerie(date)`



La formule EstFerie(date) existe dans le Module1.

- ☞ Cliquez sur Utiliser une formule pour déterminer pour quelles cellules le format sera appliqué pour saisir une formule.

Nouvelle règle de mise en forme

Sélectionnez un type de règle :

- ▶ Mettre en forme toutes les cellules d'après leur valeur
- ▶ Appliquer une mise en forme uniquement aux cellules qui contiennent
- ▶ Appliquer une mise en forme uniquement aux valeurs rangées parmi les premières ou les dernières valeurs
- ▶ Appliquer une mise en forme uniquement aux valeurs au-dessus ou en dessous de la moyenne
- ▶ Appliquer une mise en forme uniquement aux valeurs uniques ou aux doublons
- ▶ Utiliser une formule pour déterminer pour quelles cellules le format sera appliqué

Modifier la description de la règle :

Appliquer une mise en forme aux valeurs pour lesquelles cette formule est vraie :

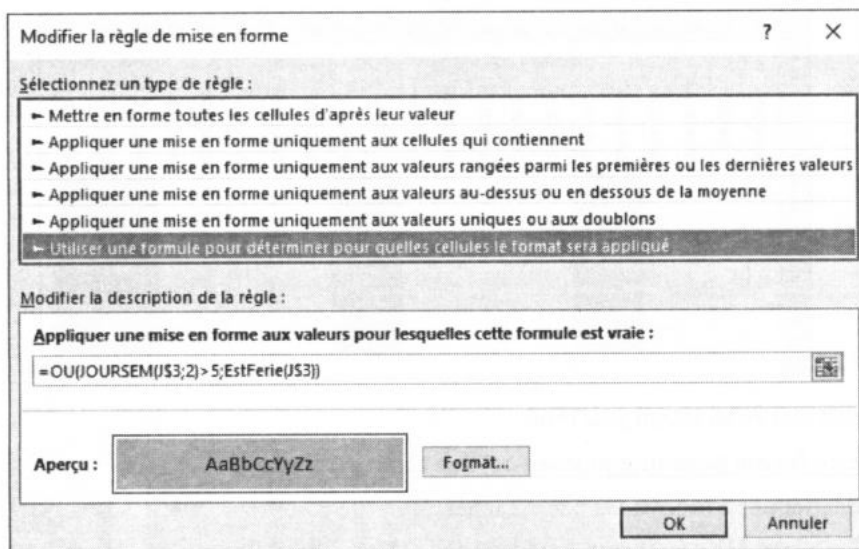
Aperçu : Sans mise en forme Format...

OK Annuler

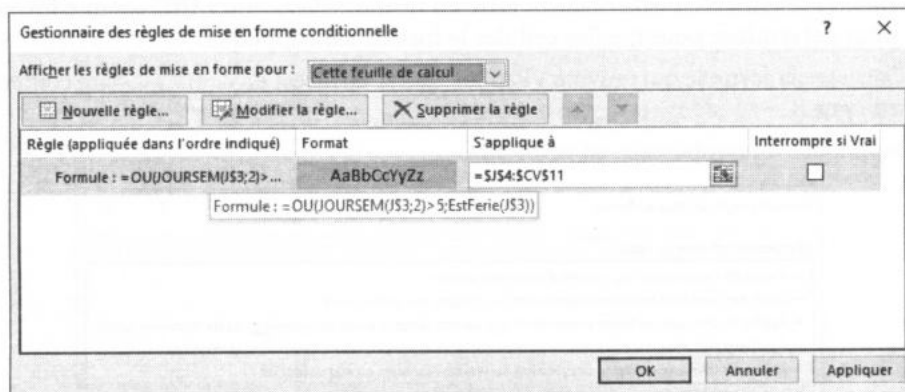
- ☞ Saisissez la formule suivante : `=OU(JOURSEM(J$3;2)>5;EstFerie(J$3))`

Bien que la plage Taches (J4:CO11) soit parcourue, la ligne 3 sera figée dans la formule puisque c'est cette ligne qui contient la date à analyser.

☞ Appliquez une mise en forme avec un fond gris pour obtenir le résultat suivant :



☞ Cliquez sur OK.



Le résultat est le suivant :

	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	A5	AT	AU	AV	AW	AX	AY	AZ	BA	BB	BC	B
1																						
2																						
3	26/04/2016	27/04/2016	28/04/2016	29/04/2016	30/04/2016	01/05/2016	02/05/2016	03/05/2016	04/05/2016	05/05/2016	06/05/2016	07/05/2016	08/05/2016	09/05/2016	10/05/2016	11/05/2016	12/05/2016	13/05/2016	14/05/2016	15/05/2016	16/05/2016	
4																						
5																						
6																						
7																						
8																						
9																						
10																						
11																						
12																						

Ici le 16 mai 2016 est un jour férié.

Mise en forme pour une journée où une tâche se déroule

- ❖ Sélectionnez à nouveau la plage Taches.
- ❖ Cliquez sur **Mise en forme conditionnelle - Nouvelle règle** pour créer la règle portant sur la tâche en cours.
- ❖ Dans la fenêtre **Nouvelle règle de mise en forme**, sélectionnez **Utiliser une formule pour déterminer pour quelles cellules le format sera appliqué**.
- ❖ Saisissez la formule qui renvoie VRAI lorsque la tâche est en **cours** à la date contenue en ligne 3 : `=ET(J$3>=$G4;J$3<=$I4)`
- ❖ Appliquez un remplissage en vert foncé.

Nouvelle règle de mise en forme ? X

Sélectionnez un type de règle :

- Mettre en forme toutes les cellules d'après leur valeur
- Appliquer une mise en forme uniquement aux cellules qui contiennent
- Appliquer une mise en forme uniquement aux valeurs rangées parmi les premières ou les dernières valeurs
- Appliquer une mise en forme uniquement aux valeurs au-dessus ou en dessous de la moyenne
- Appliquer une mise en forme uniquement aux valeurs uniques ou aux doublons
- Utiliser une formule pour déterminer pour quelles cellules le format sera appliqué

Modifier la description de la règle :

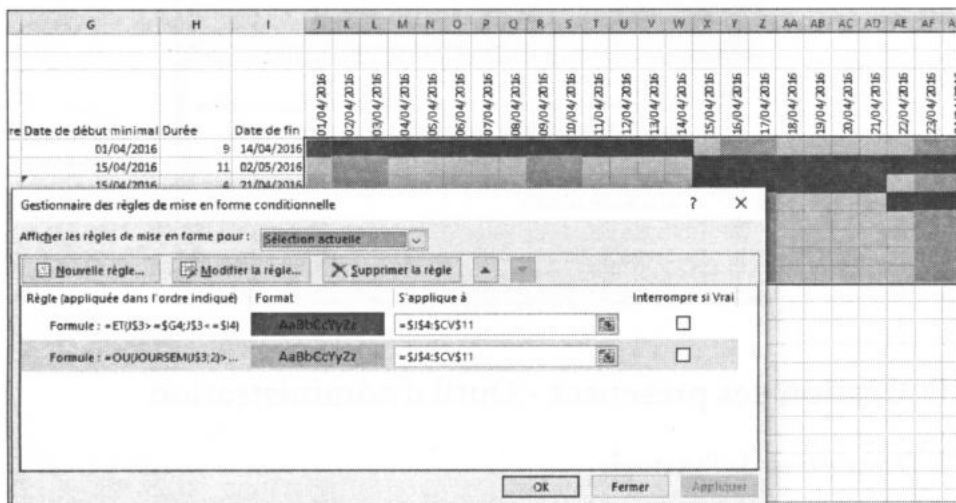
Appliquer une mise en forme aux valeurs pour lesquelles cette formule est vraie :

=ET(J\$3>=\$G4;J\$3<=\$I4)

Aperçu : AaBbCcYyZz Format...

OK Annuler

- ☞ Vérifiez dans la fenêtre **Gestionnaire des règles de mise en forme conditionnelle** que les formules sont bien appliquées sur la bonne plage :



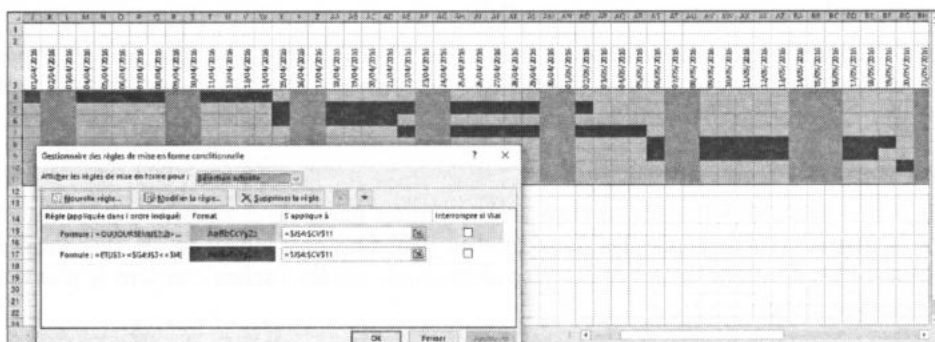
Notez que l'ordre des règles est incorrect : en effet tel que présenté, il semble que les tâches soient exécutées durant les jours fériés et les week-ends. Par conséquent, il faut changer l'ordre des règles.

- ☞ Sélectionnez la règle sur les week-ends puis mettez-la en premier dans la liste en

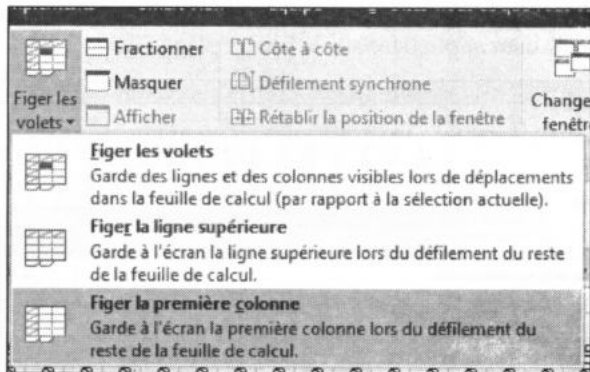
cliquant sur le bouton **Monter** .

- ☞ Cliquez sur **Appliquer**.

Vous obtenez le résultat suivant :



- ☞ Dans l'onglet **Affichage** - groupe **Fenêtre**, cliquez sur **Figer les volets** puis sur **Figer la première colonne**.



B. Gestion des présences - Outil d'administration

1. Description de l'exemple

a. Présentation de l'exemple

L'objectif de cette seconde partie est de reprendre le fichier de disponibilité des ressources et d'y ajouter certaines fonctionnalités pour faciliter son utilisation.

En l'état, n'importe quel utilisateur du fichier peut modifier les présences/absences de toutes les ressources. La suite de cet exemple permettra de restreindre par un mot de passe la modification des disponibilités.

L'autre objectif de cet exemple est aussi de calculer le coût de chaque tâche et du projet au total. Le coût d'une tâche correspond au coût de chaque ressource requise pour une tâche. Il est demandé également d'optimiser le coût de chaque tâche en prenant la ressource la moins coûteuse pour une tâche.

Une contrainte très importante est à intégrer : le calendrier n'est pas fixe et il peut s'étendre bien au-delà de juin. Le fichier doit être en capacité de s'adapter à cette contrainte et générer autant de mois que nécessaire dans le calendrier.

b. Présentation du fichier

Le fichier `Enoncé_5-B.xlsm` est basé sur la première partie de ce chapitre puisqu'il s'agit de la même structure avec les deux feuilles `Planning` et `Taches`. La feuille `Planning` contient les données sur les ressources alors que la feuille `Taches` contient le planning de chaque tâche.

Par rapport au fichier **Corrigé_5-A.xlsm**, il existe des informations supplémentaires sur la feuille **Planning** :

- ▶ **Matricule** (colonne B) : il s'agit d'un code unique et secret qui permet d'identifier les utilisateurs. Seuls eux en ont connaissance. Ce code sera utilisé pour leur permettre de s'identifier dans l'application.
- ▶ **Coût** (colonne D) : le coût correspond au montant journalier à payer pour avoir la ressource sur le projet.
- ▶ Ces modifications impliquent un décalage des plages existantes :
 - ▶ Le poste est en **colonne C** sur la feuille **Planning**.
 - ▶ La plage **Planning** est décalée : `=Planning!E4:CQ26`.
 - ▶ La plage **Férié** est décalée : `=Planning!CS4:CSS6`.
 - ▶ Une plage **Poste** a été créée contenant la liste des postes : `=Planning!CT4:CT7`.
- ▶ La cellule **B1** permettra de connaître le nombre de dates contenues dans la feuille **Planning**. Pour rappel la première date se situe en cellule **E3** (01/04/2016), les autres dates sont positionnées à la suite sur la ligne 3. Pour comptabiliser le nombre de dates, il faut compter le nombre de cellules non vides sur la ligne 3 auquel il faut soustraire les cellules non vides ne contenant pas de dates. La formule est la suivante : `=NB.SI(3:3;"<>")-6` (il existe 6 cellules non vides ne contenant pas de dates sur la ligne 3, **A3:D3** et **CS3:CT3**).
- ▶ La cellule **D1** permettra de connaître le nombre de ressources contenues dans la feuille **Planning**. Pour rappel la première ressource se situe en cellule **A4** (Ressource 1), les autres ressources sont positionnées à la suite sur la colonne A. Pour comptabiliser le nombre de ressources, il faut compter le nombre de cellules non vides sur la colonne A auquel il faut soustraire les cellules non vides ne contenant pas de ressources. La formule est la suivante : `=NB.SI(A:A;"<>")-3` (il existe 3 cellules non vides ne contenant pas de ressources dans la colonne A, **A1:A3**).
- ▶ Les cellules **B2**, **C2**, **D2** permettent le calcul de coût. Elles seront détaillées dans la réalisation de l'exemple.

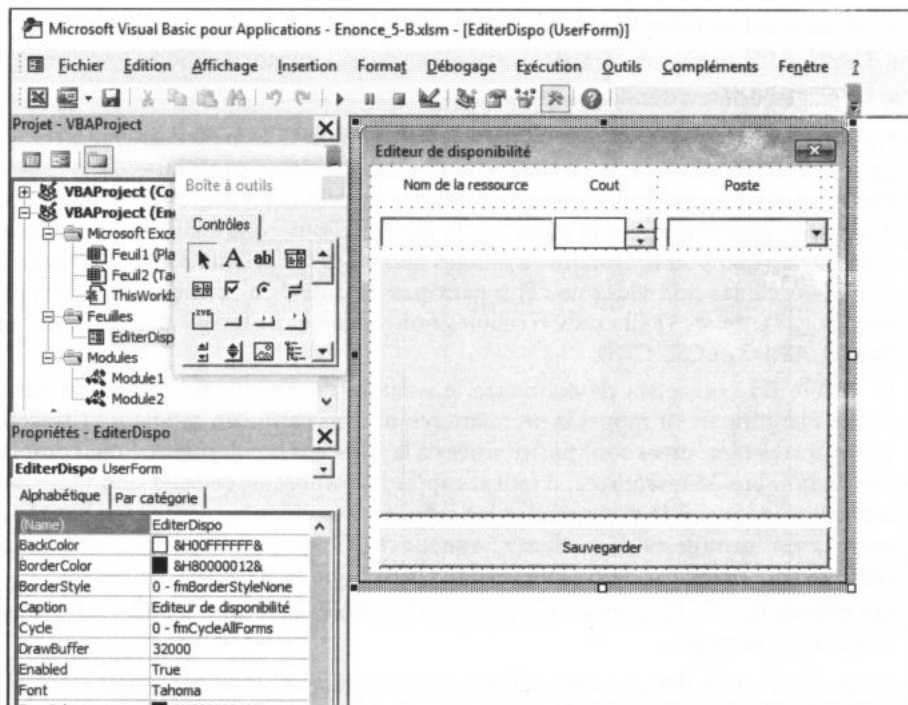
Sur la feuille **Taches**, deux nouveaux boutons sont présents et se nomment :

- ▶ **Editer la disponibilité par personne**. Il est associé à une procédure VBA pour l'instant vide se nommant `AfficherFormulairePersonne`. Cette procédure sera utilisée dans l'exemple pour charger le formulaire propre à chaque ressource.
- ▶ **Calculer le coût du projet**. Il est associé à une procédure VBA pour l'instant vide se nommant `CalculerCout`. Cette procédure sera utilisée dans l'exemple pour calculer le coût de chaque tâche et du projet.

Enfin, la cellule **G2** contiendra le coût total du projet.

	A	B	C	D	E	F	G
1							
2	Calculer la durée		Editer la disponibilité par personne.		Calculer le cout du projet		157 200,00 €

Concernant la partie VBA, il est à noter qu'un formulaire a été créé permettant l'édition de la disponibilité des ressources, ce formulaire est visible dans le dossier Feuilles du projet VBA.



Comme vous le constatez, un espace est prévu pour saisir les disponibilités mais le calendrier est actuellement vide. Voici le nom des différents contrôles du formulaire :

Nom du contrôle	Type de contrôle	Description
EditerDispo	Formulaire (UserForm)	Il s'agit du formulaire contenant les différents contrôles.
NomRessource	TextBox	Zone de texte saisissable pour saisir le nom de la ressource.
LibelleCout	Label	Coût de la ressource.

Nom du contrôle	Type de contrôle	Description
BoutonPlusMoinsCout	SpinButton (bouton ayant deux méthodes Up <i>Haut</i> et Down <i>Bas</i> permettant d'enclencher un évènement par un clic sur la flèche haute et un autre par un clic sur la flèche basse).	Boutons permettant de faire varier le coût à la hausse ou à la baisse. Le code est déjà existant pour faire varier la valeur du coût.
ListePoste	Combobox	Liste déroulante contenant les postes possibles : « Manager, Concepteur, Testeur, Développeur »
MultiPageCalendrier	MultiPage (il s'agit d'un objet permettant de contenir des objets de type Page qui correspondent à des onglets).	Il s'agit des onglets (Page) contenant les différents calendriers mois par mois.
Sauvegarder	CommandButton	Permet de sauvegarder les modifications.

c. Fonctionnalités

Les fonctionnalités proposées dans cet exemple sont les suivantes :

- Finaliser le formulaire : charger le formulaire par personne puis générer le calendrier dans le formulaire. Le formulaire ressemblera à cela :

- ▶ Calculer le coût du projet : calculer le coût du projet en fonction des ressources utilisées en optimisant l'utilisation de celles-ci. L'objectif est de prendre les ressources les moins coûteuses pour chaque tâche ;
- ▶ Bloquer l'accès à la feuille **Planning**.

2. Notions de cours

a. Création dynamique de contrôle

Qu'est-ce que c'est ?

Ajouter des contrôles dynamiquement signifie que de nouveaux contrôles sont créés sur le formulaire durant l'exécution.

L'ajout du contrôle est déclenché par une procédure que ce soit automatiquement ou par action de l'utilisateur.

Ajouter un contrôle se fait dans un contrôle conteneur, c'est-à-dire capable de contenir d'autres contrôles : Form, Frame, Page...

Comment ça fonctionne ?

Il faut créer un objet de type `Control`, puis l'ajouter sur un contrôle conteneur.

```
Dim MonControle As Control 'instanciation d'une variable de type Control
```

Ajout du contrôle dans un contrôle conteneur :

```
Set MonControle =  
Form.ControleConteneur.Controls.Add("forms.TextBox.1")  
' ajout d'une nouvelle TextBox dans le contrôle conteneur.
```

Pour les autres contrôles, voici la syntaxe à utiliser :

Contrôle	Syntaxe
ComboBox (zone de liste modifiable)	Forms.CheckBox.1
CommandButton (bouton de commande)	Forms.ComboBox.1
Frame (cadre)	Forms.CommandButton.1
Image	Forms.Frame.1
Label (étiquette)	Forms.Image.1
ListBox (zone de liste)	Forms.Label.1
MultiPage	Forms.ListBox.1
OptionButton (bouton d'option)	Forms.MultiPage.1
ScrollBar (barre de défilement)	Forms.OptionButton.1

Contrôle	Syntaxe
SpinButton (toupie)	Forms.ScrollBar.1
TabStrip (bande d'onglets)	Forms.SpinButton.1
TextBox (zone de texte)	Forms.TabStrip.1
ToggleButton (bascule)	Forms.TextBox.1

Une fois le contrôle créé, il est plus simple de modifier les propriétés du contrôle.

```
MonControle.Name = "NomDuControle"
MonControle.Left = 0 'positionnement sur l'axe des abscisses
MonControle.Top = 0 'positionnement sur l'axe des ordonnées
MonControle.Text = "Saisir un texte ici" 'Cas particulier d'une
TextBox puisque la propriété Text n'existe pas pour tous les contrôles.
```

b. Tableaux VBA

Un tableau permet de stocker des valeurs d'un type défini. Il y a autant de variables que de valeurs dans le tableau. Le tableau est particulièrement utile car c'est un outil très performant, comme par exemple lorsqu'il faut manipuler une plage de données importante. Dans ce cas, il est préférable d'utiliser les tableaux VBA plutôt que des calculs, Excel quitte à retranscrire les résultats ensuite dans Excel.

Dimension et définition

Les tableaux ont une dimension définie ou variable. La dimension est définie lorsqu'elle est indiquée à la création de la variable tableau. La dimension est variable lorsqu'elle n'est pas définie à la création de la variable tableau.

```
Dim Tableau() as String 'taille non définie
Dim TableauDef(5) as Integer 'taille définie
```

L'instruction `ReDim` permet de redéfinir la taille d'un tableau à la taille variable.

```
ReDim Tableau(4)
```

La redéfinition de la taille du tableau le réinitialise, c'est-à-dire que toutes les valeurs du précédent tableau sont supprimées et réinitialisées. L'instruction `Preserve` permet de conserver les valeurs du précédent tableau :

```
ReDim Preserve Tableau(5)
```

Le tableau peut être multidimensionnel c'est-à-dire qu'il peut avoir plusieurs dimensions :

```
Dim TableauMD(1 to 10, 1 to 15, 1 to 5) as String
Dim TableauMD2(10,15,5) as String
```

Ce tableau comportera un total de 750 valeurs possibles (10*15*5).

Option Base

L'instruction `Option Base` est écrite en amont du code dans un module avant la première procédure. Elle détermine si l'indice bas d'un tableau est 0 ou 1.

```
Option Base 0 'l'indice de base du tableau sera 0  
Option Base 1 'l'indice de base du tableau sera 1
```

Par défaut, option base 1 est appliqué.

UBound et LBound

L'instruction `UBound` permet d'obtenir l'indice le plus élevé d'un tableau.

L'instruction `LBound` permet d'obtenir l'indice le plus bas d'un tableau.

La syntaxe est la suivante :

```
UBound(Variable, [numero_dimension])
```

- ▶ `Variable` : variable sur laquelle porte l'instruction.
- ▶ `Numero_Dimension` : numéro de la dimension concernée dans le cas où il y a plusieurs dimensions.

La syntaxe est la même pour l'instruction `LBound`.

Quelques exemples :

```
Dim T1(10) as Integer  
Dim T2(1 to 5, 1 to 20) as String  
Dim T3(8,7,6) as Long  
Dim Val as String  
Val = UBound(T1) 'renvoie 10  
Val = UBound(T2,2) 'renvoie 20  
Val = UBound(T3,3) 'renvoie 6
```

3. Réalisation de l'exemple

🔗 Ouvrez le fichier `Enoncé_5-B.xlsm`.

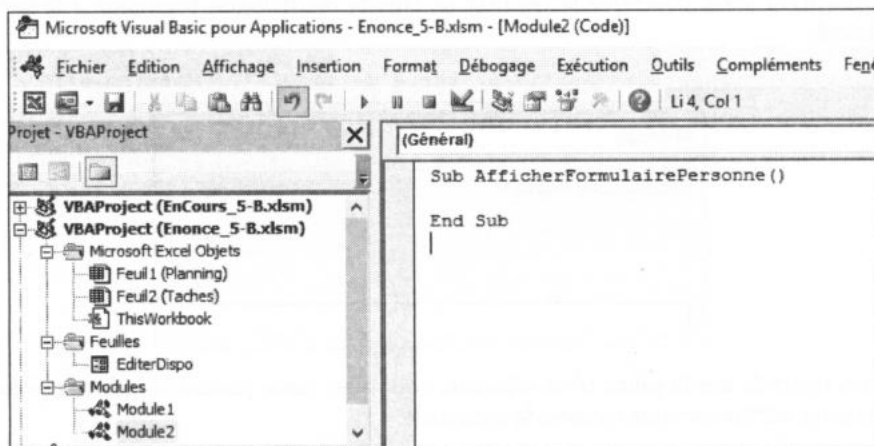
a. Initialisation du formulaire

La ressource voulant effectuer une modification de ses disponibilités ouvrira le fichier et sera positionnée sur la feuille `Taches`. Elle cliquera alors sur le bouton `Editer la disponibilité par personne` pour s'identifier.

L'identification passera par une fenêtre de type `InputBox` qui demandera le matricule de la ressource puis elle accédera au formulaire.

Connexion pour une ressource

L'ensemble de ce code doit être écrit au sein de la procédure `AfficherFormulairePersonne` puisque c'est ce formulaire qui est appelé lorsque la personne clique sur le bouton. Cette procédure est présente dans le `Module2`.



Voici la manipulation à réaliser :

- ✎ Tout d'abord, créez les variables :
 - ▶ Une variable de type chaîne de caractères pour stocker le matricule saisi par la personne ;
 - ▶ Une variable de type nombre entier pour parcourir les lignes du tableau contenant les ressources ;
 - ▶ Une variable publique contenant la ligne de la ressource trouvée. Celle-ci doit être définie avant la procédure et doit être initialisée à 0. L'intérêt d'avoir une variable publique est de connaître à tout moment durant l'exécution la ligne où se situe la ressource dans la feuille **Planning**. Cela sera particulièrement utile lors de la sauvegarde ;
 - ▶ D'autres variables seront créées plus loin dans le code.

- ✎ Écrivez le code suivant :

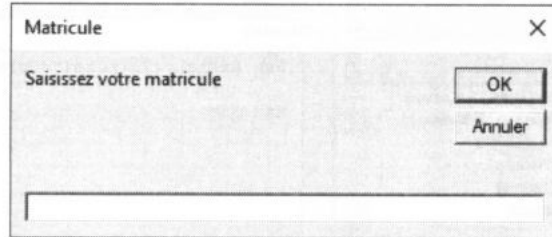
```
Public LigneRessource As Integer
Sub AfficherFormulairePersonne()
Dim Matricule As String
Dim Ligne As Integer
LigneRessource = 0
```

- ✎ Créez ensuite une invite de dialogue de type `InputBox` associée à la variable `Matricule`.

L'invite de dialogue `Inputbox` se décrit avec la méthode `Inputbox(intitulé, [titre], [valeur_par_defaut]...)` et renvoie une valeur de type `String` correspondant au texte saisi par l'utilisateur.

Dans ce cas, vous utiliserez uniquement les arguments intitulé (obligatoire) et titre (facultatif) et vous affecterez le résultat de l'invite de dialogue `Inputbox` à la variable `matricule` :

```
Matricule = InputBox("Saisissez votre matricule", "Matricule")
```



Aucun contrôle sur la saisie n'est effectué, vous allez juste parcourir la liste des matricules pour vérifier si vous trouvez le matricule.

Pour cela, vous allez créer une boucle de type `While...Wend` parcourant le tableau de la colonne B de la feuille **Planning** à partir de la ligne 4 jusqu'à ce qu'il trouve une colonne vide. Comme vu précédemment, la variable `Ligne` sera incrémentée à chaque itération de la boucle.

Si une correspondance est trouvée, c'est-à-dire que la valeur de la cellule de la colonne B et du numéro de ligne contenu dans la variable `Ligne` correspond au matricule saisi dans l'invite de dialogue `InputBox`, la variable `LigneRessource` prend la valeur de la ligne. Si la valeur de la variable `LigneRessource` est différente de 0, il s'agit également d'une condition de sortie de la boucle `While...Wend`.

☞ Saisissez donc le code suivant :

```
Ligne = 4 'première ligne parcourue est la ligne 4
Sheets("Planning").Activate 'sélection de la feuille Planning
While Cells(Ligne, 2).Value <> "" And LigneRessource = 0
    If Cells(Ligne, 2).Value = Matricule Then
        LigneRessource = Ligne
    End If
    Ligne = Ligne + 1
Wend
```

Suite à la sortie de la boucle, soit la variable `LigneRessource` est égale à 0 et cela signifie que le matricule n'a pas été trouvé et qu'il est donc incorrect, soit la variable `LigneRessource` est différente de 0 ce qui signifie que le programme a pu faire une correspondance avec l'une des valeurs existantes dans la colonne B de la feuille **Taches**.

Il convient donc de tester la valeur de la variable `LigneRessource`. Si la variable est égale à 0, une boîte de dialogue de type pop-up d'erreur `MsgBox` est affichée avec comme titre : **Erreur**, comme intitulé **Erreur sur le matricule** et comme bouton OK. Aucune instruction supplémentaire ne sera faite sur cette branche de code suite au message d'erreur. Cela reviendra à sortir du code.

En revanche, dans le cas où la variable `LigneRessource` est différente de 0, le code poursuivra avec l'affichage du formulaire.

La suite du code va se structurer ainsi :

```
If LigneRessource = 0 Then
    MsgBox "Erreur sur le matricule", vbOKOnly, "Erreur"
Else
    'génération du formulaire
End If
```

Génération du calendrier

La partie concernant la génération du calendrier est donc située à l'intérieur de l'instruction `Else` de la structure conditionnelle.

Toutes les dates seront parcourues à l'aide d'une boucle. La seule information connue est la première date : le 1^e avril. La date de fin peut-elle être variable, par conséquent, il faut passer une à une toutes les dates présentes dans la feuille **Planning**.

Le formulaire contient les informations de la ressource : nom, coût, poste mais surtout un calendrier qui se présente sous la forme d'un contrôle `MultiPage`, où chaque page correspondra à un mois. Les contrôles `Page` correspondent à des onglets. Elles seront générées dynamiquement, c'est-à-dire qu'elles seront générées uniquement si une date nécessite la création du contrôle `Page`.

À chaque date nous allons créer un contrôle de type `Checkbox` (case à cocher) qui sera contenu dans le contrôle `Page` du mois auquel il correspond. Chaque contrôle `Checkbox` sera positionné dans la page en fonction de deux critères :

- ▶ Son jour de la semaine (de gauche à droite : lundi, mardi...);
- ▶ Son numéro de semaine (de haut en bas : première semaine du mois, deuxième semaine du mois...).

Sélection ou création d'une page

Dans un premier temps, nous allons initier les variables dont nous avons besoin pour la suite de ce calcul.

La variable `Colonne` permettra de naviguer parmi les colonnes de dates. La variable `Mois` donnera la possibilité de stocker le nom du mois en cours. La variable `DateJour` contiendra la date en cours. Enfin, la variable `PageMois` permettra de créer les contrôles de type `Page` à insérer dans le `MultiPage`.

☞ Décrivez les variables ainsi :

```
Dim Colonne As Integer
Dim Mois As String
Dim DateJour As Date
Dim PageMois As Control
```

☞ Supprimez toutes les pages existantes dans le contrôle de type `MultiPage` avec la méthode `Clear`.

```
EditerDispo.MultiPageCalendrier.Pages.Clear
```

☞ Initialisez la variable `Colonne` avec le numéro de la première colonne contenant une date dans la feuille **Planning**.

```
Colonne = 5
```

Le code va parcourir l'ensemble des dates. Pour chaque date, récupérez le mois pour chercher s'il existe une page portant le nom de la page. Parcourez l'ensemble des pages grâce à la boucle `For ... Each` qui parcourt l'ensemble des objets d'une collection. Il faut contrôler si l'une des pages a un `tag` égal à la valeur de la variable `Mois`



Un tag est une propriété « libre » d'un objet. Il n'est pas visible, mais est accessible à tout moment dans le code.

☞ Rédigez le code suivant :

```
While Cells(3, Colonne).Value <> "" 'boucle sur l'ensemble des
dates
    DateJour = CDate(Cells(3, Colonne).Value) 'affectation de la
date à la variable DateJour
    Mois = Month(DateJour) & "/" & Year(DateJour) 'affectation
de la valeur à la variable Mois
'parcourir ensuite toutes les Pages du contrôle MultiPageCalendrier
pour voir si l'une d'elles possède l'attribut Tag qui est égal à la
variable Mois.
    For Each PA In EditerDispo.MultiPageCalendrier.Pages
        If CStr(PA.Tag) = Mois Then
            Set PageMois = PA
            Exit For
        End If
    Next
'Si la page n'existe pas, il faut la créer
    If PageMois Is Nothing Then
        Set PageMois = EditerDispo.MultiPageCalendrier.Pages.Add
("Page" & Mois, Mois)
        'Affecter la valeur Mois au Tag de l'objet PageMois.
        PageMois.Tag = Mois
    End If
```

Une fois le contrôle Page stocké dans la variable PageMois, il faut créer le contrôle de type CheckBox qui contiendra la date du jour et le positionner en fonction de la date.

La position du contrôle est définie en fonction de son jour (position horizontale) et de sa semaine (position verticale).

❖ Définissez deux variables stockant les positions de l'objet :

```
Dim PositionH As Integer
Dim PositionV As Integer
```

❖ Attribuez leur valeur en utilisant des fonctions Excel :

- ▶ PositionH prend la valeur du numéro du jour moins un. La position horizontale sera comprise entre 0 et 6.
- ▶ PositionV prend la valeur du numéro de semaine moins le numéro de la première semaine du mois. La position verticale sera comprise entre 0 et le nombre de semaine dans le mois.

```
PositionH = WorksheetFunction.Weekday(DateJour, 2) - 1
PositionV = WorksheetFunction.WeekNum(DateJour) -
WorksheetFunction.WeekNum(CDate("1/" & Mois))
```

Si le jour n'est ni un week-end ni un jour férié, alors il est possible de créer le contrôle dans le conteneur Page et de définir sa position et son texte. Enfin, le tag de l'objet sera utilisé pour faire le lien avec la cellule de la feuille Excel puisque nous stockerons dans cette propriété la colonne.

```
'Contrôler si le numéro de jour est inférieur à 5 et si le jour n'est
pas férié
If PositionH < 5 Or EstFerie(DateJour) Then
    'Création d'un contrôle de type CheckBox
    Dim CBJ As Control
    Set CBJ = PageMois.Controls.Add("forms.checkbox.1")
'Définir la position de la CheckBox
    CBJ.Top = 5 + PositionV * 25
    CBJ.Left = 5 + PositionH * 50
    CBJ.Width = 50
'Le texte du contrôle correspond au jour
    CBJ.Caption = Day(DateJour)
'Si la valeur de la cellule est X, le contrôle CheckBox est coché
    If Cells(LigneRessource, Colonne).value = "X" Then
        CBJ.Value = True
    End If
    'Affecter la valeur de la variable colonne au tag du nouveau
    contrôle créé.
    CBJ.Tag = Colonne
    Set CBJ = Nothing
End If
```

- ☞ Terminez les opérations en réinitialisant la variable PageMois, puis en fermant la boucle sans oublier d'incrémenter la variable colonne.

```
Set PageMois = Nothing
  Colonne = Colonne + 1
Wend
```

Il faut ensuite terminer la procédure en initialisant les champs des autres formulaires.

La liste des postes est contenue dans la propriété RowSource du contrôle Combobox. Cette propriété permet d'affecter une plage en tant que source de données. Dans le cas présent, la propriété RowSource de la ComboBox prend la valeur "Poste".

Dans notre cas, la propriété RowSource du contrôle ListePoste est égale à Poste.

- ☞ Affichez dans le formulaire les informations de la personne :

```
'Nom de la ressource
EditerDispo.NomRessource.Text = Cells(LigneRessource, 1).Value
'Poste de la ressource
EditerDispo.ListePoste.Value = Cells(LigneRessource, 3).Value
'Coût de la ressource
EditerDispo.LibelleCout.Caption = Cells(LigneRessource, 4).Value
```

- ☞ Terminez en affichant le formulaire :

```
EditerDispo.Show
```

Voici le code en entier :

```
Public LigneRessource As Integer
Sub AfficherFormulairePersonne()
Dim Matricule As String
Dim Ligne As Integer
LigneRessource = 0
'Saisie du matricule
Matricule = InputBox("Saisissez votre matricule", "Matricule")
Ligne = 4 'première ligne parcourue est la ligne 4
Sheets("Planning").Activate 'sélection de la feuille Planning
While Cells(Ligne, 2).Value <> "" And LigneRessource = 0
  If Cells(Ligne, 2).Value = Matricule Then
    LigneRessource = Ligne
  End If
  Ligne = Ligne + 1
Wend
If LigneRessource = 0 Then
  MsgBox "Erreur sur le matricule", vbOKOnly, "Erreur"
Else
  'génération du formulaire
  Dim Colonne As Integer
  Dim Mois As String
```

```

Dim DateJour As Date
Dim PageMois As Control
EditerDispo.MultiPageCalendrier.Pages.Clear
Colonne = 5
While Cells(3, Colonne).Value <> "" 'boucle sur l'ensemble
des dates
    DateJour = CDate(Cells(3, Colonne).Value) 'affectation
de la date à la variable DateJour
    Mois = Month(DateJour) & "/" & Year(DateJour) 'affectation
de la valeur à la variable Mois
    'parcourir ensuite toutes les Pages du contrôle
MultiPageCalendrier pour voir si l'une d'elles possède
l'attribut Tag qui est égal à la variable Mois.
    For Each PA In EditerDispo.MultiPageCalendrier.Pages
        If CStr(PA.Tag) = Mois Then
            Set PageMois = PA
            Exit For
        End If
    Next
    'Si la page n'existe pas, il faut la créer
    If PageMois Is Nothing Then
        Set PageMois =
EditerDispo.MultiPageCalendrier.Pages.Add("Page" & Mois, Mois)
        'affecter la valeur Mois au Tag de l'objet PageMois.
        PageMois.Tag = Mois
    End If

    Dim PositionH As Integer
    Dim PositionV As Integer
    PositionH = WorksheetFunction.Weekday(DateJour, 2) - 1
    PositionV = WorksheetFunction.WeekNum(DateJour) -
WorksheetFunction.WeekNum(CDate("1/" & Mois))
    'contrôler si le numéro de jour est inférieur à 5 et
si le jour n'est pas férié
    If PositionH < 5 Or EstFerie(CStr(DateJour)) Then
        'Création d'un contrôle de type CheckBox
        Dim CBJ As Control
        Set CBJ = PageMois.Controls.Add("forms.checkbox.1")
        'définir la position de la CheckBox
        CBJ.Top = 5 + PositionV * 25
        CBJ.Left = 5 + PositionH * 50
        CBJ.Width = 50
        'Le texte du contrôle correspond au jour
        CBJ.Caption = Day(DateJour)
        'Si la valeur de la cellule est X, le contrôle CheckBox
est coché
        If Cells(LigneRessource, Colonne).Value = "X" Then

```

```

        CBJ.Value = True
    End If
    'affecter la valeur de la variable colonne au tag
du nouveau contrôle créé.
    CBJ.Tag = Colonne
    Set CBJ = Nothing
End If
Set PageMois = Nothing
Colonne = Colonne + 1
Wend
'Nom de la ressource
EditerDispo.NomRessource.Text = Cells(LigneRessource, 1).Value
'Poste de la ressource
EditerDispo.ListePoste.Value = Cells(LigneRessource, 3).Value
'Coût de la ressource
EditerDispo.LibelleCout.Caption = Cells(LigneRessource, 4).Value
'Afficher le formulaire
EditerDispo.Show
End If
End Sub

```

Sauvegarder les modifications

La sauvegarde se produit lorsque l'utilisateur clique sur le bouton **Sauvegarder** dans le formulaire. Par conséquent, la procédure suivante est rattachée à l'évènement `Click` sur le bouton nommé `Sauvegarder`.

Pour la sauvegarde des modifications, il est nécessaire de parcourir tous les contrôles `Page` du contrôle `MultiPageCalendrier` et dans chacun des contrôles `Page`, il faut rechercher tous les contrôles, et plus particulièrement les contrôles de type `Checkbox`.

Pour chaque contrôle `CheckBox`, il faut récupérer le contenu de sa propriété `Value` pour savoir si la ressource est disponible (`Value` à `False` si la ressource a décoché la case donc qu'elle n'est pas disponible, `Value` à `True` si la ressource a coché la case donc qu'elle est disponible), et son tag qui contient la colonne correspondante dans la feuille `Planning`.

L'instruction `For Each ... Next` sera utilisée pour parcourir les contrôles `Page`, puis une nouvelle fois l'instruction `For Each ... Next` pour parcourir l'ensemble des contrôles `Checkbox`.

- ☛ Dans le volet de gauche, réalisez un clic droit sur le formulaire `EditerDispo` puis sélectionnez `Code` et saisissez le code suivant :

```

Private Sub Sauvegarder_Click
For Each PA In EditerDispo.MultiPageCalendrier.Pages
    For Each CB In PA.Controls
        'code
    Next
Next

```


- ☞ Pour chaque Checkbox, récupérez la colonne correspondante dans la feuille Planning qui est stockée dans la propriété Tag.

```
Dim Colonne As Integer
    Colonne = CInt(CB.Tag)
```

- ☞ En fonction de la propriété Value du contrôle Checkbox, mettez un X ou non dans la cellule. Si la propriété Value est égale à True, appliquez le fond vert, sinon retirez tout remplissage.

```
If CB.Value = True Then
    Cells(LigneRessource, Colonne).Value = "X"
Else
    Cells(LigneRessource, Colonne).Value = ""
End If
```

- ☞ Après les deux boucles, sauvegardez également les nouvelles valeurs saisies pour le nom, le poste et le coût :

```
Cells(LigneRessource, 1).Value = EditerDispo.NomRessource.Text
Cells(LigneRessource, 3).Value = EditerDispo.ListePoste.Value
Cells(LigneRessource, 4).Value = EditerDispo.LibelleCout.Caption
```

- ☞ Terminez avec un message situé dans un pop-up de type MsgBox qui a pour intitulé **Sauvegarde OK** et fermez le formulaire EditerDispo avec la méthode Hide.

```
MsgBox "Sauvegarde OK"
```

Voici la procédure complète pour la sauvegarde des données :

```
Private Sub Sauvegarder_Click()
For Each PA In EditerDispo.MultiPageCalendrier.Pages
    For Each CB In PA.Controls
        Dim Colonne As Integer
        Colonne = CInt(CB.Tag)
        If CB.Value = True Then
            Cells(LigneRessource, Colonne).Value = "X"
        Else
            Cells(LigneRessource, Colonne).Value = ""
        End If
    Next
Next
Cells(LigneRessource, 1).Value = EditerDispo.NomRessource.Text
Cells(LigneRessource, 3).Value = EditerDispo.ListePoste.Value
Cells(LigneRessource, 4).Value = EditerDispo.LibelleCout.Caption
MsgBox "Sauvegarde OK"
EditerDispo.Hide
End Sub
```

b. Bloquer l'accès à la feuille Planning

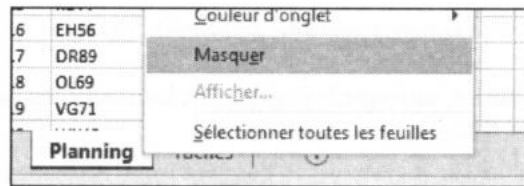
Pour bloquer l'accès à la feuille **Planning**, il faut procéder en deux temps :

- ▶ masquer la feuille **Planning** ;
- ▶ verrouiller la structure du document.

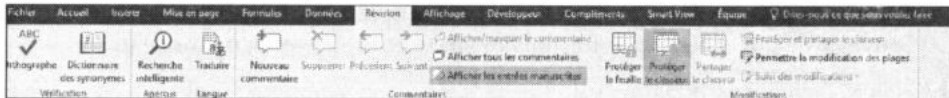
Pour réaliser cette manipulation :

- ❖ Dans Excel, faites un clic droit sur l'onglet correspondant à la feuille **Planning**.
- ❖ Dans le menu contextuel, cliquez sur **Masquer**.

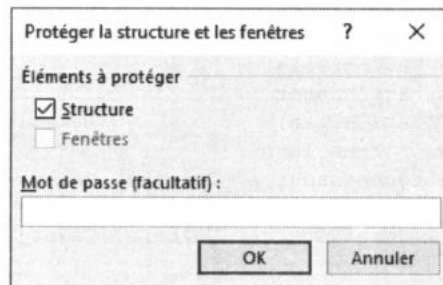
Votre feuille n'est plus visible mais un utilisateur peut l'afficher facilement.



- ❖ Dans l'onglet Révision - groupe Modifications, cliquez sur **Protéger le classeur**.



- ❖ Dans la fenêtre de verrouillage de la structure, saisissez le mot de passe **mdpenni**.



- ❖ Après avoir cliqué sur OK, confirmez le mot de passe dans la fenêtre suivante.

La feuille **Planning** est désormais masquée ; retirer la protection requiert la possession du mot de passe défini.

c. Calculer le coût du projet

Comme vous avez pu le constater, chaque ressource possède un coût qui lui est propre qui correspond à sa facturation journalière lorsqu'elle travaille.

Calculer le coût du projet correspond donc à la somme des journées travaillées par les ressources affectées au projet. Toutefois, l'équation entre les disponibilités et le travail effectué par une ressource n'est pas parfaite. En effet, dans certains cas la tâche requiert moins de ressources que l'ensemble des ressources disponibles pour travailler sur le projet. Dans ce cas de figure, l'intérêt sera d'optimiser les coûts en ne sélectionnant que les ressources les moins chères.

Comment faire pour trouver les ressources les moins chères pour une tâche ?

Tout d'abord, il convient d'identifier la période où se déroule la tâche. Pour cela, il suffit de récupérer la valeur contenue pour la date de début minimal en colonne G de la feuille **Taches**, et la valeur contenue pour la date de fin en colonne I de la feuille **Taches**. Il faut ensuite faire la correspondance avec les colonnes de la feuille **Planning**.

La cellule B2 de la feuille **Planning** contient la ligne de la tâche dans la feuille **Taches**. À partir de la valeur de la cellule B2, la cellule C2 contiendra une formule permettant de rechercher la date de début minimale de la tâche au sein de la ligne 3 de la feuille **Planning**.

Colonne de la date de début minimale de la tâche

La formule INDEX permet de récupérer la valeur de la date de début minimal de la feuille **Taches** à partir de la ligne de la tâche : =INDEX(Taches!G1:I11;Planning!B2;1)

À partir de cette valeur, on recherche la colonne contenant la valeur trouvée grâce à la formule précédente sur la ligne 3. La formule EQUIV permet de chercher la date minimale de la tâche dans la ligne 3.

```
=EQUIV(INDEX(Taches!G1:I11;Planning!B2;1);3:3;0)
```

Colonne de la date de fin de la tâche

La formule INDEX permet de récupérer la valeur de la date de fin de la feuille **Taches** à partir de la ligne de la tâche : =INDEX(Taches!G1:I11;Planning!B2;3)

À partir de cette valeur, il faut rechercher la colonne contenant la valeur trouvée grâce à la formule précédente sur la ligne 3. La formule EQUIV permet de chercher la date minimale de la tâche dans la ligne 3.

```
=EQUIV(INDEX(Taches!G1:I11;Planning!B2;3);3:3;0)
```

Parcourir la feuille Planning

L'ensemble des tâches sera parcouru pour récupérer le nombre de ressources requises par poste pour chacune des tâches.

Il convient ensuite de trier le tableau par coût en commençant par la ressource la moins coûteuse jusqu'à la ressource la plus coûteuse. Il faut parcourir ensuite l'ensemble du tableau trié, en partant de la ressource la moins coûteuse. Pour chaque ressource, le programme va parcourir ses disponibilités pour la durée de la tâche. S'il reste des ressources requises pour réaliser la tâche, il faut consommer la ressource et ajouter le coût de la ressource au coût global de la tâche.

Stocker les informations dans un tableau

Comment ne pas se perdre avec tant d'informations ? Avec un tableau. Nous stockerons dans une variable tableau les informations récupérées. Le tableau comportera deux dimensions :

La première dimension aura pour valeur 4, ce qui correspond aux postes de tâche. La seconde dimension aura pour valeur 3 et aura pour but de stocker le libellé du poste, le nombre de ressources requises et le coût total de chaque poste pour la tâche.

Visuellement, le tableau prend la forme suivante :

Poste	Nombre de ressources requises	Coût total par poste
1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3
4,1	4,2	4,3

Afficher les données en commentaire

Le coût de chaque tâche est récupéré et affiché au sein d'un commentaire contenu sur la colonne A de la feuille **Taches**. Le commentaire sera visible uniquement si l'utilisateur se positionne sur la cellule le contenant.

	A	B	C	D
1				Edition de la disponibilité
2		Calculer la durée		par personne.
3				Developpeur
4	Tache 1			
5	Tache 2			
6	Tache 3			
7	Tache 4	9	10	
8	Tache 5	6	2	
9	Tache 6	3	3	
10	Tache 7	3	0	
11	Tache 8	2	0	
12				

Le coût total du projet sera affiché en cellule G2.

Rédaction de la procédure

La procédure CalculCout est située dans le Module1 que vous retrouverez dans Visual Basic Editor. Elle est rédigée après la procédure CalculerDuree.

```

Case 28
    ValeurAttendue = NbConcepteur
Case 29
    ValeurAttendue = INDev
Case 30
    ValeurAttendue = INText
End Select
For Colonne = 3 To NbCol + 4
    Dim DateEnCours As Date
    DateEnCours = CDate(FPlanning.Cells(5, Colonne).Value)
    'On contrôle si la date de début est passée
    If DateEnCours >= DateDebut And WeekDay(DateEnCours, 2) < 6 Or EstFerie(CStr(DateEnCours)) Then
        'décompte des ressources par rapport à la valeur attendue
        ValeurAttendue = ValeurAttendue - FPlanning.Cells(Foetus, Colonne).Value
        'si la valeur attendue est inférieure ou égale à 0, on calcule la durée relative pour faire la tâche
        If ValeurAttendue <= 0 Then
            DureeRelative = WorksheetFunction.NetworkDays(DateDebut, DateEnCours, Range("Ferie"))
            'si la durée relative est supérieure à la durée maximum jusqu'à, on met à jour la durée maximum.
            If DureeRelative > DureeMax Then
                DureeMax = DureeRelative
            End If
        End If
    End If
Next
' Mise à jour de la durée
FPlanning.Cells(Ligne, 5).Value = DureeMax
End Sub
Sub CalculCout()
End Sub
    
```

⚡ Dans un premier temps, créez les variables NbRessource et NbDate qui contiendront respectivement le nombre de jours dans le calendrier et le nombre de ressources dans la feuille Planning. Ces valeurs correspondent aux cellules D1 et B1 de la feuille Planning.

```
'Sélection de la feuille Planning
Sheets("Planning").Activate
'Création des variables contenant le nombre de jours du calendrier,
et le nombre de ressource
Dim NbRessource, NbDate As Integer
'Récupération du nombre de ressources dans la feuille Planning
NbRessource = Sheets("Planning").Cells(1, 4).Value
'Récupération du nombre de dates dans la feuille Planning
NbDate = Sheets("Planning").Cells(1, 2).Value
```

- ❏ Créez ensuite les variables `LigneFinPlanning` et `ColonneFinPlanning` qui contiendront respectivement la dernière ligne et la dernière colonne de la plage contenant les disponibilités des ressources dans la feuille **Planning**. Pour la variable `LigneFinPlanning`, ajoutez le nombre de ressources à la ligne 3 (dernière ligne avant la liste des ressources). Pour la variable `ColonneFinPlanning`, ajoutez le nombre de dates à la colonne 4 (dernière colonne avant la liste des dates).

```
'Définition des variables qui seront les limites du tableau
Dim LigneFinPlanning, ColonneFinPlanning As Integer
LigneFinPlanning = 3 + NbRessource
ColonneFinPlanning = 4 + NbDate
```

- ❏ Pour effectuer le tri de la plage, sélectionnez l'ensemble de la plage, en-tête comprise, contenant les ressources et les dates dans la feuille **Planning**.

```
'Selection de la plage complète contenant le planning
Range(Cells(3, 1), Cells(LigneFinPlanning, ColonneFinPlanning)).Select
```

- ❏ Faites ensuite un tri ascendant (`Order1:=xlAscending`) sur la colonne des coûts (`Key1:=Range("D3")`) en incluant les en-têtes (`Header:=xlGuess`) grâce à la méthode `Sort` sur la plage sélectionnée.

```
'Tri ascendant sur la colonne D en prenant en compte les en-têtes
Selection.Sort Key1:=Range("D3"), Order1:=xlAscending,
Header:=xlGuess, _
OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
```

- ❏ Définissez ensuite :

- ▶ Le tableau à deux dimensions 4 et 3 contenant des champs textes (`String`);
- ▶ Les variables de type nombre entier long `CoutTotalTache` et `CoutTotalProjet` qui contiendront les cumuls des coûts de la tâche et du projet. Initialisez la variable `CoutTotalProjet` à 0 pour faire le calcul. La variable `CoutTotalTache` ne nécessite pas d'initialisation à 0.

```
'Définition des variables
'Le tableau stockant les coûts et le nombre de ressources requises
pour chaque tâche
Dim TableauCout(4, 3) As String
'Variable pour stocker les coûts
```

```
Dim CoutTotalTache, CoutTotalProjet As Long
'Initialisation de la variable du coût total
CoutTotalProjet = 0
```

- ☞ Parcourez ensuite l'ensemble des tâches avec une boucle de type For ... Next. La variable LigneTache contiendra la ligne de la feuille Taches. Affectez la valeur de la variable LigneTache à la cellule B2 de la feuille Planning. Cela permet de calculer la colonne de début et la colonne de fin de la plage à analyser. Cette plage à analyser correspond aux dates de la tâche en cours. Les valeurs de la colonne de début et de la colonne de fin sont calculées dans les cellules C2 et D2 de la feuille Planning.

```
For LigneTache = 4 To 11
  'On affecte le numéro de la tâche à cellule B2 de la feuille
  Planning => cela met à jour les cellules C2 et D2
  Sheets("Planning").Cells(2, 2).Value = LigneTache
  'Récupération la colonne de début et de fin de la plage à analyser
  Dim ColonneDebut, ColonneFin as Integer
  ColonneDebut = Sheets("Planning").Cells(2, 3).Value
  ColonneFin = Sheets("Planning").Cells(2, 4).Value
```

- ☞ Initialisez ensuite la variable TableauCout. Pour cela, faites une boucle pour parcourir les 4 index de la première dimension du tableau. Pour chaque index, renseignez le poste (seconde dimension, index 1), le nombre de ressources nécessaires (seconde dimension, index 2), et initialisez le coût de chaque tâche par poste (seconde dimension, index 3). La boucle For ... Next qui fait itérer la variable IndexPoste permet de parcourir les 4 index de la première dimension. À partir de cette variable, les colonnes de la feuille Taches (colonne 2 à 5) sont également déterminées. La variable LignePoste correspond à la variable IndexPoste + 1. La variable LignePoste prendra donc les valeurs 2 à 5.

```
'initialisation du tableau
  For IndexPoste = 1 To 4
    LignePoste = IndexPoste + 1
    'Nom du poste
    TableauCout(IndexPoste, 1) = Sheets("Taches").Cells(3,
    LignePoste).Value
    'Nombre total de ressources requises pour le poste
    TableauCout(IndexPoste, 2) =
    Sheets("Taches").Cells(LigneTache,
    LignePoste).Value
    'Initialisation du coût pour le poste
    TableauCout(IndexPoste, 3) = 0
  Next
```

- ☞ Parcourez l'ensemble des ressources et récupérez le coût de la ressource au sein d'une variable.

```
'Parcours de l'ensemble des ressources => de la moins chère à
la plus chère
For Ressource = 4 To LigneFinPlanning
  'Récupération du coût de la ressource
  CoutRessource = Sheets("Planning").Cells(Ressource, 4).Value
```

☞ Parcourez la plage entre la date de début minimale de la tâche (variable `ColonneDebut`) jusqu'à la date de fin (variable `ColonneFin`). Contrôlez si la ressource a indiqué sa disponibilité. Pour cela il faut tester si la propriété `Value` de la cellule parcourue est égale à `X`.

☞ Si la ressource a indiqué sa disponibilité, mettez à jour le tableau :

- ▶ Parcourez l'ensemble du tableau jusqu'à faire correspondre le nom du poste du tableau avec le poste de la ressource. Une fois le poste identifié, vérifiez s'il reste des ressources requises pour la tâche. Si c'est le cas, retirez une ressource au total des ressources requises et ajoutez au coût de la tâche par poste le coût de la ressource.

```
'Parcours de la durée de la tâche
For ColonnePlanning = ColonneDebut To ColonneFin
  'Contrôle sur la disponibilité de la ressource
  If Sheets("Planning").Cells(Ressource, ColonnePlanning).
Value = "X" Then
  'Recherche du poste de la ressource
  For LigneTableau = 1 To 4
    If Sheets("Planning").Cells(Ressource, 3).Value
= TableauCout(LigneTableau, 1) Then
      'S'il reste des ressources requises, alors
      comptabilisation de la ressource
      If TableauCout(LigneTableau, 2) > 0 Then
        'Retrait d'une ressource au nombre de
      ressources requises
        TableauCout(LigneTableau, 2) =
TableauCout(LigneTableau, 2) - 1
        'Ajout du coût de la ressource au coût
      total de la tâche pour le poste.
        TableauCout(LigneTableau, 3) =
CLng(TableauCout(LigneTableau, 3)) + CoutRessource
      End If
    End If
  Next LigneTableau
End If
Next ColonnePlanning
```

☞ Fermez la boucle `Ressource`.

```
Next Ressource
```

À ce stade, vous avez récupéré le coût de chaque tâche par poste.

- Additionnez ensuite les coûts des postes pour la tâche. Terminez en ajoutant le coût total de la tâche au coût total du projet.

```
'Cumul du coût de chaque poste pour définir le coût total de la
tâche
  CoutTotalTache = CLng(TableauCout(1, 3)) + CLng(TableauCout(2, 3))
+ CLng(TableauCout(3, 3)) + CLng(TableauCout(4, 3))
  'Ajout du coût de la tâche au coût total du projet
  CoutTotalProjet = CoutTotalProjet + CoutTotalTache
```

- Pour ajouter le commentaire sur la ligne de la tâche dans la feuille **Taches**, commencez par supprimer les commentaires existants avec la méthode `ClearComments`. Utilisez la méthode `AddComment` pour ajouter un nouveau commentaire.

```
'Suppression du commentaire existant
  Sheets("Taches").Cells(LigneTache, 1).ClearComments
  'Création d'un nouveau commentaire contenant
  Sheets("Taches").Cells(LigneTache, 1).AddComment
```

- Rendez le commentaire visible uniquement lorsque l'utilisateur est positionné sur la cellule en donnant la valeur `False` à la propriété `Visible` de l'objet `Comment` de la cellule en cours.

```
'Rendre le commentaire masqué, visible uniquement si
l'utilisateur se positionne sur la cellule
  Sheets("Taches").Cells(LigneTache, 1).Comment.Visible = False
```

- Écrivez le commentaire en modifiant la propriété `Text` de l'objet `Comment` de la cellule en cours. Récupérez la valeur de la variable `CoutTotalTache` que vous insérez dans le commentaire.

```
'Écrire le contenu du commentaire
  Sheets("Taches").Cells(LigneTache, 1).Comment.Text Text="Cout:"
& Chr(10) & "Cout de la tâche : " & CStr(CoutTotalTache) & " euro
```

- Fermez la boucle `LigneTache`.

```
Next LigneTache
```

- Terminez la procédure en affichant le coût total du projet dans cellule G2 de la feuille **Taches**.

```
Sheets("Taches").Cells(2, 7).Value = CoutTotalProjet
End Sub
```

Voici la procédure complète :

```
Sub CalculCout()
  Sheets("Planning").Activate
  'Création des variables contenant le nombre de jours du calendrier,
  et le nombre de ressources
  Dim NbRessource, NbDate As Integer
```

```

'Récupération du nombre de ressources dans la feuille Planning
NbRessource = Sheets("Planning").Cells(1, 4).Value
'Récupération du nombre de dates dans la feuille Planning
NbDate = Sheets("Planning").Cells(1, 2).Value
'Définition des variables qui seront les limites du tableau
Dim LigneFinPlanning, ColonneFinPlanning As Integer
LigneFinPlanning = 3 + NbRessource
ColonneFinPlanning = 4 + NbDate
'Selection de la plage complète contenant le planning
Range(Cells(3, 1), Cells(LigneFinPlanning, ColonneFinPlanning)).Select
'Tri ascendant sur la colonne D en prenant en compte les en-têtes
Selection.Sort Key1:=Range("D3"), Order1:=xlAscending,
Header:=xlGuess, _
OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom

'Définition des variables
'Le tableau stockant les coûts et le nombre de ressources requises
pour chaque tâche
Dim TableauCout(4, 3) As String
'Variable pour stocker les coûts
Dim CoutTotalTache, CoutTotalProjet As Long
'Initialisation de la variable du coût total
CoutTotalProjet = 0
'Parcours de l'ensemble des tâches du projet
For LigneTache = 4 To 11
    'affecter le numéro de la tâche à cellule B2 de la feuille
    Planning => cela met à jour les cellules C2 et D2
    Sheets("Planning").Cells(2, 2).Value = LigneTache
    'Récupération la colonne de début et de fin de la plage à analyser
    Dim ColonneDebut, ColonneFin as Integer
    ColonneDebut = Sheets("Planning").Cells(2, 3).Value
    ColonneFin = Sheets("Planning").Cells(2, 4).Value
'initialisation du tableau
    For IndexPoste = 1 To 4
        LignePoste = IndexPoste + 1
        'Nom du poste
        TableauCout(IndexPoste, 1) = Sheets("Taches").Cells(3,
LignePoste).Value
        'Nombre total de ressources requises pour le poste
        TableauCout(IndexPoste, 2) = Sheets("Taches").Cells(LigneTache,
LignePoste).Value
        'Initialisation du coût pour le poste
        TableauCout(IndexPoste, 3) = 0
    Next
    'Parcourir de l'ensemble des ressources => de la moins chère
à la plus chère
    For Ressource = 4 To LigneFinPlanning
        'Récupération du coût de la ressource

```

```

    CoutRessource = Sheets("Planning").Cells(Ressource, 4).Value
    'Parcours de la durée de la tâche
    For ColonnePlanning = ColonneDebut To ColonneFin
        'Contrôle sur la disponibilité de la ressource
        If Sheets("Planning").Cells(Ressource, ColonnePlanning).
Value = "X" Then
            'Recherche du poste de la ressource
            For LigneTableau = 1 To 4
                If Sheets("Planning").Cells(Ressource, 3).
Value = TableauCout(LigneTableau, 1) Then
                    'S'il reste des ressources requises,
alors comptabilisation de la ressource
                        If TableauCout(LigneTableau, 2) > 0 Then
                            'Retrait d'une ressource au nombre de
ressources requises
                                TableauCout(LigneTableau, 2) =
TableauCout(LigneTableau, 2) - 1
                            'Ajout du coût de la ressource au coût
total de la tâche pour le poste.
                                TableauCout(LigneTableau, 3) =
CLng(TableauCout(LigneTableau, 3)) +
CoutRessource
                            End If
                        End If
                    Next LigneTableau
                End If
            Next ColonnePlanning
        Next Ressource
        'Cumul du coût de chaque poste pour définir le coût total
de la tâche
        CoutTotalTache = CLng(TableauCout(1, 3)) + CLng(TableauCout(2, 3))
+ CLng(TableauCout(3, 3)) + CLng(TableauCout(4, 3))
        'Ajout du coût de la tâche au coût total du projet
        CoutTotalProjet = CoutTotalProjet + CoutTotalTache
        'Suppression du commentaire existant
        Sheets("Taches").Cells(LigneTache, 1).ClearComments
        'Création d'un nouveau commentaire contenant
        Sheets("Taches").Cells(LigneTache, 1).AddComment
        'Rendre le commentaire masqué, visible uniquement si l'utilisateur
se positionne sur la cellule
        Sheets("Taches").Cells(LigneTache, 1).Comment.Visible = False
        'Écrire le contenu du commentaire
        Sheets("Taches").Cells(LigneTache, 1).Comment.Text
Text:="Cout:" & Chr(10) & "Cout de la tâche : " & CStr(CoutTotalTache)
& " euros"
        Next LigneTache
        Sheets("Taches").Cells(2, 7).Value = CoutTotalProjet
    End Sub

```

Chapitre 6

Consolidation et partage de données

A. Consolidation de données diverses	283
B. Partage des données	309

A. Consolidation de données diverses

1. Description de l'exemple

a. Présentation de l'exemple

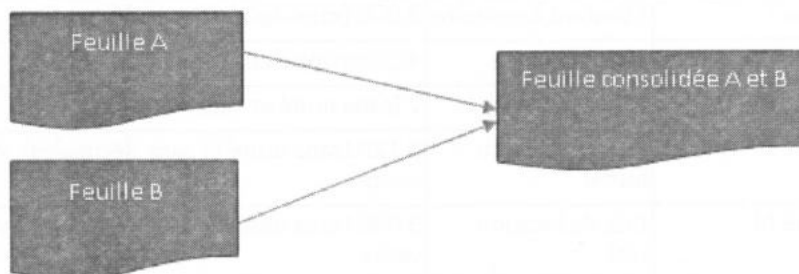
En informatique, la consolidation permet de regrouper les données provenant de différentes sources afin d'obtenir un rapport structuré.

L'exemple qui va suivre va permettre de consolider dans un premier temps plusieurs sources de données en une seule puis de travailler sur le tableau consolidé afin d'en extraire les informations clés.

Il s'agit d'un groupe immobilier composé de deux agences distinctes qui sont situées à Paris et à Reims. Bien que semblable dans l'organisation, la saisie des opérations en cours ne se fait pas de la même manière pour les deux agences. Elles utilisent cependant toutes les deux un fichier Excel retraçant leur activité.

L'objectif sera donc de consolider ces deux sources de données en un seul fichier regroupant l'ensemble des informations. L'agence immobilière souhaite également proposer un tableau récapitulatif de son activité.

Cet exemple ne propose pas d'interface utilisateur. Le résultat se présente sous la forme d'une source de données consolidées.



b. Présentation des classeurs

Cet exemple se présente avec trois classeurs distincts :

- Le classeur **Immo-Paris.xlsx** contient une feuille pour l'agence parisienne comportant les données suivantes :

Colonne Excel	Libellé	Valeur exemple
Colonne A	Date	Date de la réalisation de l'opération. Elle sera stockée sous le format Timestamp qui est très répandu en informatique : il s'agit d'un compteur numérique correspondant au nombre de secondes écoulées depuis le 1 ^e janvier 1970. Par exemple : 1 ^e janvier 2016 : 1451602800 Un des avantages de ce format est la facilité d'effectuer la comparaison de date, puisque cela revient à faire une différence entre deux nombres
Colonne B	Agent	Nom de l'agent immobilier s'occupant de la vente
Colonne C	Arrondissement	Il s'agit des arrondissements si la commune en possède.
Colonne D	Commune	Paris, Ivry-sur-Seine...
Colonne E	Code Postal	75013, 94200
Colonne F	Type de bien	Maison, Appartement, Loft, Villa
Colonne G	Prix de vente initial	100 000 (sans unité et sans décimales), vide si location
Colonne H	Prix de vente réel	120 000 (sans unité et sans décimales), vide si location
Colonne I	Montant honoraire	3 000 (sans unité et sans décimales)
Colonne J	Nombre visite	4 (sans unité et sans décimales)
Colonne K	Nombre d'offres	2 (sans unité et sans décimales)
Colonne L	Prix de location initial	3 120 (sans unité et sans décimales), vide si vente
Colonne M	Prix de location réel	3 000 (sans unité et sans décimales), vide si vente

Colonne Excel	Libellé	Valeur exemple
Colonne N	Prix au m ²	5 123,2 au m ² : prix au mètre carré avec des décimales potentielles et une unité de mesure qui est toujours le prix au mètre carré
Colonne O	Opération réussie	VRAI / FAUX
Colonne P	Durée de l'opération	25 : durée en jours entre la date de parution de l'offre et la signature

- Le classeur **Immo_Reims.xlsx** contient une feuille pour l'agence de Reims comportant les données suivantes :

Colonne Excel	Libellé	Valeur exemple
Colonne A	Agent immobilier	Nom de l'agent immobilier
Colonne B	Vente/Location	V pour vente ou L pour location
Colonne C	Honoraires en %	4,17 % : montant en pourcentage avec 2 décimales possibles
Colonne D	Prix de vente / location	200 000 € (montant avec unité sans décimales), commun pour location et vente
Colonne E	Surface	43,5 m ² (montant avec unité et décimales)
Colonne F	Type de bien	M pour maison, V pour villa, A pour appartement ou loft
Colonne G	Date de parution	15/12/2015 : date de la parution de l'offre au format JJ/MM/AAAA
Colonne H	Commune	Reims, Tinquieux : pas de mention du canton
Colonne I	Code postal	51100, 51430
Colonne J	Parking	Oui / Non
Colonne K	Piscine	Oui / Non
Colonne L	Prix de départ	210 000 € (montant avec unité sans décimales), commun pour location et vente
Colonne M	Date de signature	Date de la signature de la vente / location
Colonne N	Statut opération	En cours, Réussie, Echouée
Colonne O	Visite	5 : nombre de visites
Colonne P	Offre	3 : nombre d'offres

La direction de la société d'agences a réalisé un fichier unique. C'est ce fichier qui contiendra les données consolidées.

► Les données consolidées :

Colonne Excel	Libellé	Valeur exemple
Colonne A	Agence	Reims ou Paris
Colonne B	Agent immobilier	Nom de l'agent immobilier
Colonne C	Type d'opération	Vente ou Location
Colonne D	Date de parution de l'offre	01/01/2016 : format JJ/MM/AAAA
Colonne E	Date de signature	31/12/2016 : format JJ/MM/AAAA
Colonne F	Type de bien	Maison, Loft, Villa, Appartement
Colonne G	Surface	45 m ² (avec unité m ² et sans décimales)
Colonne H	Commune	Nom de la ville : Paris, Reims, Ivry-sur-Seine, Tinquex
Colonne I	Code postal	75013, 51100, 94200, 51430
Colonne J	Avec parking	VRAI / FAUX / ou Non renseigné
Colonne K	Prix proposé	210 000 € (montant avec unité sans décimales), commun pour location et vente : Prix auquel le bien est proposé.
Colonne L	Prix final	200 000 € (montant avec unité sans décimales) : commun pour location et vente : Prix auquel le bien est vendu / loué.
Colonne M	Honoraires	4 000 € (montant avec unité sans décimales)
Colonne N	Nombre de visite	10 : entier numérique
Colonne O	Nombre d'offres	7 : entier numérique
Colonne P	Opération réussie	VRAI / FAUX

c. Fonctionnalités

Dans cet exemple, une seule fonctionnalité sera proposée à savoir la mutualisation et consolidation des données des deux agences (fichiers **Immo_Reims.xlsx** et **Immo_Paris.xlsx**) au sein d'un même tableau (**Enoncé_6-A.xlsm**).

Pour déclencher la fonctionnalité, une macro sera lancée par l'utilisateur. Elle a vocation à être utilisée une seule fois pour cet import.

Le tableau des indicateurs de l'activité des agences sera réalisé une fois les données importées dans la seconde partie du chapitre.

2. Notions de cours

a. Manipulation de feuilles et classeurs

La manipulation de feuilles et classeurs consiste à manipuler des variables objets.

Gestion de l'application Excel

L'import va consister à ouvrir des classeurs Excel (Reims et Paris) puis à sélectionner les feuilles nécessaires. Toutes ces informations seront stockées dans des variables de type objet.

```
Dim ExcApp As Excel.Application 'Variable de gestion de l'application
Dim WB As Excel.Workbook 'Variable classeur
Dim WS As Excel.Worksheet 'Variable feuille de calcul
```



La gestion de l'application n'est pas nécessaire si l'application est déjà ouverte. En revanche, le même mécanisme s'applique pour manipuler d'autres applications comme PowerPoint.

b. Sélection et ouverture d'un classeur Excel

Méthode GetOpenFileName

Pour la sélection du fichier Excel, il est possible d'utiliser la méthode `GetOpenFileName` de la classe `Application` qui permet l'ouverture d'une fenêtre de sélection de fichier. Cette méthode renvoie le chemin de l'application sélectionnée, et il peut être utilisé pour ouvrir le fichier sélectionné.

```
Nom_Fichier = Application.GetOpenFilename("Fichiers Excel (*.xlsm),
*.xlsm") 'filtre sur les fichiers Excel
If Nom_Fichier <> False Then
'Ouverture fichier
Else
Msgbox ("Fichier non sélectionné")
End if
```

Méthode FileDialog

Toutefois, l'exemple proposé utilisera une autre méthode plus complète `Application.FileDialog` qui permet tout type d'échange avec les répertoires et fichiers :

En fonction de l'argument associé à la méthode `FileDialog`, la boîte de dialogue prendra une forme différente :

- ▶ Argument `msoFileDialogFilePicker` : sélection de fichier.
- ▶ Argument `msoFileDialogFolderPicker` : sélection de dossier.
- ▶ Argument `msoFileDialogOpen` : ouverture de fichier.
- ▶ Argument `msoFileDialogSaveAs` : enregistrement de fichier.

Par exemple :

```
With Application.FileDialog(msoFileDialogFolderPicker)
'Définition de la boîte de dialogue de sélection d'un répertoire.
    .Title = "Sélectionnez le répertoire" 'Ajout d'un titre à
la boîte de dialogue
    .Show 'Affichage de la fenêtre
If .SelectedItems.Count > 0 Then
    MsgBox .SelectedItems(1) 'Affichage du premier élément sélectionné.
End If
```



Cette méthode offre plus de possibilités, c'est pour cela qu'il est plus intéressant de l'apprendre.

c. Les boucles

Les boucles vont permettre ici de parcourir les données des deux feuilles sources. La boucle fait évoluer le numéro de la ligne à chaque itération. Nous utiliserons la boucle de type `While ... Wend` avec comme condition de sortie la cellule vide.

```
Dim Ligne as Integer
Ligne = 2 'Définition de la valeur de la première ligne
While Cells(Ligne, 1).value <> "" 'Tant que la cellule n'est pas vide,
on continue la boucle.
Ligne = Ligne +1 'Itération de la variable
Wend 'Retour au début de la boucle
```

d. Format de la cellule

Afin de modifier le format de la cellule via VBA, il faut modifier la propriété de la cellule nommée `NumberFormat`. Voici quelques exemples de valeurs que peut prendre la propriété `NumberFormat` de la classe `Range`.

Format	NumberFormat
Standard	General
Nombre	0
Monétaire	# ##0.00 €
Date	@dd/mm/yyyy
Heure	[\$-F400]h:mm:ss
Pourcentage	0.00%
Fraction	#?/?
Scientifique	0.00E+00
Texte	@

e. Formule Excel dans le code VBA

Utiliser une formule Excel dans VBA

Comme certaines fonctions existent déjà dans Excel, il n'est pas nécessaire de les recréer dans le code. Par exemple, la fonction comptant le nombre de valeurs dans une plage existe déjà.

Dans Excel :

```
=NB (A1:A10)
```

Équivalent avec VBA :

```
Var = Application.WorksheetFunction.Count (Range ("A1:A10"))
```

La principale différence est la langue utilisée. En effet, ces fonctions sont toujours écrites en anglais et les arguments ne sont pas décrits comme dans Excel.

Propriété Formule d'une cellule

La cellule est un objet à part entière et elle contient ses propres propriétés. L'une des propriétés de la cellule est donc la formule. La formule est celle qui apparaît dans les cellules et démarre par le signe =, comme par exemple :

```
=SOMME (A1:A30)
```

Il est possible de saisir la formule directement via VBA, mais à un delta près puisque la formule est en anglais.

Ce changement est mineur car il est très facile de trouver l'équivalent d'une formule en français, en anglais. En revanche, il sera nécessaire d'adapter deux éléments de syntaxe :

- ▶ Le point-virgule séparant les arguments devient une virgule.
- ▶ La virgule qui sépare les décimales du reste du nombre devient un point.

Dans Excel :

```
=RECHERCHEV(A1;C4:E10;3;FAUX)
=ARRONDI.AU.MULTIPLE(A1 ;0,1)
```

Équivalent de la propriété Formule :

```
Cells(1,1).Formula = "=VLOOKUP(A1,C4:E10,3,False)"
Cells(2,1).Formula = "=MROUND(A1,0.1)"
```

f. Select Case et structure conditionnelle

L'instruction `Select Case` permet de définir différents scénarios pour une expression donnée en entrée. En fonction de la valeur de l'expression, une instruction sera attribuée. Cette structure simplifie l'usage d'une structure conditionnelle.

Par exemple, le test de la variable numérique `Var` est :

```
Select Case Var 'Test de la variable Var
Case 1 'Cas où la valeur de Var est 1
Msgbox("Var est égal à 1" ) 'Dans le cas ou Var est égale à 1 on lit
le groupe d'instruction en dessous du Case correspondant
Msgbox("2eme instruction") 'Ce n'est pas limité à 1 seule instruction
Code 0, 2, 3 'Cas où la Var est égale à 0, 2 ou 3.
'Code
Case 4 to 10 'Cas où la valeur est comprise entre 4 et 10
'Code
Case Is > 10 'Cas où la valeur est supérieure à 10
'Code
Case Else 'autres cas qui ne remplissent pas les conditions précédentes.
'Code
End Select 'Fin de l'instruction.
```

La structure conditionnelle permet d'enchaîner les conditions et d'associer des instructions à la réussite de la condition. Certes, ce code est plus lourd que `Select Case` mais il a l'avantage de dissocier chacune des conditions testées :

```
If Var = 1 Then
'Code
Elseif Var2 = 1 Then
'Code
Elseif Var3 And IsNumeric(Var4) Then
'Code
```

```
Else 'Cas où aucune condition n'est remplie.  
'Code  
End if
```

Quand utiliser `Select Case` ou `If ... ElseIf ... End If` ?

La structure **Select Case** aura pour objet de simplifier un code un peu lourd, notamment pour tester N fois la même valeur dans une condition. Par exemple, un code du type :

```
If Var = 1 then  
'Code  
ElseIf Var > 2 and Var <4 then  
'Code  
ElseIf Var = 12 then  
...  
End if
```

Il est plus intéressant d'utiliser **Select Case** car les différentes instructions sont affectées en fonction de la valeur de `Var`. En revanche, lorsque la valeur testée dans la condition n'est pas toujours la même, il convient d'utiliser plutôt une structure conditionnelle. Il sera aussi possible d'utiliser les instructions **And** (Et) et **Or** (Ou) dans la condition du test.

3. Réalisation de l'exemple

- 🔗 Ouvrez le fichier `Enoncé_6-A.xlsm` qui contiendra les données consolidées. Les fichiers `Immo_Paris.xlsx` et `Immo_Reims.xlsx` seront utilisés dans l'import, mais ils ne seront pas ouverts dans l'exemple.

a. Structure du code

Le code va être réalisé au sein d'une seule et unique procédure qui sera utilisée une seule fois pour récupérer les données des deux feuilles.

Il faut donc créer une procédure qui stockera l'ensemble du traitement et sera nommée `InsertionDonnees`.

- 🔗 Insérez un module et saisissez les lignes de code suivantes :

```
Sub InsertionDonnees  
'Le code sera inséré ici  
End sub
```

b. Déclaration des variables feuille et classeur

Pour réaliser cet exemple, il va falloir dans un premier temps créer les variables de type objet pour stocker et manipuler les feuilles et classeurs.

Il sera nécessaire d'avoir six variables :

- ▶ Trois variables de types Classeur :
 - ▶ Fichier Consolidé,
 - ▶ Fichier Reims,
 - ▶ Fichier Paris.
- ▶ Trois variables de types Feuille :
 - ▶ Feuille Consolidée : au sein du fichier consolidé `Enoncé_6-A.xlsx`, feuille contenant les données consolidées.
 - ▶ Feuille Reims : au sein du fichier `Immo_Reims.xlsx`, feuille contenant les données du fichier Reims.
 - ▶ Feuille Paris : au sein du fichier `Immo_Paris.xlsx`, feuille contenant les données du fichier Paris.

☞ Déclarez les variables de la manière suivante :

```
'Définition des variables
Dim WBreims As Excel.Workbook
Dim WBParis As Excel.Workbook
Dim WBFinal As Excel.Workbook
Dim WSReims As Excel.Worksheet
Dim WSParis As Excel.Worksheet
Dim WSFinal As Excel.Worksheet
```



La déclaration de variables peut être regroupée en une seule ligne par type de variable. Les noms de variables doivent être séparés par une virgule :

```
Dim WBFinal, WBreims, WBParis As Excel.Workbook
Dim WSFinal, WSReims, WSParis As Excel.Worksheet
```

Pour affecter des valeurs aux variables, il est nécessaire d'utiliser le mot clé **Set** qui permet d'assigner une référence à l'objet. Dans ce cas, `WBFinal` aura pour référence le classeur actuel : `ThisWorkbook`.

Enfin, la feuille `WSFinal` sera la feuille **Données** de `WBFinal`.

☞ Saisissez le code suivant :

```
Set WBFinal = ThisWorkbook
Set WSFinal = WBFinal.Sheets("Donnees")
```

c. Définition de la boîte de dialogue d'ouverture de fichier

La première étape consiste à ouvrir une boîte de dialogue pour sélectionner les deux fichiers à importer.

Pour définir les caractéristiques de la boîte de dialogue, il faut appeler la méthode `FileDialog` de la classe `Application` avec comme argument `msoFileDialogFilePicker` pour préciser qu'il s'agit d'une sélection de fichier.

Dans un premier temps, il faut créer l'objet correspondant à cette boîte de dialogue puis en définir ses caractéristiques.

Nous allons stocker en variable de fichier de Reims.

Pour simplifier le code, utilisez la structure `With ... End with` afin de ne pas réécrire systématiquement le code de boîte de dialogue :

```
With Application.FileDialog(msoFileDialogFilePicker)
End with
```

Ensuite, définissez les caractéristiques de la boîte de dialogue comme son titre, le type de fichier accepté, la possibilité de sélectionner un ou plusieurs fichiers.

```
'Définit un titre pour la boîte de dialogue
.Title = "Choisir le fichier Excel pour Reims:"
'Autorise la multi-sélection : non dans notre cas
.AllowMultiSelect = False
'Définit un nom de fichier par défaut
.InitialFileName = "Immo_Reims.xlsx"
'Efface les filtres existants.
.Filters.Clear
'Définit une liste de filtres pour le champ "Type de fichiers".
.Filters.Add "Classeurs Excel", "*.xls; *.xlsx; *.xlsm"
'Définit le numéro du filtre qui s'affiche par défaut dans le champ
"Type de fichiers" : ici il n'y a qu'un seul filtre, l'index n'est
donc pas nécessaire.
.FilterIndex = 1
```

La prochaine étape consiste à afficher la boîte de dialogue.

☞ Pour cela, appelez la méthode `Show` de `FileDialog`.

```
'Affiche la boîte de dialogue
.Show
```

La dernière étape consiste à récupérer le résultat de la sélection du fichier dans la boîte de dialogue. Pour cela, il est nécessaire de tester s'il y a bien des fichiers sélectionnés avec la propriété `SelectedItem.Count`. Cela permettra de compter le nombre de fichiers sélectionnés.

S'il existe au moins un élément sélectionné, une référence sera affectée aux variables `WSReims` et `WBReims` avec les informations récupérées dans la boîte de dialogue. Dans le cas où aucun élément n'est sélectionné, l'instruction `Exit Sub` permettra de quitter la procédure.

☞ Saisissez le code suivant :

```
If .SelectedItems.Count > 0 Then
'Affecte à la variable workbook le fichier sélectionné qui sera ouvert
dans l'application.
Set WBReims = Workbooks.Open(.SelectedItems(1))
'Affecte à la variable worksheet la première feuille du classeur
définie précédemment.
Set WSReims = WBReims.Sheets(1)
Else
'Si aucun fichier n'est sélectionné, quitter la procédure
MsgBox "Vous n'avez pas sélectionné de fichiers, veuillez recommencer."
Exit Sub
End If
```

☞ Appliquez le même code pour sélectionner le fichier de Paris.

```
With Application.FileDialog(msoFileDialogFilePicker)
.Title = "Choisir le fichier Excel pour Paris:"
.AllowMultiSelect = False
.InitialFileName = "Immo_Paris.xlsx"
.Filters.Clear
.Filters.Add "Classeurs Excel", "*.xls; *.xlsx; *.xlsm"
.FilterIndex = 1
.Show
If .SelectedItems.Count > 0 Then
Set WBParis = Workbooks.Open(.SelectedItems(1))
Set WSParis = WBParis.Sheets(1)
End If
End With
```

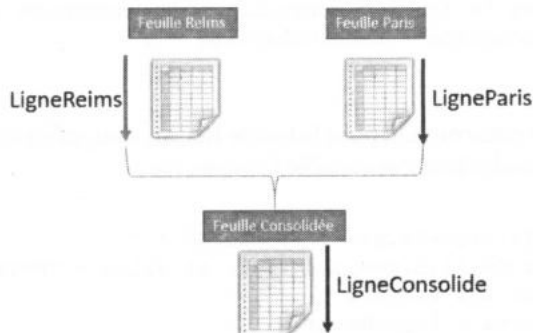


Attention, ce code ne gère pas le cas où l'utilisateur ne sélectionne pas le bon fichier ou qu'il ne sélectionne pas de fichier et qu'il ferme la boîte de dialogue.

d. Parcours des feuilles

Toutes les lignes des feuilles `Reims` et `Paris` vont être parcourues.

Le numéro de ligne va être stocké dans une variable et cette variable va être itérée pour parcourir chaque ligne. De même, une autre variable va être utilisée pour stocker la valeur de la ligne de la feuille `Consolidée`. Cette variable va également être itérée à chaque fois qu'une ligne va être insérée dans le fichier consolidé.



Déclaration des variables

- ☞ Tout d'abord, déclarez les variables au format nombre entier puis affectez-leur une valeur.

```
'Définition des variables
Dim LigneParis, LigneReims, LigneConsolide As Integer
'Affectation des valeurs
LigneParis = 2 'le nombre 2 correspond à la première ligne parcourue
dans le fichier
LigneReims = 2
LigneConsolide = 2
```

Ensuite, une boucle va parcourir chacune des lignes de la feuille de données jusqu'à rencontrer une cellule vide et stopper la boucle. Pour chaque ligne rencontrée dans les fichiers de données de Reims ou de Paris, celle-ci sera ajoutée dans le fichier consolidé au format attendu.

Structure d'une boucle

- ☞ Écrivez la boucle de la manière suivante :

```
'Mise en place de la structure With ... End With pour ne pas avoir à
réécrire à chaque fois le nom de la feuille WSParis
With WSParis
'Création d'une boucle dont la condition de sortie est la rencontre
d'une cellule vide
While .Cells(LigneParis, 1).Value <> ""
'La cellule de colonne 1 dans le fichier consolidé prendra la valeur
Paris quelle que soit la valeur de LigneConsolide.
WSFinal.Cells(LigneConsolide, 1).Value = "Paris"
'La ligne de la feuille Paris va être adaptée pour s'insérer dans
le fichier consolidé.
'Itération de la variable LigneParis afin d'explorer la ligne suivante
LigneParis = LigneParis + 1
'Itération de la variable LigneConsolide afin que la ligne de
```

destination dans le fichier consolidé soit augmentée de 1.

```

    LigneConsolide = LigneConsolide + 1
Wend
End With

```

- Utilisez la même construction pour la feuille **Reims**, sauf qu'au lieu d'itérer la variable `LigneParis`, il faudra itérer la variable `LigneReims`.

```

With WSReims
    While .Cells(LigneReims, 1).Value <> ""
        WSFinal.Cells(LigneConsolide, 1).Value = "Reims"
' récupération des valeurs
        LigneReims = LigneReims + 1
        LigneConsolide = LigneConsolide + 1
    Wend
End With

```

Correspondance des valeurs

La feuille **Consolidée** va hériter des valeurs de la feuille **Paris**. Ces valeurs peuvent être récupérées à l'identique ou transformées pour correspondre au format attendu dans le format de la cellule de la feuille **Consolidée**.

Certaines données sont reprises à l'identique (pas de modification du contenu ou du format) :

- Reprenez les valeurs de la cellule de la feuille **Paris** ;
- Affectez cette valeur à la cellule de la feuille **Consolidée**.

Voici les correspondances :

Colonne dans la feuille consolidée	Colonne dans la feuille Paris	Contenu de la cellule
B (2)	B (2)	Agent immobilier
F (6)	F (6)	Type de bien
H (8)	D (4)	Commune
I (9)	E (5)	Code postal
M (13)	I (9)	Honoraires
N (14)	J (10)	Nombre de visite
O (15)	K (11)	Nombre d'offre
P (16)	O (15)	Opération réussie (VRAI/FAUX)

☞ Insérez le code suivant :

```
WSFinal.Cells(LigneConsolide, 2).Value = .Cells(LigneParis, 2).Value
WSFinal.Cells(LigneConsolide, 6).Value = .Cells(LigneParis, 6).Value
WSFinal.Cells(LigneConsolide, 8).Value = .Cells(LigneParis, 4).Value
WSFinal.Cells(LigneConsolide, 9).Value = .Cells(LigneParis, 5).Value
WSFinal.Cells(LigneConsolide, 13).Value = .Cells(LigneParis, 9).Value
WSFinal.Cells(LigneConsolide, 14).Value = .Cells(LigneParis,
10).Value
WSFinal.Cells(LigneConsolide, 15).Value = .Cells(LigneParis,
11).Value
WSFinal.Cells(LigneConsolide, 16).Value = .Cells(LigneParis,
15).Value
'L'information de la présence d'un parking n'est pas présente dans la
feuille Paris.
WSFinal.Cells(LigneConsolide, 10).Value = "Non renseigné"
```

Cas Location/Vente :

Sur la feuille **Paris**, les ventes et les locations sont clairement distinguées puisqu'elles ont leurs propres cellules pour afficher les informations.

Par exemple, le prix de la location et le prix de vente ne seront pas stockés dans une même cellule. Pour autant, sur la feuille **Consolidée**, les prix sont centralisés au sein d'une même cellule qu'il s'agisse d'une vente ou d'une location.

Une autre colonne contiendra le type d'opération (colonne C) pour savoir s'il s'agit d'une vente ou d'une location.

Le code va se baser sur deux conditions, soit la cellule de la feuille **Paris** possède un **Prix de vente réel** ce qui signifie que l'opération en cours porte sur une **vente**, soit l'opération en cours porte sur une **location**.

☞ Saisissez le code suivant :

```
If .Cells(LigneParis, 7).Value = "" Then
WSFinal.Cells(LigneConsolide, 3).Value = "Location"
'récupération des valeurs
Else
WSFinal.Cells(LigneConsolide, 3).Value = "Vente"
'récupération des valeurs
End If
```

Les prix proposés et prix finaux vont être récupérés dans les cellules correspondant à leur situation. Les prix proposés et prix final seront stockés au sein de variables. Leur valeur sera assignée à la cellule cible dans la feuille **Consolidée**.

La cellule cible est déterminée dans le tableau ci-dessous.

	Cas d'une vente	Cas d'une location
Prix proposé	Colonne G (7)	Colonne L (12)
Prix final	Colonne H (8)	Colonne M (13)

Surface du bien

Parmi les données attendues dans la feuille **Consolidée**, la surface du bien n'est pas présente dans la feuille **Paris**.

Pour calculer la surface du bien, le prix de vente est divisé par le prix au mètre carré. La surface du bien sera arrondie à l'unité via la formule d'arrondi au multiple.

☞ Appliquez la formule d'arrondi au multiple de la manière suivante :

```
Var = Application.WorksheetFunction.MRound(Arg1, Arg2) 'Arg1
correspond au nombre à arrondir, Arg2 correspond au multiple de
l'arrondi.
```

```
Dim Surf As Double
Dim Propose, Compromis As Long
If .Cells(LigneParis, 7).Value = "" Then
    WSFinal.Cells(LigneConsolide, 3).Value = "Location"
    'La surface de location correspond au prix de location divisé
    par le prix au m². La fonction Excel MROUND (arrondi au multiple) est
    appliquée au calcul pour obtenir un nombre entier.
    Surf = Application.WorksheetFunction.MRound((.Cells(LigneParis,
    13).
    Value / .Cells(LigneParis, 14).Value), 1)
    'récupération du prix de location proposé initialement
    Propose = .Cells(LigneParis, 12).Value
    'récupération du prix de location
    PrixFinal = .Cells(LigneParis, 13).Value
Else
    'Le même raisonnement que ci-dessus s'applique.
    WSFinal.Cells(LigneConsolide, 3).Value = "Vente"
    Surf = Application.WorksheetFunction.MRound((.Cells(LigneParis,
    8).
    Value / .Cells(LigneParis, 14).Value), 1)
    Propose = .Cells(LigneParis, 7).Value
    PrixFinal = .Cells(LigneParis, 8).Value
End If
'Une fois les valeurs récupérées, elles sont affichées dans la feuille
Consolidée.
WSFinal.Cells(LigneConsolide, 7).Value = Surf
```

```
WSFinal.Cells(LigneConsolide, 11).Value = Propose
WSFinal.Cells(LigneConsolide, 12).Value = PrixFinal
```

Date au format TimeStamp

Comme expliqué précédemment, le format **TimeStamp** est un format de stockage de date régulièrement utilisé qui correspond au nombre de secondes écoulées depuis le 1^e janvier 1970.

Par exemple :

Il s'est écoulé 1451602800 secondes au 1^e janvier 2016 à minuit depuis le 1^e janvier 1970. Au format Excel, il s'agit du nombre de jours écoulés depuis le 1^e janvier 1900. (Source : <http://www.timestamp.fr/>).

Lorsqu'une fonction spécifique est réalisée dans le code et que celle-ci peut être réutilisée, il est préférable de créer une fonction publique accessible à tout moment dans l'application. Cette fonction sera nommée ici `ConvTS` et aura pour paramètre la variable `Num` qui correspond à la valeur en **TimeStamp** à convertir en date Excel.

☞ Au sein du module, déclarez la fonction en dehors de la procédure InsertionDonnées :

```
Function ConvTS(Num As Long)
'Contenu de la fonction
End Function
```

Pour convertir une valeur de type **TimeStamp** en date Excel, voici l'opération à entreprendre :

- ☞ Convertissez la valeur **TimeStamp** en jours ;
- ☞ Ajoutez à ce total la valeur chiffrée au format Excel du nombre de jours écoulés entre le 1^e janvier 1900 et le 1^e janvier 1970.

Il va donc être nécessaire de transposer ces opérations en code VBA :

```
Function ConvTS(Num As Long)
'diviser la valeur par 24h, 60 minutes et 60 secondes pour convertir
en jour
ConvTS = Int(Num / 24 / 60 / 60)
'ajouter la valeur du 1er janvier 1970 soit 25569
ConvTS = ConvTS + 25569
End Function
```



*Pour connaître la valeur chiffrée d'une date Excel ou la valeur date d'un chiffre, sélectionnez la cellule contenant l'information à convertir et dans l'onglet **Accueil**, choisissez le format nombre (ou date en fonction de ce que vous voulez faire).*

- ☞ Appelez la fonction dans l'exemple afin de convertir la date en Timestamp dans la colonne A de la feuille Paris, en date classique.

```
'Appel de la fonction ConvTS
WSFinal.Cells(LigneConsolide, 4).Value = ConvTS(.Cells(LigneParis,
1).Value)
```

Attribuer un format à la cellule

Pour attribuer un format à une cellule, il faut utiliser la propriété `NumberFormat` de l'objet `Range` (ici une cellule). Dans le cadre de cet exemple, deux formats vont être employés :

- ▶ Format date : "dd/mm/yyyy;@" qui correspond à un format **Date courte**.
- ▶ Format monétaire : "#,##0.00 €" qui correspond à un format **numérique monétaire**.

- ☞ Dans un premier temps, appliquez ce format à la date de parution de l'offre.

```
WSFinal.Cells(LigneConsolide, 4).NumberFormat = "dd/mm/yyyy;@"
```

Ce format permet de traduire une valeur chiffrée en date.

La date de signature (colonne C) correspond à la date de parution de l'offre à laquelle est ajoutée la durée de l'opération, ces deux informations sont disponibles sur la feuille Paris, mais la date de signature n'est pas directement présente.

- ☞ Saisissez le code ainsi :

```
'Calcul de la valeur
WSFinal.Cells(LigneConsolide, 5).Value =
WSFinal.Cells(LigneConsolide, 4).Value + .Cells(LigneParis, 16).Value
'Modification du format
WSFinal.Cells(LigneConsolide, 5).NumberFormat = "dd/mm/yyyy;@"
```

- ☞ Appliquez ce format pour les valeurs numériques restantes :

```
WSFinal.Cells(LigneConsolide, 11).NumberFormat = "#,##0.00 €"
WSFinal.Cells(LigneConsolide, 12).NumberFormat = "#,##0.00 €"
WSFinal.Cells(LigneConsolide, 13).NumberFormat = "#,##0.00 €"
```

Ainsi toutes les données de la feuille Paris sont importées dans la feuille Consolidée.

Feuille Reims

L'import de la feuille Reims se fait après la fin de la boucle de la feuille Paris. Au terme de l'import de la feuille Paris, la première ligne vide de la feuille Consolidée est celle de la variable `LigneConsolide`. Il faut donc démarrer une nouvelle boucle pour parcourir les lignes de la feuille Reims.

Comme pour la feuille Paris, le code est à insérer au sein de la boucle.

- ☞ Saisissez cette portion de code dans la procédure `InsertionDonnees` après le code portant sur l'import de la feuille `Immo_Paris`.

```
With WSReims
    While .Cells(LigneReims, 1).Value <> ""
        WSFinal.Cells(LigneConsolide, 1).Value = "Reims"
        'Le code de récupération des données est à mettre ici.
        LigneReims = LigneReims + 1
        LigneConsolide = LigneConsolide + 1
    Wend
End With
```

Importer la feuille `Reims` paraît plus simple car à l'instar de la feuille `Paris`, certaines données sont donc reprises à l'identique (pas de modification du contenu ou du format).

Il faut :

- ▶ reprendre les valeurs de la cellule de la feuille `Reims` ;
- ▶ affecter cette valeur à la cellule de la feuille `Consolidée`.

Voici les correspondances :

Colonne dans la feuille consolidée	Colonne dans la feuille Reims	Contenu de la cellule
B (2)	A (1)	Agent immobilier
D (4)	G (7)	Date de parution
E (5)	M (13)	Date de signature
H (8)	D (4)	Commune
I (9)	I (9)	Code postal
J (10)	J (10)	Parking
K (11)	D (4)	Prix proposé
L (12)	L (12)	Prix final
N (14)	O (15)	Nombre de visites
O (15)	P (16)	Nombre d'offres
P (16)	N (14)	Opération réussie (VRAI/FAUX)

- ☞ Insérez le code suivant :

```
WSFinal.Cells(LigneConsolide, 2).Value = .Cells(LigneReims, 1).Value
WSFinal.Cells(LigneConsolide, 4).Value = .Cells(LigneReims, 7).Value
WSFinal.Cells(LigneConsolide, 5).Value = .Cells(LigneReims, 13).Value
WSFinal.Cells(LigneConsolide, 8).Value = .Cells(LigneReims, 8).Value
WSFinal.Cells(LigneConsolide, 9).Value = .Cells(LigneReims, 9).Value
```

```

WSFinal.Cells(LigneConsolide, 10).Value = .Cells(LigneReims, 10).Value
WSFinal.Cells(LigneConsolide, 11).Value = .Cells(LigneReims, 4).Value
WSFinal.Cells(LigneConsolide, 12).Value = .Cells(LigneReims, 12).Value
WSFinal.Cells(LigneConsolide, 14).Value = .Cells(LigneReims, 15).Value
WSFinal.Cells(LigneConsolide, 15).Value = .Cells(LigneReims, 16).Value
WSFinal.Cells(LigneConsolide, 16).Value = .Cells(LigneReims, 14).Value

```

Pour le type d'opération de la feuille **Consolidée** (colonne 3), il suffit d'interpréter la cellule type d'opération de la feuille **Reims** (colonne 2). En effet, la cellule prend la valeur V pour vente et L pour location.

✎ Insérez le code suivant :

```

'Situation de vente
If .Cells(LigneReims, 2).Value = "L" Then
    WSFinal.Cells(LigneConsolide, 3).Value = "Location"
Else
    WSFinal.Cells(LigneConsolide, 3).Value = "Vente"
End If

```



Le code aurait pu être rédigé de manière différente comme par exemple :

- ▶ Poser la condition sur la valeur V au lieu de L, dans ce cas les résultats de condition auraient été inversés.
- ▶ Effectuer un *Select Case*.
- ▶ Employer la syntaxe *Else If* pour tester si la valeur est bien égale à V. En effet, un simple *Else* ne vérifie pas la valeur de la colonne 2 de la feuille **Reims**.

Pour le type de bien, il faut faire la correspondance entre les valeurs de la feuille **Reims** et les valeurs attendues dans la feuille **Consolidée**.

Feuille Reims	Feuille Consolidée
A	Appartement
M	Maison
V	Villa

Deux possibilités s'offrent pour traiter ce cas :

- ▶ Structure conditionnelle (*If ... Then ... ElseIf ... Else ... End If*)
- ▶ *Select Case*

Comme vu dans la partie *Notions de cours* de ce chapitre, la structure **Select Case** semble plus appropriée car elle est plus légère à rédiger quand la condition porte uniquement sur la valeur d'un champ précis, comme c'est le cas ici.

☞ Utilisez la structure **Select Case** :

```
Select Case .Cells(LigneReims, 6).Value
    Case "A"
        WSFinal.Cells(LigneConsolide, 6).Value = "Appartement"
    Case "V"
        WSFinal.Cells(LigneConsolide, 6).Value = "Villa"
    Case "M"
        WSFinal.Cells(LigneConsolide, 6).Value = "Maison"
End Select
```

Découpage de chaîne de caractères

Le format de la surface dans la feuille **Reims** contient deux décimales, or la feuille **Consolidée** attend un nombre entier. Avec l'information « m² » positionnée en fin de champ, il n'est donc pas possible de faire une simple conversion de format, il est nécessaire de « découper » la valeur de la cellule.

Comment transformer 123.45 m² en 123 ?

- ▶ Récupérez la valeur sans l'unité ;
- ▶ Convertissez la valeur récupérée en nombre entier.

Pour appliquer cette manipulation il convient de récupérer la valeur chiffrée du champ : la valeur chiffrée correspond à l'ensemble du champ moins les 3 derniers caractères qui sont « m² ».

Par conséquent nous appliquons l'instruction **Left** avec comme argument le champ, et qui a pour taille la longueur du champ, définie par l'instruction **Len** moins le nombre de caractères à retirer.

```
ValChiffée = Left(Champ, Len(Champ)-3)
```

☞ Saisissez le code VBA :

```
WSFinal.Cells(LigneConsolide, 7).Value = CInt(Left(.Cells(LigneReims, 5).Value, Len(.Cells(LigneReims, 5).Value) - 3))
```

- ☞ Calculez les honoraires par simple multiplication entre le pourcentage des honoraires et le prix de la vente/location récupérée.

```
'Calcul des honoraires
WSFinal.Cells(LigneConsolide, 13).Value = .Cells(LigneReims, 12).Value * .Cells(LigneReims, 3).Value
```

- ☞ Enfin, dans la dernière étape de l'importation affectez le bon format aux cellules contenant des prix.

```
WSFinal.Cells(LigneConsolide, 11).NumberFormat = "#,##0.00 €"
WSFinal.Cells(LigneConsolide, 12).NumberFormat = "#,##0.00 €"
WSFinal.Cells(LigneConsolide, 13).NumberFormat = "#,##0.00 €"
```

Terminer l'opération

Pour terminer l'opération :

- ❖ Fermez les deux classeurs ouverts avec la méthode `Close` de l'objet `Workbook`.
- ❖ Affichez un message de confirmation avec une instruction `MsgBox`.

Le code se rédige donc ainsi :

```
'Fermeture des classeurs ouverts
WBParis.Close
WBReims.Close
'Message pour annoncer la fin du traitement
MsgBox ("Traitement terminé, les données ont été correctement
importées.")
```

Déclenchement de la procédure

Afin de déclencher la procédure, le plus simple est de lancer la macro via la commande appropriée :

- ❖ Allez sur l'onglet **Développeur**.
- ❖ Cliquez sur **Macro**.
- ❖ Dans la fenêtre, choisissez la macro à déclencher (ici `InsertionDonnées`).

Récapitulatif du code

La procédure terminée, voici le code une fois finalisé :

```
Sub InsertionDonnees()
'définition des variables
Dim WBReims As Excel.Workbook
Dim WBParis As Excel.Workbook
Dim WBFinal As Excel.Workbook
Dim WSReims As Excel.Worksheet
Dim WSParis As Excel.Worksheet
Dim WSFinal As Excel.Worksheet
'Affectation des valeurs du classeur consolidé
Set WBFinal = ThisWorkbook
Set WSFinal = WBFinal.Sheets("Donnees")

'Ouverture du fichier Reims
'création d'un objet boîte de dialogue avec pour argument
la sélection de fichier.
With Application.FileDialog(msoFileDialogFilePicker)
'Définit un titre pour la boîte de dialogue
.Title = "Choisir le fichier Excel pour Reims:"
'Autorise la multi-sélection : non dans notre cas
.AllowMultiSelect = False
'Définit un nom de fichier par défaut
```

```
.InitialFileName = "Immo_Reims.xlsx"
'Efface les filtres existants.
.Filters.Clear
'Définit une liste de filtres pour le champ "Type de fichiers".
.Filters.Add "Classeurs Excel", "*.xls; *.xlsx; *.xlsm"
'Définit le filtre qui s'affiche par défaut dans le champ
"Type de fichiers".
.FilterIndex = 1

'Indique le type d'affichage dans la boîte de dialogue
(exemple visualisation des propriétés)
'Affiche la boîte de dialogue
.Show
If .SelectedItems.Count > 0 Then
    'Affecte à la variable workbook le fichier sélectionné
    qui sera ouvert dans l'application.
    Set WBreims = Workbooks.Open(.SelectedItems(1))
    'Affecte à la variable worksheet la première feuille
    du classeur défini précédemment.
    Set WSReims = WBreims.Sheets(1)
Else
    'Si aucun fichier n'est sélectionné, on sort de la procédure
    MsgBox "Vous n'avez pas sélectionné de fichiers,
    veuillez recommencer."
    Exit Sub
End If

End With
With Application.FileDialog(msoFileDialogFilePicker)
    .Title = "Choisir le fichier Excel pour Paris:"
    .AllowMultiSelect = False
    .InitialFileName = "Immo_Paris.xlsx"
    .Filters.Clear
    .Filters.Add "Classeurs Excel", "*.xls; *.xlsx; *.xlsm"
    .FilterIndex = 1
    .Show
    If .SelectedItems.Count > 0 Then
        Set WBPParis = Workbooks.Open(.SelectedItems(1))
        Set WSPParis = WBPParis.Sheets(1)
    End If
End With

'Définition des variables
Dim LigneParis, LigneReims, LigneConsolide As Integer
'Affectation des valeurs
LigneParis = 2
```

```

LigneReims = 2
LigneConsolide = 2

'Mise en place de la structure With ... End With pour ne pas avoir
à réécrire à chaque fois le nom de la feuille WSParis
With WSParis
'Création d'une boucle dont la condition de sortie est la rencontre
d'une cellule vide
  While .Cells(LigneParis, 1).Value <> ""
    'La cellule de colonne 1 prendra la valeur Paris quel que soit la
    valeur de LigneConsolide.
      WSFinal.Cells(LigneConsolide, 1).Value = "Paris"

      WSFinal.Cells(LigneConsolide, 2).Value = .Cells(LigneParis,
2).Value
      WSFinal.Cells(LigneConsolide, 6).Value = .Cells(LigneParis,
6).Value
      WSFinal.Cells(LigneConsolide, 8).Value = .Cells(LigneParis,
4).Value
      WSFinal.Cells(LigneConsolide, 9).Value = .Cells(LigneParis,
5).Value
      WSFinal.Cells(LigneConsolide, 14).Value = .Cells(LigneParis,
10).Value
      WSFinal.Cells(LigneConsolide, 15).Value = .Cells(LigneParis,
11).Value
      WSFinal.Cells(LigneConsolide, 16).Value = .Cells(LigneParis,
15).Value
      WSFinal.Cells(LigneConsolide, 13).Value = .Cells(LigneParis,
9).Value

      Dim Surf As Double
      Dim Propose, Compromis As Long
      If .Cells(LigneParis, 7).Value = "" Then
        WSFinal.Cells(LigneConsolide, 3).Value = "Location"
        'La surface de location correspond au prix de location
        divisé par le prix au m2. La fonction Excel MROUND (arrondi au multiple)
        est appliquée au calcul pour obtenir un nombre entier.
        Surf =
Application.WorksheetFunction.MRound((.Cells(LigneParis, 13).Value /
.Cells(LigneParis, 14).Value), 1)
        'récupération du prix proposé initialement
        Propose = .Cells(LigneParis, 12).Value
        'récupération du prix final
        Compromis = .Cells(LigneParis, 13).Value
      Else
        'Le même raisonnement que ci-dessus s'applique.
        WSFinal.Cells(LigneConsolide, 3).Value = "Vente"
        Surf = Application.WorksheetFunction.MRound((.Cells

```

```

(LigneParis, 8).Value / .Cells(LigneParis, 14).Value), 1)
    Propose = .Cells(LigneParis, 7).Value
    PrixFinal = .Cells(LigneParis, 8).Value
End If
'Une fois les valeurs récupérées, elles sont affichées dans
la feuille Consolidée.
WSFinal.Cells(LigneConsolide, 7).Value = Surf
WSFinal.Cells(LigneConsolide, 10).Value = "Non renseigné"
WSFinal.Cells(LigneConsolide, 11).Value = Propose
WSFinal.Cells(LigneConsolide, 12).Value = Compromis
'Appel de la fonction ConvTS
WSFinal.Cells(LigneConsolide, 4).Value = ConvTS(.Cells
(LigneParis, 1).Value)
WSFinal.Cells(LigneConsolide, 4).NumberFormat = "dd/mm/yyyy;@"
WSFinal.Cells(LigneConsolide, 5).Value = WSFinal.Cells
(LigneConsolide, 4).Value + .Cells(LigneParis, 16).Value
WSFinal.Cells(LigneConsolide, 5).NumberFormat = "dd/mm/yyyy;@"
WSFinal.Cells(LigneConsolide, 11).NumberFormat = "#,##0.00 €"
WSFinal.Cells(LigneConsolide, 12).NumberFormat = "#,##0.00 €"
WSFinal.Cells(LigneConsolide, 13).NumberFormat = "#,##0.00 €"
'Itération de la variable LigneParis afin d'explorer la ligne
suivante
    LigneParis = LigneParis + 1
'Itération de la variable LigneConsolide afin que la ligne de
destination dans le fichier consolidé soit augmentée de 1.
    LigneConsolide = LigneConsolide + 1
Wend
End With

With WSReims
    While .Cells(LigneReims, 1).Value <> ""
        WSFinal.Cells(LigneConsolide, 1).Value = "Reims"
        'données reprises sans modification
        WSFinal.Cells(LigneConsolide, 2).Value = .Cells(LigneReims,
1).Value
        WSFinal.Cells(LigneConsolide, 4).Value = .Cells(LigneReims,
7).Value
        WSFinal.Cells(LigneConsolide, 5).Value = .Cells(LigneReims,
13).Value
        WSFinal.Cells(LigneConsolide, 8).Value = .Cells(LigneReims,
8).Value
        WSFinal.Cells(LigneConsolide, 9).Value = .Cells(LigneReims,
9).Value
        WSFinal.Cells(LigneConsolide, 10).Value = .Cells(LigneReims,
10).Value
        WSFinal.Cells(LigneConsolide, 11).Value = .Cells(LigneReims,

```

```

4).Value
    WSFinal.Cells(LigneConsolide, 12).Value = .Cells(LigneReims,
12).Value
    WSFinal.Cells(LigneConsolide, 14).Value = .Cells(LigneReims,
15).Value
    WSFinal.Cells(LigneConsolide, 15).Value = .Cells(LigneReims,
16).Value
    WSFinal.Cells(LigneConsolide, 16).Value = .Cells(LigneReims,
14).Value
    'Situation de vente
    If .Cells(LigneReims, 2).Value = "L" Then
        WSFinal.Cells(LigneConsolide, 3).Value = "Location"
    Else
        WSFinal.Cells(LigneConsolide, 3).Value = "Vente"
    End If
    'Type de bien
    Select Case .Cells(LigneReims, 6).Value
    Case "A"
        WSFinal.Cells(LigneConsolide, 6).Value = "Appartement"
    Case "V"
        WSFinal.Cells(LigneConsolide, 6).Value = "Villa"
    Case "M"
        WSFinal.Cells(LigneConsolide, 6).Value = "Maison"
    End Select
    ' Définition de la surface
    WSFinal.Cells(LigneConsolide, 7).Value = Cint(Left(.Cells
(LigneReims, 5).Value, Len(.Cells(LigneReims, 5).Value) - 3))
    'Calcul des honoraires
    WSFinal.Cells(LigneConsolide, 13).Value = .Cells
(LigneReims, 12).Value * .Cells(LigneReims, 3).Value
    'Mise à jour des formats
    WSFinal.Cells(LigneConsolide, 11).NumberFormat = "#,##0.00 €"
    WSFinal.Cells(LigneConsolide, 12).NumberFormat = "#,##0.00 €"
    WSFinal.Cells(LigneConsolide, 13).NumberFormat = "#,##0.00 €"

    LigneReims = LigneReims + 1
    LigneConsolide = LigneConsolide + 1
Wend
End With
'Fermeture des classeurs ouverts
WBParis.Close
WBReims.Close
'Message pour annoncer la fin du traitement
MsgBox ("Traitement terminé, les données ont été correctement
importées.")
End Sub

```

```
Function ConvTS(Num As Long)
'diviser la valeur par 24h, 60 minutes et 60 secondes pour convertir
en jour
ConvTS = Int(Num / 24 / 60 / 60)
'ajouter la valeur du 1er janvier 1970 soit 25569
ConvTS = ConvTS + 25569
End Function
```

B. Partage des données

1. Description de l'exemple

a. Présentation de l'exemple

L'objectif de cet exemple est de proposer une solution permettant aux deux agences immobilières la saisie des données. La problématique est que ce fichier Excel n'a pas vocation à être maintenu par une seule agence, mais il doit être accessible et modifiable par les deux agences et peut-être à terme, par une multitude d'agences.

b. Présentation des classeurs et outils utilisés

Dans le cadre de cet exemple, le fichier généré dans la première partie de ce chapitre sera utilisé puisqu'il contient les données saisies dans les agences. Toutefois, le fichier a subi quelques améliorations avec notamment un onglet **Paramètres**. Le fichier sur lequel nous allons nous baser est le fichier **Enoncé_6-B.xlsm**.

Cet exemple va requérir un compte sur Microsoft OneDrive (<https://onedrive.live.com/>) et dans le cadre de la dernière partie, il est nécessaire d'avoir Outlook 2016 (ou version antérieure) installé sur votre poste. Si vous ne possédez pas Outlook, le code peut être facilement adapté pour d'autres solutions.

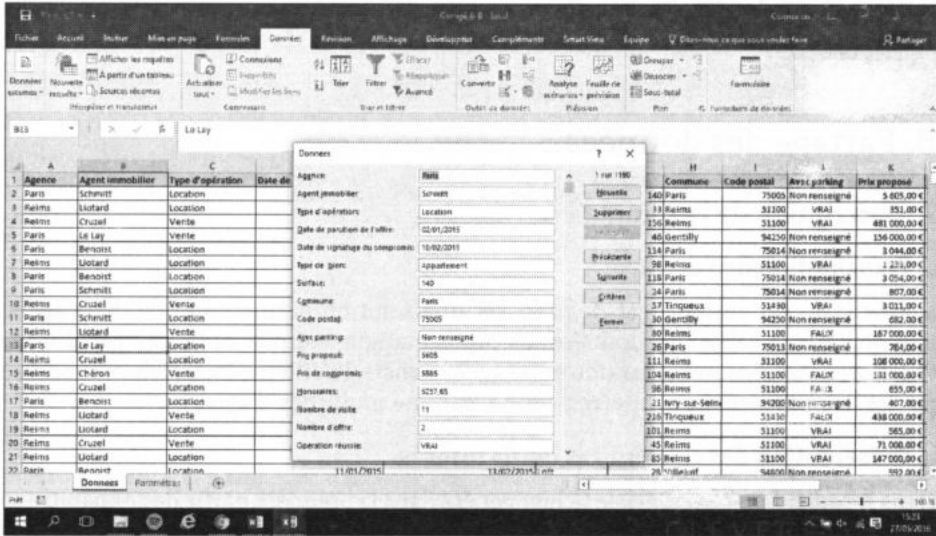
Le compte OneDrive permettra de créer un formulaire Excel en ligne dont les valeurs seront stockées au sein d'un fichier.

L'application Outlook permettra d'envoyer un e-mail à partir de données contenues dans Excel.

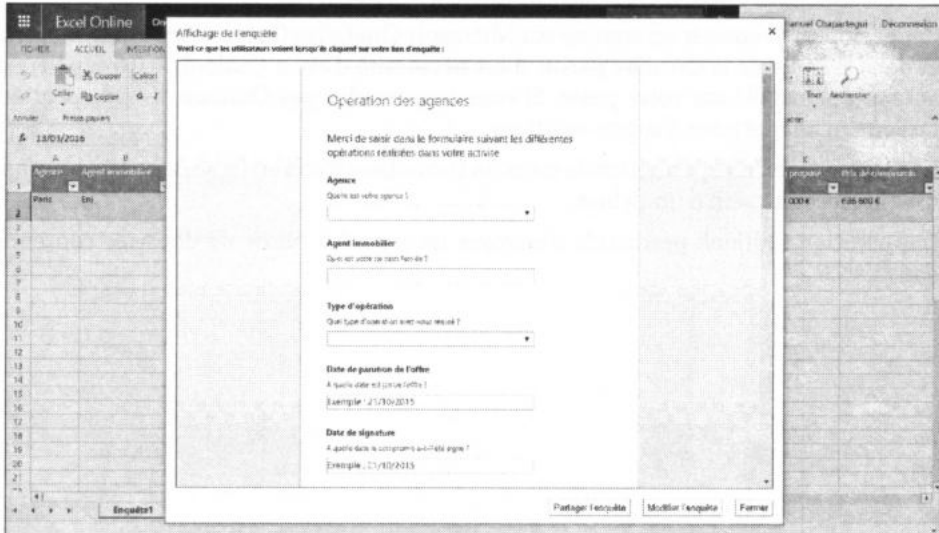
c. Fonctionnalités

Les fonctionnalités qui seront abordées sont les suivantes :

- Créer un formulaire de saisie automatique pour faciliter la saisie des données ;



- Créer une enquête partagée via OneDrive et la diffuser ;



- Envoyer un e-mail aux agences avec les statistiques mensuelles des opérations réalisées par les différentes agences. Le corps de cet e-mail doit ressembler à cela :

Bonjour,

Voici les résultats du mois

Location à Paris : 40

Vente à Paris : 27

Location à Reims : 27

Vente à Reims : 13

Cordialement

2. Notions de cours

a. Formulaire de tableau

L'option **Formulaire** est une fonctionnalité d'Excel permettant d'ajouter/modifier/supprimer des données dans une série de données. Généralement utilisée avec des tableaux, cette fonctionnalité peut être utilisée également avec une simple série de données.

Avantages

L'avantage de cette fonctionnalité est de générer un formulaire de saisie et de modification de manière simple, juste via un clic. L'édition, l'ajout et la suppression sont simples d'accès et il est même possible de rechercher un élément.

Inconvénients

L'inconvénient majeur est le manque de possibilités d'aide à la saisie.

Il est impossible de qualifier la donnée à insérer. De plus, si vous avez appliqué des contraintes sur les données (onglet **Données - Validation des données**), vous risquez de ne pas pouvoir insérer vos données avec le formulaire. En effet, si la valeur saisie ne répond pas à la valeur attendue, l'ensemble de la ligne ne sera pas inséré.

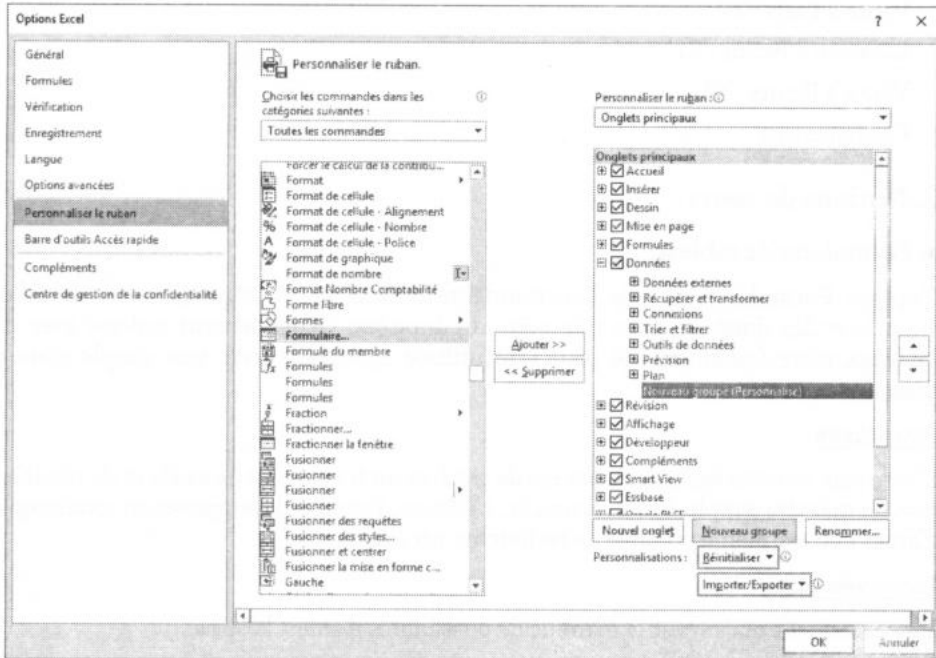
Comment insérer le formulaire ?

🔗 Ouvrez le fichier **ExempleCours_Chapitre_6.xlsx**.

Vous trouverez sur la feuille deux tableaux identiques en **A1:C9** et **H1:J9**. Chaque tableau comporte trois colonnes listant les prénoms, le sexe et le nom de l'équipe.

Le premier tableau sur la plage **A1:C9** est une plage de données, non déclarée en tant que tableau Excel. Aucune case ne porte de contrainte. Le second tableau sur la plage **H1:J9**, est un tableau Excel. Les colonnes concernant le sexe et le nom de l'équipe sont obligatoires : l'utilisateur doit choisir une des valeurs.

- ❏ Dans l'onglet **Fichier**, choisissez **Options**.
- ❏ Cliquez sur **Personnaliser le ruban**, puis dans la zone **Choisir les commandes dans les catégories suivantes**, sélectionnez **Toutes les commandes**.
- ❏ Dans la liste de gauche, sélectionnez **Formulaire**.
- ❏ Dans la liste de droite, sélectionnez **Données** puis cliquez sur **Nouveau groupe**.



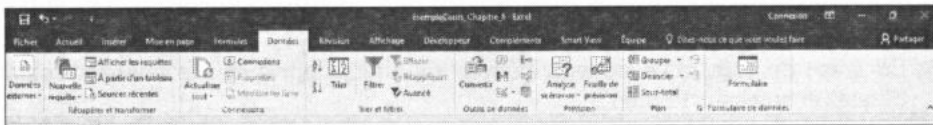
- ❏ Cliquez sur **Ajouter >>** pour insérer le bouton **Formulaire** dans ce nouveau groupe de l'onglet **Données**.

- ☞ Cliquez sur **Renommer** pour appeler le groupe **Formulaire de données** comme ci-dessous.



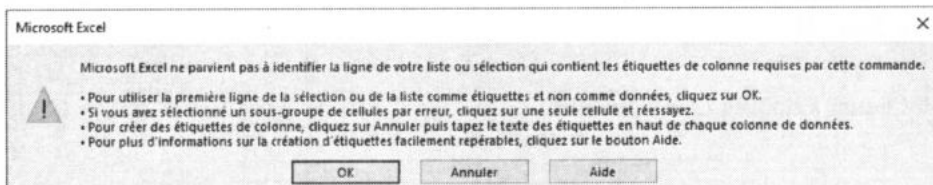
- ☞ Cliquez deux fois sur **OK** pour fermer la fenêtre **Renommer** et la fenêtre **Options Excel**.

Vous obtenez l'affichage suivant :



- ☞ Sélectionnez dans le tableau de gauche la cellule **A1**, puis dans l'onglet **Données**, cliquez sur **Formulaire**.

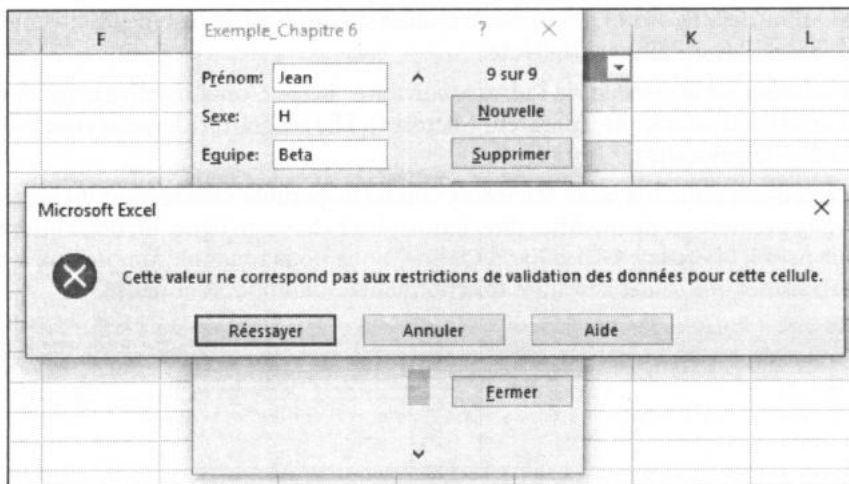
La fenêtre suivante apparaît, cliquez sur **OK** pour afficher le formulaire.



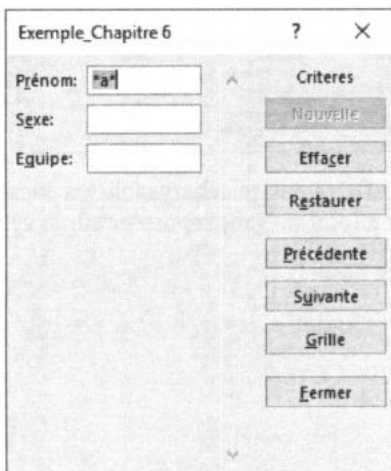
Le formulaire s'affiche ainsi :


1. Le bouton **Nouvelle** permet de créer un nouvel enregistrement vide.
 2. Le bouton **Supprimer** permet de supprimer l'enregistrement en cours.
 3. Les boutons **Précédente**/**Suivante** permettent de naviguer parmi les enregistrements.
 4. Le bouton **Critères** permet de saisir des critères pour afficher uniquement les enregistrements qui respectent ces critères.
- ☞ En guise de test, vous pouvez ajouter/modifier/supprimer des enregistrements. Cliquez ensuite sur **Fermer**.
 - ☞ Cliquez ensuite sur la cellule H1, puis dans l'onglet **Données**, cliquez sur **Formulaire**. Là aussi le formulaire s'affiche pour le nouveau tableau.
 - ☞ Cliquez sur **Nouvelle**, puis saisissez les valeurs suivantes :
 - ▶ Prénom : Jean ;
 - ▶ Sexe : H ;
 - ▶ Equipe : Beta.
 - ☞ Cliquez à nouveau sur **Nouvelle**.

Une contrainte a été appliquée sur le tableau : seules les valeurs "Homme" et "Femme" sont acceptées pour la colonne "Sexe". Le message suivant nous informe que la valeur "H" n'est pas conforme.



- ☞ Cliquez sur Réessayer pour modifier H en Homme.
- ☞ Cliquez ensuite Critères, puis dans la zone de texte du Prénom saisissez *a*. Cela permettra de filtrer les éléments contenant un a. Le caractère * est un caractère joker qui remplace tous les autres caractères.



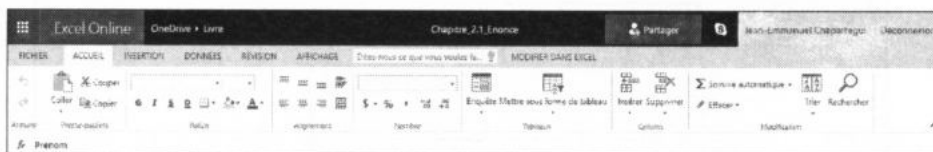
- ☞ Appuyez ensuite sur la touche  puis naviguez avec les boutons Précédente et Suivante parmi les enregistrements dont le prénom contient un « a ».

b. OneDrive

OneDrive est un espace de stockage en ligne autrefois connu sous le nom de SkyDrive. Il permet de stocker et de partager des fichiers de tout type mais aussi d'utiliser les versions Online de Microsoft Office pour travailler sur des fichiers de type Microsoft Office Word, Excel, PowerPoint et OneNote.

Cette solution est accessible via l'adresse suivante : <https://onedrive.live.com/> et nécessite la création d'un compte utilisateur Microsoft. Elle est gratuite lorsque vous possédez moins de 5 Giga-octets de stockage.

Vous accédez donc à la suite Microsoft Online disponible directement via un navigateur vous permettant de modifier directement les fichiers en ligne. En nous intéressant de plus près à Microsoft Office Excel Online, nous nous rendons compte que le menu est restreint et que seules certaines fonctionnalités sont opérationnelles.



Cette solution permet de gérer une édition partagée, c'est-à-dire que plusieurs utilisateurs peuvent être amenés à éditer le même fichier en simultanément. Certaines fonctions sont donc indisponibles, comme l'usage de macro VBA ou de contrôle de formulaires.

En revanche, il sera plus aisé de commenter et d'interagir avec les autres utilisateurs de ce fichier.

	A	B	C	D	E	F	G	H	I	J
1	Prénoms	Noms	Sexe	Date de naissance	Grade	Filiale	Salaires Brut annuel	Primes annuelles en cours	Ancienneté	
2	Dexter	Alcott	H	02/07/1994	180	Business	70900		3500	
3	Robert	Albernetry	H	07/04/1971	180	Fonctionnaires	70900		2000	1
4	André	Acoves	H	28/11/1969	180	Business	100200		3500	2
5	Mike	Aczer	H	03/12/1978	140	Business	55400		2500	
6	Bruce	Acosta	H	04/07/1975	140	Fonctionnaires	60900		2500	
7	John	Adair	H	18/10/1976	180	Fonctionnaires	71500		6500	
8	Benjamin	Adams	H	15/05/1980	120	Fonctionnaires	50600		500	

Le fichier stocké sur OneDrive est téléchargeable en local sur votre poste, mais les modifications apportées en local ne sont reportées sur la version stockée sur OneDrive qu'après synchronisation des fichiers.

OneDrive permet de gérer les différentes versions d'un même fichier et conserve autant de versions que nécessaire.

The screenshot shows the Excel Online interface. At the top, it displays 'Version actuelle' (Current version) as '15/05/2016 00:05' by 'Jean-Emmanuel Chapartegui'. Below this, a list of 'Versions antérieures' (Previous versions) is shown, with the most recent being '14/05/2016 15:59' by 'Jean-Emmanuel Chapartegui'. The main area shows a spreadsheet with columns A through J and rows 1 through 10. The data in the spreadsheet is as follows:

	A	B	C	D	E	F	G	H	I	J
1	Prenon	Sexe	Equipe					Prenon	Sexe	Equipe
2	Cécile	Femme	Alpha					Cécile	Femme	Alpha
3	Alexandra	Femme	Alpha					Alexandra	Femme	Alpha
4	David	Homme	Alpha					David	Homme	Alpha
5	Jerémy	Homme	Beta					Jerémy	Homme	Beta
6	Charlie	Femme	Alpha					Charlie	Femme	Alpha
7	Geoffroy	Homme	Beta					Geoffroy	Homme	Beta
8	Lidye	Femme	Beta					Lidye	Femme	Beta
9	Emmanuel	Homme	Beta					Emmanuel	Homme	Beta
10										

c. Enquêtes

Une enquête est un formulaire partagé avec d'autres utilisateurs dont les réponses sont concaténées au sein d'un fichier.

L'enquête n'est pas disponible sur Microsoft Office Excel en local, c'est une fonctionnalité réservée à Excel Online.

L'enquête se présente ainsi :

The screenshot shows a dialog box titled 'Modifier l'enquête' (Edit Survey). It contains the following text and input fields:

- Veuillez entrer un titre pour votre enquête ici
- Veuillez entrer une description pour votre enquête ici
- Veuillez entrer votre première question ici
-
- + Ajouter une nouvelle question

At the bottom of the dialog box, there are three buttons: 'Partager l'enquête' (Share Survey), 'Enregistrer et afficher' (Save and Show), and 'Fermer' (Close).

La construction d'une enquête consiste à ajouter une liste de questions se caractérisant par :

- ▶ Un titre ;
- ▶ Une sous question ;
- ▶ Un format de réponse attendu.

La question est créée ainsi :

The screenshot shows the 'Modifier l'enquête' (Edit Survey) interface. On the left, under 'Mon enquête', there is a preview of a question: 'Content' with the sub-question 'Êtes vous content de votre apprentissage' and a dropdown menu set to 'Oui'. Below this is a button 'Ajouter une nouvelle question'. On the right, a 'MODIFIER LA QUESTION' dialog box is open, showing the same question details. It includes fields for 'Question' (Content), 'Sous-titre de la question' (Êtes vous content de votre apprentissage), 'Type de réponse' (Oui/Non), 'Nécessaire' (checkbox), and 'Valeur par défaut' (Oui). There are also buttons for 'Réponse entrée automatiquement', 'Terminer', and 'Supprimer la question'. At the bottom of the main window are buttons for 'Partager l'enquête', 'Enregistrer et afficher', and 'Fermer'.

En partageant l'enquête, vous obtenez un lien à transmettre aux participants de l'enquête. Vous pouvez aussi l'afficher directement avec le bouton **Enregistrer et afficher**.

The screenshot shows the 'Obtenir un lien vers votre enquête « Mon enquête »' (Get a link to your survey) interface. On the left is a preview of the survey form. On the right, there is a text box explaining: 'Les utilisateurs disposant de ce lien pourront envoyer des réponses sans se connecter. Ils ne verront pas le résultat.' Below this is a 'Créer un lien' button. At the bottom right is a 'Terminé' button.

Le résultat de l'enquête est inséré dans un fichier Excel qui n'est pas partagé par défaut.



d. Envoyer un e-mail avec VBA via Outlook

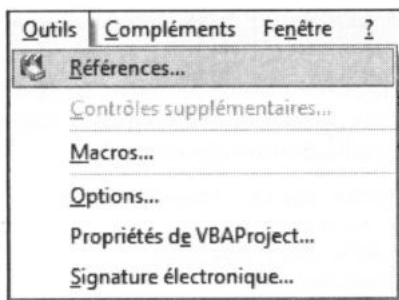
L'objectif est d'envoyer un e-mail Outlook avec VBA. Cela nécessite donc d'avoir un compte Outlook paramétré.

Toutefois, il est possible, mais plus complexe, de configurer manuellement le serveur d'envoi de mails.

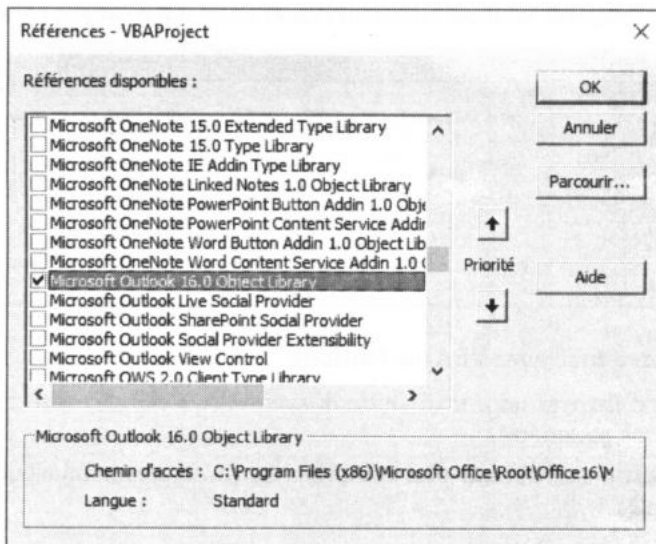
Ajouter la référence Outlook

La première étape consiste à ajouter la **bibliothèque Microsoft Office** en tant que référence au projet VBA. Cette référence vous permettra d'utiliser Microsoft Outlook à partir de VBA.

Pour ajouter cette référence, allez sur **Visual Basic Editor**. Dans le menu **Outils**, cliquez sur **Références** puis recherchez votre version de Microsoft Outlook.



Sélectionnez la version d'Outlook disponible Microsoft Outlook XX Object Library où XX dépend de votre version de Microsoft Office.



Rédaction du code

Ce code Excel VBA va permettre d'envoyer un e-mail via l'application Outlook. Les informations à envoyer par mail sont collectées dans le code VBA Excel... L'application Outlook est contrôlée via Excel VBA.

```
Sub EnvoiMail()  
'Contenu de la procédure  
End Sub
```

Dans un premier temps, il est nécessaire de créer un objet Outlook à partir duquel le message sera instancié. Cette opération se fait en deux étapes : création des objets (définition des variables) puis initialisation des objets.

```
'Création de la variable application Outlook  
Dim Email As Outlook.Application  
'Création de la variable Mail Outlook  
Dim EmailMsg As Outlook.MailItem  
'Initialisation de l'objet application Outlook  
Set Email = CreateObject("Outlook.Application")  
'Création du mail dans l'objet application Outlook  
Set EmailMsg = Email.CreateItem(olMailItem)
```

Manipulation de l'objet

Une fois l'objet mail créé, il est possible d'ajouter un ou plusieurs destinataires, un sujet ou un corps de message. Tous les champs ci-dessous sont des valeurs de type chaîne de caractères qui peuvent être récupérées en dehors du code (par exemple dans une cellule).

```
' Ajout d'un destinataire
EmailMsg.Recipients.Add "destinataire@mail.com"
'Ajout d'un sujet à l'email
EmailMsg.Subject = "Titre de l'email"
'Ajout d'un contenu à l'email
EmailMsg.Body = "Corps du mail"
```

Envoi du mail et destruction d'objet

Pour envoyer le mail, il faut utiliser la méthode `Send` de l'objet `MailItem`.

```
'Envoi de l'email
EmailMsg.Send
```

Même si cette notion a été peu abordée jusque-là, la destruction d'objet fait partie intégrante de la programmation. Visual Basic est un langage simplifié adapté pour des non-programmeurs, par conséquent de nombreuses *bonnes pratiques* ne sont pas nécessaires. Prendre de bonnes habitudes permet néanmoins de fluidifier, simplifier votre code mais également de le rendre plus lisible et plus facile à maintenir.

La destruction de variable revient à ne plus allouer de mémoire à une variable. Si vous avez 100 variables non typées avec de la mémoire allouée, votre programme se retrouvera ralenti. Or, si vous prenez l'habitude de typer vos variables et de les « détruire » après usage, la mémoire utilisée sera moindre et par conséquent votre programme sera plus fluide. Vous aurez également plus de facilité pour suivre votre programme.

Pour détruire une variable, la solution la plus commune est de leur affecter une valeur vide ou rien (`Nothing`). C'est ce que nous allons faire pour les objets mail et application Outlook en leur affectant rien (`Nothing`) en valeur. Cette opération est faite via une instruction `Set` qui permet d'attribuer une valeur à un objet.

```
'Destruction des variables
Set EmailMsg = Nothing
Set Email = Nothing
End Sub
```

Il est possible d'aller beaucoup plus loin dans le paramétrage d'un e-mail comme par exemple :

- ▶ Contenu de l'e-mail en HTML ;
- ▶ Ajout de destinataires multiples ;
- ▶ Ajout de personnes en copie cachée ;
- ▶ Ajout de pièces jointes ;
- ▶ Ajout de l'importance dans l'e-mail...

3. Réalisation de l'exemple

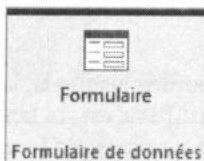
☞ Commencez par ouvrir le fichier `Enoncé_6-B.xlsm`.

a. Créer un formulaire de saisie automatique pour faciliter la saisie des données

Le formulaire de saisie va permettre la saisie de données sur un tableau sans avoir à utiliser un formulaire VBA.

Afficher le formulaire

- ☞ Si vous n'avez pas ajouté le bouton **Formulaire**, ajoutez-le à l'onglet **Données** (référez-vous à la partie Formulaire de tableau dans les Notions de cours de ce chapitre).
- ☞ Sélectionnez la cellule A1.
- ☞ Dans l'onglet **Données**, cliquez sur bouton **Formulaire**.




Le formulaire apparaît ainsi :

Donnees		?	×
Agence:	Paris	1 sur 1190	Nouvelle
Agent immobilier:	Schmitt		Supprimer
Type d'opération:	Location		Réinitialiser
Date de parution de l'offre:	02/01/2015		Précédente
Date de signature:	10/02/2015		Suivante
Type de bien:	Appartement		Critères
Surface:	140		Fermer
Commune:	Paris		
Code postal:	75005		
Avec parking:	Non renseigné		
Prix proposé:	5605		
Prix final:	5885		
Honoraires:	5237,65		
Nombre de visite:	11		
Nombre d'offre:	2		
Opération réussie:	VRAI		

Modifier une donnée

L'agent Benoist vous informe qu'une vente n'a finalement pas eu lieu suite à un problème de dernière minute sur l'attribution du prêt. Il sait que l'offre est parue le 15/12/2016.

- ☞ Cliquez sur **Critères** puis saisissez 15/12/2015 dans la zone **Date de parution de l'offre**.
- ☞ Appuyez sur la touche , puis cliquez sur les boutons **Précédente** et **Suivante** pour accéder à l'opération effectuée par l'agent Benoist le 15/12/2015.

Donnees		?	×
Agence:	<input type="text" value="Paris"/>	1129 sur 1190	<input type="button" value="Nouvelle"/>
Agent immobilier:	<input type="text" value="Benoist"/>		<input type="button" value="Supprimer"/>
Type d'opération:	<input type="text" value="Vente"/>		<input type="button" value="Restaurer"/>
Date de parution de l'offre:	<input type="text" value="15/12/2015"/>		<input type="button" value="Précédente"/>
Date de signature:	<input type="text" value="20/01/2016"/>		<input type="button" value="Suivante"/>
Type de bien:	<input type="text" value="Appartement"/>		<input type="button" value="Critères"/>
Surface:	<input type="text" value="35"/>		<input type="button" value="Fermer"/>
Commune:	<input type="text" value="Paris"/>		
Code postal:	<input type="text" value="75014"/>		
Avec parking:	<input type="text" value="Non renseigné"/>		
Prix proposé:	<input type="text" value="270000"/>		
Prix final:	<input type="text" value="243000"/>		
Honoraires:	<input type="text" value="12150"/>		
Nombre de visite:	<input type="text" value="4"/>		
Nombre d'offre:	<input type="text" value="2"/>		
Opération réussie:	<input type="text" value="VRAI"/>		

- ☞ Modifiez la valeur du champ Opération réussie de VRAI en FAUX.

Donnees		?	×
Agence:	Paris	1129 sur 1190	
Agent immobilier:	Benoist		<input type="button" value="Nouvelle"/>
Type d'opération:	Vente		<input type="button" value="Supprimer"/>
Date de parution de l'offre:	15/12/2015		<input type="button" value="Restaurer"/>
Date de signature:	20/01/2016		<input type="button" value="Précédente"/>
Type de bien:	Appartement		<input type="button" value="Suivante"/>
Surface:	35		<input type="button" value="Critères"/>
Commune:	Paris		<input type="button" value="Fermer"/>
Code postal:	75014		
Avec parking:	Non renseigné		
Prix proposé:	270000		
Prix final:	243000		
Honoraires:	12150		
Nombre de visite:	4		
Nombre d'offre:	2		
Opération réussie:	FAUX		

- ☞ Vérifiez la donnée avec la valeur de la cellule P1131 : elle est bien passée à FAUX.

Rechercher une donnée

L'agent Cruzel recherche une de ses ventes sur laquelle s'est glissée une erreur : le client avait un parking. Il se souvient que c'était la vente d'une villa d'environ 250 000 €. Utilisez l'outil pour effectuer la recherche

- ☞ Affichez à nouveau le formulaire en cliquant sur le bouton **Formulaire** dans l'onglet **Données**. Le formulaire apparaît.
- ☞ Cliquez sur le bouton **Critères**.

☞ Saisissez les informations qui permettent de trouver ces deux lignes :

Donnees		? X
Agence:	<input type="text"/>	Criteres <input type="button" value="Nouvelle"/> <input type="button" value="Effacer"/> <input type="button" value="Restaurer"/> <input type="button" value="Précédente"/> <input type="button" value="Suivante"/> <input type="button" value="Grille"/> <input type="button" value="Fermer"/>
Agent immobilier:	Cruzel	
Type d'opération:	Vente	
Date de parution de l'offre:	<input type="text"/>	
Date de signature:	<input type="text"/>	
Type de bien:	Villa	
Surface:	<input type="text"/>	
Commune:	<input type="text"/>	
Code postal:	<input type="text"/>	
Avec parking:	<input type="text"/>	
Prix proposé:	<input type="text"/>	
Prix final:	25*	
Honoraires:	<input type="text"/>	
Nombre de visite:	<input type="text"/>	
Nombre d'offre:	<input type="text"/>	
Opération réussie:	<input type="text"/>	

☞ Après avoir vérifié que vous avez bien trouvé l'enregistrement, changez la valeur du champ Avec parking de FAUX à VRAI.

Donnees		? X
Agence:	Reims	502 sur 1191
Agent immobilier:	Cruzel	<input type="button" value="Nouvelle"/>
Type d'opération:	Vente	<input type="button" value="Supprimer"/>
Date de parution de l'offre:	09/06/2015	<input type="button" value="Restaurer"/>
Date de signature:	20/06/2015	<input type="button" value="Précédente"/>
Type de bien:	Villa	<input type="button" value="Suivante"/>
Surface:	132	<input type="button" value="Critères"/>
Commune:	Reims	<input type="button" value="Fermer"/>
Code postal:	51100	
Avec parking:	VRAI	
Prix proposé:	1557	
Prix final:	253000	
Honoraires:	0	
Nombre de visite:	1	
Nombre d'offre:	0	
Opération réussie:	FAUX	

b. Créer une enquête partagée via OneDrive et la diffuser

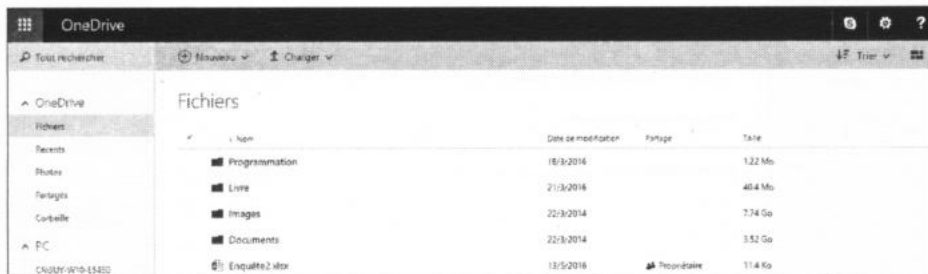
Le formulaire permettant une saisie en local, l'enquête semble en tout point répondre au besoin de partager avec les agences un même formulaire de données. L'enquête va permettre aux agences d'avoir un lien web pour saisir leurs données qui seront consolidées dans un même fichier.

Créer un compte OneDrive

- ☞ Si vous ne l'avez pas fait pour la partie cours, créez un compte OneDrive sur le site : <https://onedrive.live.com/>

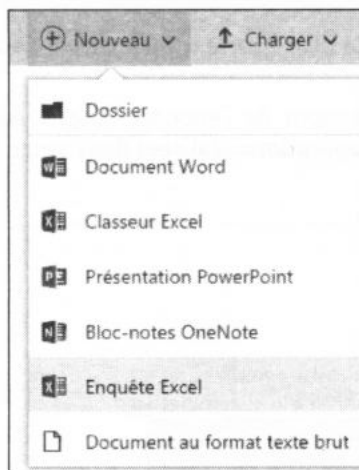
The image shows a screenshot of the Microsoft login page. At the top, there is a navigation bar with icons for various services. The main heading is "Connexion". Below it, the text reads "Utilisez votre compte Microsoft. Qu'est-ce que c'est ?". There are two input fields: "Adresse e-mail ou téléphone" and "Mot de passe". Below these fields is a checkbox labeled "Maintenir la connexion". A large black button with white text says "Se connecter". Below the button, there is a link: "Vous n'avez pas encore de compte ? Créez-en un !". At the bottom, there are two more links: "J'ai oublié mon mot de passe" and "Se connecter avec un code à usage unique". The Microsoft logo is at the very bottom.

L'interface de OneDrive s'affiche ainsi :

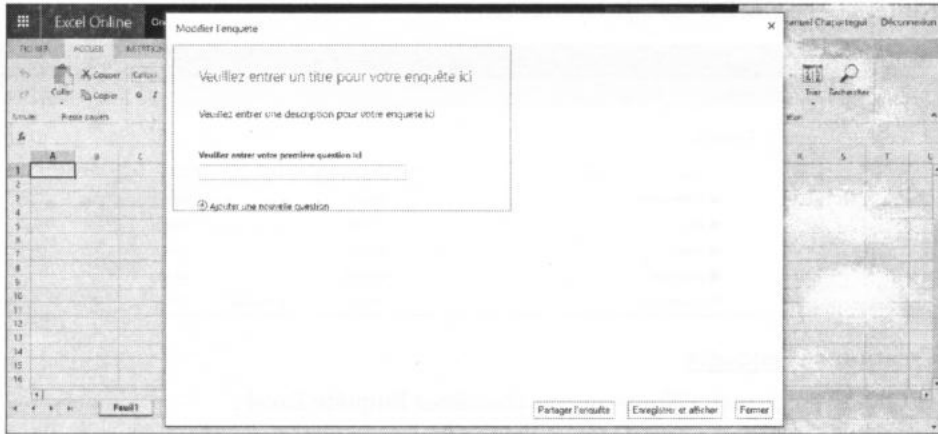


Création de l'enquête

☞ Cliquez sur le menu Nouveau puis choisissez **Enquête Excel** :



Un nouveau fichier Excel nommé **Enquête1.xlsx** s'affiche et vous propose d'emblée de créer votre enquête.



- ❏ Commencez par saisir un titre à votre enquête, ici nous allons choisir **Opération des agences**.
- ❏ Saisissez ensuite le descriptif de l'enquête : **Merci de saisir dans le formulaire suivant les différentes opérations réalisées dans votre activité.**

Le résultat est le suivant :



☞ Cliquez sur la première question pour la sélectionner.

Une icône apparaît :


Modifier l'enquête

Opération des agences

Merci de saisir dans le formulaire suivant les différentes opérations réalisées dans votre activité

Veillez entrer votre première question ici

+ Ajouter une nouvelle question

☞ Cliquez sur l'icône  pour afficher le détail de la question.

Il s'affichera ainsi :

Modifier l'enquête

Opération des agences

Merci de saisir dans le formulaire suivant les différentes opérations réalisées dans votre activité

Veillez entrer votre première question ici

+ Ajouter une nouvelle question

MODIFIER LA QUESTION ✕

Question

Sous-titre de la question

Type de réponse

Nécessaire

Réponse par défaut

Terminé

Supprimer la question

Le questionnaire doit se caler parfaitement avec les données importées, par conséquent la première question doit porter sur le nom de l'agence. En effet, la réponse à la question 1 sera saisie dans la colonne A du fichier Excel.

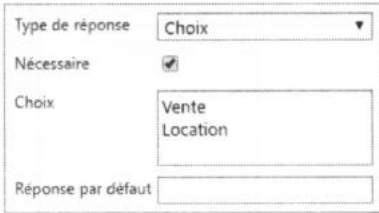

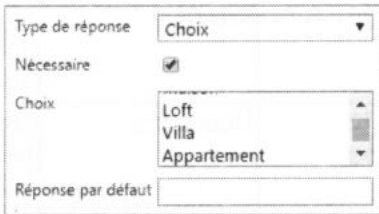
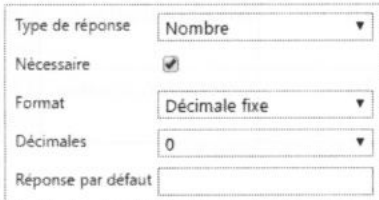
☞ Dans la zone **Question**, saisissez **Agence**. Ce terme agence sera le titre de notre colonne dans le fichier de restitution, c'est pourquoi il est nécessaire de se caler sur la structure du format consolidé.

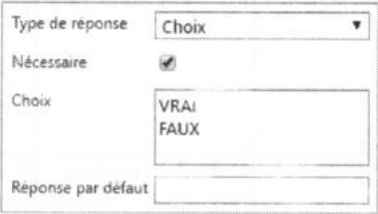
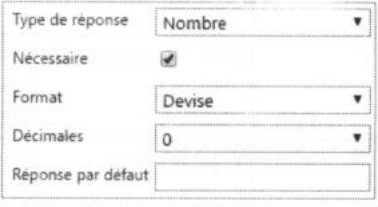
- ☞ Dans la zone **Sous-titre de la question**, saisissez **Quelle est votre agence ?**, qui correspond à la question visible par l'utilisateur.
- ☞ En valeur attendue, choisissez le **Type de réponse Choix** puis rentrez dans la liste les valeurs **Reims** et sur la ligne d'en dessous **Paris**.
- ☞ Cochez la case **Nécessaire** pour signifier que la réponse est obligatoire.


Comme tous les champs de notre questionnaire, il n'y aura pas de valeur par défaut et ils seront tous obligatoires (nécessaires).

- ☞ Reproduisez le questionnaire pour les 16 questions tel qu'indiqué sur le tableau :

Question	Question	Sous-titre de la question	Valeur attendue
1	Agence	Quelle est votre agence ?	Choix : Reims ou Paris

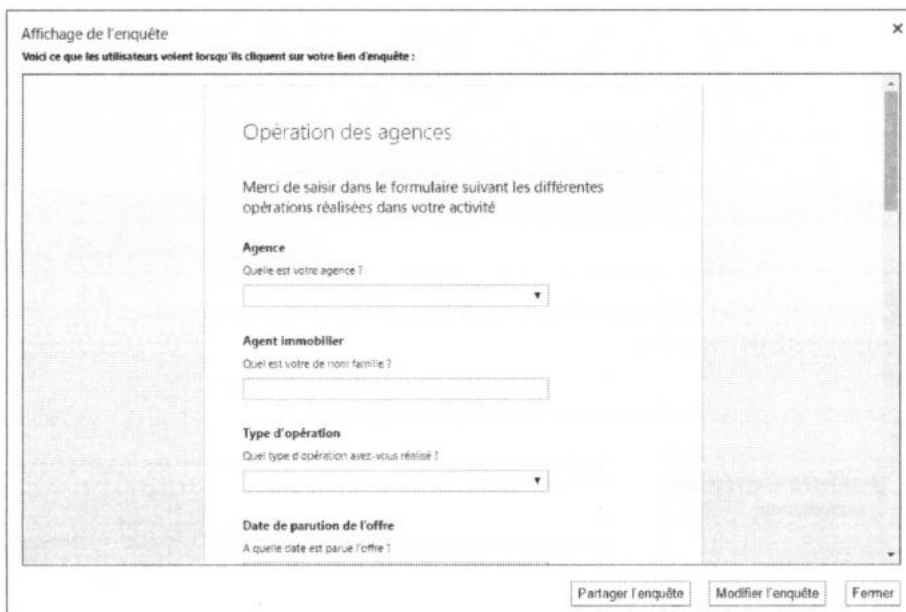
Question	Question	Sous-titre de la question	Valeur attendue
2	Agent immobilier	Quel est votre de nom famille ?	Texte
3	Type d'opération	Quel type d'opération avez-vous réalisé ?	Choix Vente ou Location 
4	Date de parution de l'offre	À quelle date est parue l'offre ?	Date 
5	Date de signature	À quelle date l'opération a-t-elle été signée ?	Date
6	Type de bien	De quel type de bien s'agit-il ?	Choix : Maison, Loft, Villa, Appartement 
7	Surface	Quelle est la surface du bien ?	Décimale fixe 

Question	Question	Sous-titre de la question	Valeur attendue
8	Commune	Quel est le nom de la ville du bien ?	Texte
9	Code postal	Quel est le code postal de la ville ?	Texte
10	Avec parking	Le bien possède-t-il un parking ?	Choix : VRAI ou FAUX 
11	Prix proposé	Quel est le prix du bien lors de la parution de l'offre ?	Nombre avec Format devise sans décimale. 
12	Prix final	Quel est le prix du bien lors de la signature ?	Nombre avec Format devise sans décimale.
13	Honoraires	Quel est le montant des honoraires ?	Nombre avec Format devise sans décimale.
14	Nombre de visites	Combien y a-t-il eu de visite ?	Nombre sans décimale
15	Nombre d'offres	Combien d'offre a-t-on reçu ?	Nombre sans décimale
16	Opération réussie	L'opération a-t-elle été réussie ?	Choix : VRAI ou FAUX

 L'enquête permet d'avoir un choix OUI ou NON qui aurait pu être approprié pour les questions portant sur le Parking et Opération réussie. Toutefois, il subsiste un problème sur la version actuelle : les valeurs remontées par le choix OUI ou NON semblent ne pas être constantes : « Yes / Oui » ou « No/Non ». Les valeurs « VRAI » et « FAUX » sont interprétées sans erreur dans Excel ce qui rend ce choix plus pertinent que le « OUI ou NON ».

- ☞ Cliquez sur **Enregistrer et afficher** pour générer votre enquête et permettre une première saisie.

Cela s'affiche ainsi :



Affichage de l'enquête

Voici ce que les utilisateurs voient lorsqu'ils cliquent sur votre lien d'enquête :

Opération des agences

Merci de saisir dans le formulaire suivant les différentes opérations réalisées dans votre activité

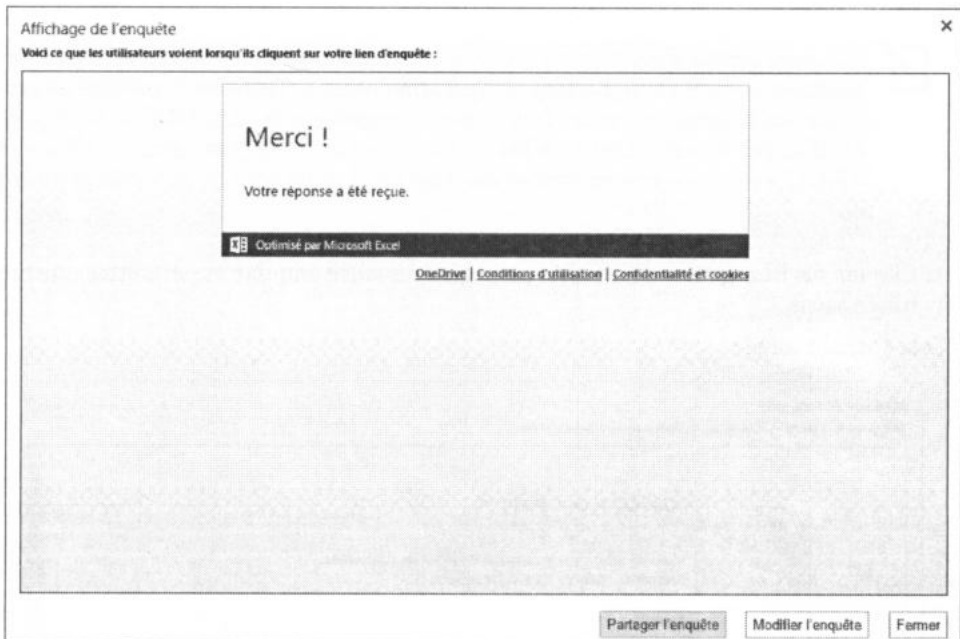
Agence
Quelle est votre agence ?

Agent immobilier
Quel est votre de nom famille ?

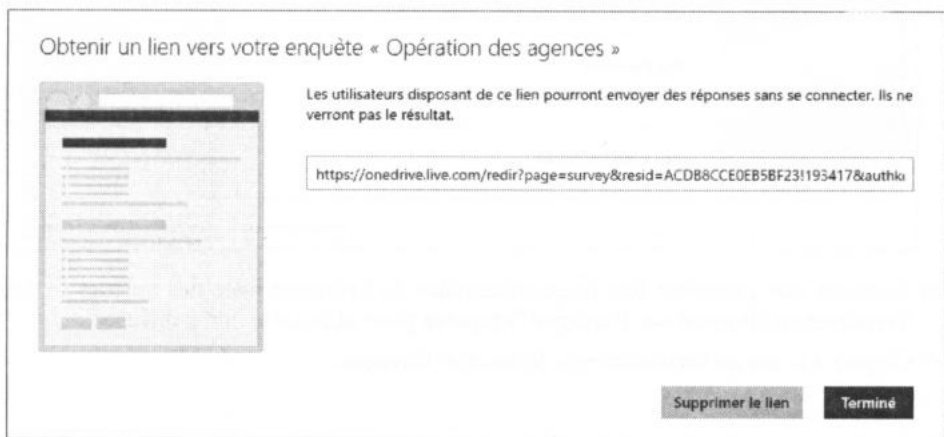
Type d'opération
Quel type d'opération avez-vous réalisé ?

Date de parution de l'offre
A quelle date est parue l'offre ?

- ☞ Saisissez une première fois le questionnaire de l'enquête avec des valeurs fictives. Terminez en cliquant sur **Partager l'enquête** pour obtenir le lien à diffuser.
- ☞ Cliquez à la fin du formulaire sur le bouton **Envoyer**.



- ☞ Cliquez sur **Créer un lien** pour obtenir le lien de l'enquête. Vous pouvez l'envoyer aux utilisateurs de l'enquête.



- ☞ Cliquez sur **Terminé** puis consultez le fichier Excel de l'enquête pour constater que les réponses sont bien intégrées.

A	B	C	D	E	F	G	H	I	J	K	L
Agence	Agencé/destinataire	Type d'opération	Date de passage de	Date de signature	Type de bien	Surface	Commune	Cofin. partiel	Avec parking	Prix proposé	Prix de compromis
Paris	Eri	Vente	15/12/2015	13/03/2016	Maison	145	Argentan-Sarre	92200	VRA	673 000 €	636 600 €

c. Envoyer un e-mail avec les statistiques des ventes aux agences

Objectif

L'objectif de cette partie est de récupérer les données mensuelles puis d'envoyer un e-mail automatique aux agences.

L'utilisateur peut saisir dans la feuille **Paramètres** les destinataires du mail et le mois choisi. Les données du mois correspondent aux signatures réalisées durant le mois désigné par l'utilisateur.

Un tableau croisé dynamique sera donc construit pour synthétiser les données des opérations réalisées. Il faudra ensuite récupérer les valeurs du TCD pour les envoyer par mail aux destinataires. Le corps du message doit ressembler à cela :

« Bonjour,

Voici les résultats du mois

Location à Paris : 40

Vente à Paris : 27

Location à Reims : 27

Vente à Reims : 13

Cordialement »

L'onglet **Paramètres** comporte cinq zones clés :

- ▶ La cellule **A2** compte le nombre d'enregistrements dans le tableau. Il évoluera au fur et à mesure des données qui seront ajoutées. La formule qui permet de compter le nombre d'enregistrements est un NB.SI portant sur l'ensemble de la colonne A avec comme critère « différent de vide ». Il faut retirer 1 à cette somme pour exclure la ligne de l'en-tête : $=NB.SI(Donnees!A:A;"<>")-1$.
- ▶ Les cellules de la colonne **B** contiennent les adresses mail des destinataires.
- ▶ Les cellules **C2** et **C3** contiennent respectivement le mois et l'année sur lesquels nous allons filtrer les données.
- ▶ Le bouton **Envoyer les données du moi** sera utilisé pour générer le TCD et l'envoyer par mail aux destinataires.
- ▶ Enfin la cellule **H1** sera le coin supérieur gauche du futur TCD.

Création de la procédure

- ❖ Revenez sous Excel dans le fichier **Enoncé_6-B.xlsm**.
- ❖ Ouvrez Visual Basic Editor puis créez un nouveau module.
- ❖ Démarrez une nouvelle procédure : **GenTCDMail**.

Si un TCD existe déjà sur la plage H1:O10, il faut le supprimer pour pouvoir en créer un nouveau. Pour cela il faut supprimer le contenu des cellules H1:O10 de la feuille **Paramètres** :

```
Sheets("Paramètres").Select
For Each Sh In ActiveSheet.Shapes
    Sh.Delete
Next
```

- ❖ Créez un nouveau TCD en se basant sur la valeur de **A2** pour connaître l'étendue de la plage. Positionnez le TCD **TCD_Mail** sur la cellule H1.

```
ActiveWorkbook.PivotCaches.Create(SourceType:=xlDatabase,
SourceData:="Donnees!R1C1:R" & Cells(2, 1).Value & "C16",
Version:=6).CreatePivotTable TableDestination:="Paramètres!R1C8",
TableName:="TCD_Mail", DefaultVersion:=6
```

- ❖ Ajoutez le champ **Type de bien** en tant qu'axe en ligne et l'agence en colonne :

```
With ActiveSheet.PivotTables("TCD_Mail").PivotFields("Type
d'opération")
    .Orientation = xlRowField
    .Position = 1
End With
With ActiveSheet.PivotTables("TCD_Mail").PivotFields("Agence")
    .Orientation = xlColumnField
    .Position = 1
End With
```

- ❖ Positionnez le champ **Agent immobilier** en tant que données pour compter le nombre d'opérations :

```
ActiveSheet.PivotTables("TCD_Mail").AddDataField
ActiveSheet.PivotTables("TCD_Mail").PivotFields("Agent immobilier"),
"Nombre opération", xlCount
```

- ❖ Ajoutez le champ **Date de signature** en tant que filtre de Page (xlPageField) :

```
With ActiveSheet.PivotTables("TCD_Mail").PivotFields("Date de
signature")
    .Orientation = xlPageField
    .Position = 1
End With
```

Envoyer les données mensuelles signifie que nous allons filtrer les données sur les lignes ayant une **Date de signature** qui correspond aux valeurs saisies par l'utilisateur dans la feuille **Paramètres**. Les cellules C2 et C3 de la feuille **Paramètres** contiennent respectivement le mois et l'année sur lesquels va se baser le filtre.

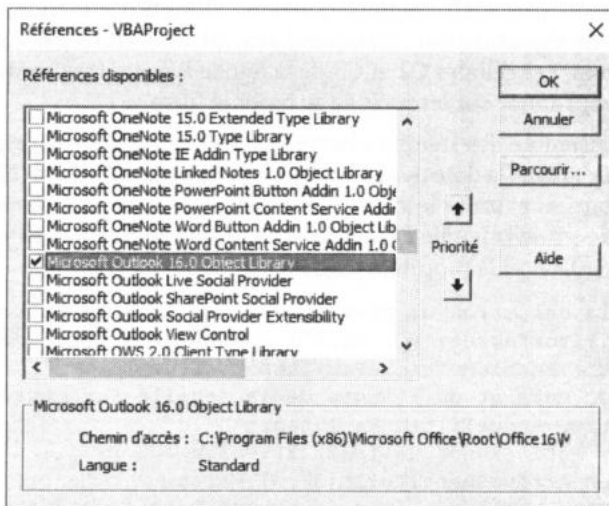
- ☞ Parcourez l'ensemble des items `PivotItems` du champ **Date de signature**. Chaque item a comme valeur "la date de signature" pour sa propriété `nom (Name)`. Récupérez cette date pour en extraire le mois et l'année. Si le mois et l'année récupérés sont égaux aux valeurs de la feuille **Paramètres**, la propriété d'affichage `Visible` de l'item est à vrai (`True`) sinon la propriété d'affichage est à faux (`False`).

```
'Autoriser la sélection de plusieurs items dans le filtre
ActiveSheet.PivotTables("TCD_Mail").PivotFields("Date de signature
du compromis").EnableMultiplePageItems = True
'Récupérer du mois et de l'année de la feuille Paramètres
Dim MoisFiltre, AnneeFiltre As Integer
MoisFiltre = ActiveSheet.Cells(2, 3).Value
AnneeFiltre = ActiveSheet.Cells(3, 3).Value
'Parcourir l'ensemble des items du champ Date de signature
du compromis
Dim ItemDate as Date
For Each it In ActiveSheet.PivotTables("TCD_Mail").PivotFields
("Date de signature du compromis").PivotItems
    'Valeur par défaut : visible à vrai
    it.Visible = True
    'Convertir le nom de l'item au format date
    ItemDate = Format(it.Name, "mm/dd/yyyy")
    'tester si le mois et l'année correspondent entre l'item et les
valeurs de la feuille. Si KO, passer la valeur de visible à Faux
    If Month(ItemDate) <> MoisFiltre Or Year(ItemDate) <> AnneeFiltre
Then
        it.Visible = False
    End If
Next
```

Création du message

Une fois le tableau croisé dynamique créé, il convient de positionner les valeurs dans le corps du message à envoyer aux destinataires.

- ☞ Ajoutez la référence à Microsoft Outlook dans les références. Cliquez sur le menu **Outils - Références**. Cochez la case Microsoft Outlook XX Object Library correspondant à votre version.



- ☞ Ouvrez l'application Outlook via VBA Excel pour la manipuler et créez un nouveau mail.

```
'Création de la variable application Outlook
Dim Email As Outlook.Application
'Création de la variable Mail Outlook
Dim EmailMsg As Outlook.MailItem
'Initialisation de l'objet application OutlookSet Email =
CreateObject("Outlook.Application")
'Création du mail dans l'objet application Outlook
Set EmailMsg = Email.CreateItem(olMailItem)
```

- ☞ Faites une boucle de type `While ... Wend` pour ajouter les destinataires. Des destinataires ont été mis par défaut, mais vous pouvez librement ajouter des destinataires de votre choix pour réceptionner les messages envoyés.

```
' Boucle pour ajouter tous les destinataires saisis en cellule
Dim Ligne As Integer
Ligne = 2
While ActiveSheet.Cells(Ligne, 2).Value <> ""
EmailMsg.Recipients.Add ActiveSheet.Cells(Ligne, 2).Value
Ligne = Ligne + 1
Wend
```

- ☞ Ajoutez le titre du message **Résultat du mois** :

```
'Ajout d'un sujet à l'email
EmailMsg.Subject = "Résultat du mois"
```

- ❏ Récupérez les valeurs du TCD situées sur la plage I5:J6 pour les insérer dans le corps du message. Effectuez une concaténation du texte brut et des champs grâce à l'opérateur &. Le caractère 13 est celui qui permet le saut de ligne. Insérez Chr(13) pour chaque saut de ligne.

```
'Ajout d'un contenu à l'email
EmailMsg.Body = "Bonjour," & Chr(13) & "Voici les résultats du mois" &
Chr(13) & _
"Location à Paris : " & ActiveSheet.Cells(5, 9).Value & Chr(13) & _
"Vente à Paris : " & ActiveSheet.Cells(6, 9).Value & Chr(13) & _
"Location à Reims : " & ActiveSheet.Cells(5, 10).Value & Chr(13) & _
"Vente à Reims : " & ActiveSheet.Cells(6, 10).Value & Chr(13) & _
"Cordialement"
```

- ❏ Envoyez le message.

```
EmailMsg.Send
```

- ❏ Détruisez les variables pour libérer de l'espace.

```
'Destruction des variables
Set EmailMsg = Nothing
Set Email = Nothing
```

- ❏ Affichez un pop-up pour la fin du traitement.

```
'Affichage du pop-up d'information
MsgBox "Traitement terminé"
```

Voici l'ensemble de la procédure GenTCDMail :

```
Sub GenTCDMail()
'Effacer le TCD si existant
Sheets("Paramètres").Select
For each sh In Active Sheet.Shapes
Sh.Delete
Next
'Création du TCD en cellule H1
ActiveWorkbook.PivotCaches.Create(SourceType:=xlDatabase,
SourceData:="Donnees!R1C1:R" & Cells(2, 1).Value & "C16",
Version:=6).CreatePivotTable TableDestination:="Paramètres!R1C8",
TableName:="TCD_Mail", DefaultVersion:
=6
'Ajout du champ type d'opération en ligne
With ActiveSheet.PivotTables("TCD_Mail").PivotFields("Type
d'opération")
.Orientation = xlRowField
.Position = 1
End With
'Ajout du champ Agence en colonne
With ActiveSheet.PivotTables("TCD_Mail").PivotFields("Agence")
```

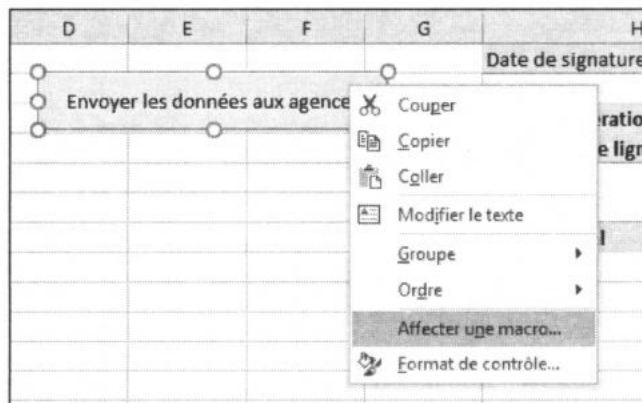
```

        .Orientation = xlColumnField
        .Position = 1
    End With
    'Ajout du nombre d'opérations
    ActiveSheet.PivotTables("TCD_Mail").AddDataField
    ActiveSheet.PivotTables("TCD_Mail").PivotFields("Agent immobilier"),
    "Nombre opération", xlCount
    'Ajout du champ de filtre de page
    With ActiveSheet.PivotTables("TCD_Mail").PivotFields("Date de
    signature du compromis")
        .Orientation = xlPageField
        .Position = 1
    End With
    'Autorisation de la sélection de plusieurs items dans le filtre
    ActiveSheet.PivotTables("TCD_Mail").PivotFields("Date de signature
    du compromis").EnableMultiplePageItems = True
    'Récupération du mois et de l'année de la feuille Paramètres
    Dim MoisFiltre, AnneeFiltre As Integer
    MoisFiltre = ActiveSheet.Cells(2, 3).Value
    AnneeFiltre = ActiveSheet.Cells(3, 3).Value
    'Parcours de l'ensemble des items du champ Date de signature du
    compromis
    Dim ItemDate As Date
    For Each it In ActiveSheet.PivotTables("TCD_Mail").PivotFields
    ("Date de signature du compromis").PivotItems
        'Valeur par défaut : visible à vrai
        it.Visible = True
        'Convertir le nom de l'item en date
        ItemDate = Format(it.Name, "mm/dd/yyyy")
        'Tester si le mois et l'année correspondent entre l'item et
        les valeurs de la feuille. Si KO, passer la valeur de visible à Faux
        If Month(ItemDate) <> MoisFiltre Or Year(ItemDate) <> AnneeFiltre
        Then
            it.Visible = False
        End If
    Next
    'Création de la variable application Outlook
    Dim Email As Outlook.Application
    'Création de la variable Mail Outlook
    Dim EmailMsg As Outlook.MailItem
    'Initialisation de l'objet application Outlook
    Set Email = CreateObject("Outlook.Application")
    'Création du mail dans l'objet application Outlook
    Set EmailMsg = Email.CreateItem(olMailItem)
    ' Boucle pour ajouter tous les destinataires saisis en cellule
    Dim Ligne As Integer

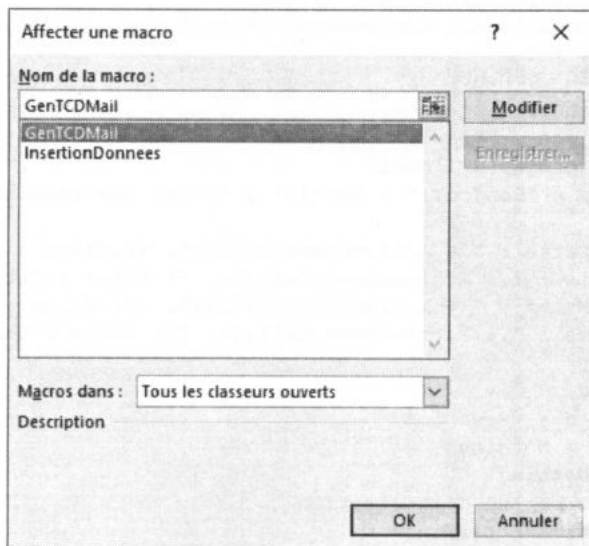
```

```
Ligne = 2
While ActiveSheet.Cells(Ligne, 2).Value <> ""
EmailMsg.Recipients.Add ActiveSheet.Cells(Ligne, 2).Value
Ligne = Ligne + 1
Wend
'Ajout d'un sujet à l'email
EmailMsg.Subject = "Résultat du mois"
'Ajout d'un contenu à l'email
EmailMsg.Body = "Bonjour," & Chr(13) & "Voici les résultats du mois"
& Chr(13) & _
"Location à Paris : " & ActiveSheet.Cells(5, 9).Value & Chr(13) & _
"Vente à Paris : " & ActiveSheet.Cells(6, 9).Value & Chr(13) & _
"Location à Reims : " & ActiveSheet.Cells(5, 10).Value & Chr(13) & _
"Vente à Reims : " & ActiveSheet.Cells(6, 10).Value & Chr(13) & _
"Cordialement"
EmailMsg.Send
'Destruction des variables
Set EmailMsg = Nothing
Set Email = Nothing
'Affichage du pop-up d'information
MsgBox "Traitement terminé"
End Sub
```

- ☞ Reliez la procédure au bouton de la feuille **Paramètres**. Assurez-vous d'être en **Mode Création** puis faites un clic droit sur le bouton **Envoyer les données aux agences**. Sélectionnez **Affecter une macro...**



☞ Dans la liste des macros, choisissez la macro GenTCDMail puis cliquez sur OK.



☞ Pour constater l'envoi du message, allez dans le dossier Éléments envoyés de Microsoft Outlook.

C**CELLULE**

Commentaires.....	273
Format pourcentage.....	43
Liste déroulante.....	25
Message d'erreur.....	25
Message d'information.....	25
Nommer.....	36
Protéger.....	137, 149
Validation.....	25

CODE

Voir VBA

CONTRÔLE

Bouton.....	130, 148, 227
CheckBox.....	99
ComboBox.....	99
CommandButton.....	99
Création dynamique.....	258
Créer.....	96
Label.....	97
ListBox.....	99
TextBox.....	99

Voir aussi FORMULAIRE

D**DATE**

Format TimeStamp.....	299
-----------------------	-----

E**ENQUÊTE**

Voir OFFICE

ÉVÈNEMENT

Définition.....	101
-----------------	-----

EXCEL

Formules avancées.....	21
Gestionnaire de noms.....	36
Matrice.....	35
OpenOffice.org.....	17
Personnaliser le ruban.....	162, 312
Tableau de données.....	67
Version Mac.....	17
Versions.....	15

Voir aussi OFFICE

F**FEUILLE**

Afficher/masquer.....	133, 142, 270
Insérer.....	44
Protéger.....	137
Protéger la structure.....	135, 143, 270

FONCTION

CENTILE.....	70
Créer.....	120
DATE.....	42
EQUIV.....	30, 38, 271
EST.ERR.....	34, 44
EST.ERREUR.....	34, 44
EST.PAIR.....	238
ESTNUM.....	43
FIN.MOIS.....	236
INDEX.....	30, 39, 271
JOURSEM.....	250
MOYENNE.....	48, 70
NB.JOURS.OUVRES.....	235
NB.SI.....	33, 49, 255
NB.SI.ENS.....	34, 47, 51, 175, 242
RECHERCHEH.....	29
RECHERCHEV.....	29, 44

SERIE.JOUR.OUVRE	236, 241
SI	32, 42
SOMME	48
SOMME.SI, SOMME.SI.ENS	33, 47

FORMAT DE FICHIER

XLSM	80
------------	----

FORMULAIRE

Créer.....	92, 143
Initialisation.....	105
Propriétés	94
<i>Voir aussi CONTRÔLE</i>	

FORMULE

Analyse	59
Concaténation	51
Évaluation	59
Exemple.....	21
Figurer les lignes et/ou colonnes d'une cellule	240
Gestion des erreurs	34
Matricielle.....	35
Saisie et recherche de formules	57
Statistique	69
Tableau de données	68
Test de condition.....	32
<i>Voir aussi FONCTION</i>	

G**GRAPHIQUE**

Axe principal, axe secondaire	54
Camembert.....	187
Courbe	74, 176
Créer avec VBA.....	207
Étiquettes de données.....	182, 187
Histogramme empilé.....	181
Insérer	52
Mettre en forme	54

Mise en forme de l'axe	55
Modification de la source de données	177
Sparkline.....	80, 75
Style de graphique.....	188

L**LISTE DÉROULANTE**

Dans une cellule	25
------------------------	----

M**MACRO**

Affecter	248
Enregistrement.....	203, 210, 218
Exécution.....	92

**MISE EN FORME
CONDITIONNELLE**

Avec formule	237, 250
Barre de données.....	64
Gestion des règles	252
Jeux d'icônes	66
Mise en surbrillance des cellules...62, 73	
Nuances de couleurs.....	65
Valeur les +/- élevées.....	63, 71

O**OFFICE**

Création d'enquête	317, 326
Créer un compte OneDrive	326
Envoyer	333
OneDrive	316

Restitution des données d'enquête
sur OneDrive317, 335

P

POWERPOINT

Manipuler avec VBA209

PROCÉDURE

Définition101

T

TABLEAU CROISÉ DYNAMIQUE

Actualiser210
Assistant162, 189
Axe161
Champs calculés169
Créer159, 172, 178, 184, 194
Créer avec VBA205
Définition159
Données161
Éléments calculés169
Filtre de page168
Filtre sur axe180

TABLEAU DE DONNÉES

Créer68
Formulaire de saisie311, 322
Formule68, 69
Mise en forme68
Modifier une donnée323
Rechercher324

TCD

Voir TABLEAU CROISÉ DYNAMIQUE

V

VARIABLE

Déclaration292
Définition104
Initialisation110
Mettre à jour122
Objet : application Excel287, 292
Objet : application Outlook319, 338

VBA

Affichage dans la console122
Ajout, comptage
et suppression d'item124
Boîte de dialogue ... 114, 263, 288, 293, 304
Boucle For244, 275
Boucle For Each268
Boucle While262, 295
Boucles106, 288
Call227
Chart207, 216
Chart Axes, Line216
ChartDataSource207, 216
ChartElement207, 216
ChartType207, 216
Clear215, 264
Close210, 304
Collection83
Commentaires de cellules Excel273
Control.Add258
Contrôles84, 145, 258, 259
Conversion de données300
Conversion de données CInt, CStr ...269
Conversion de valeurs112
Copier-coller211
DataField206, 215
Débogage92
Delete227
Événements88, 101
FileDialog288, 293
Fonctions87

Fonctions de texte Left, Len.....	303	ShapeRange	221
Fonctions Excel.....	289	ShapeStyle.....	221
Format de cellule	289, 300	Sort.....	274
Formulaires	84, 143, 256	Structure conditionnelle : IF	108
Formule :instr.....	125	Tableau.....	259, 272
GetOpenFileName	288	Tag.....	264
If Then Else	246, 290, 297	Variable	83
Impression de feuilles	128	Variable : application Outlook..	319, 338
InputBox	262	Variable contrôle.....	258
Instructions.....	89	Versions d'Excel.....	15
LBound, UBound.....	260	Visual Basic Editor	90
Library	208, 223, 319, 335	Vocabulaire	13
Mail	321, 335	With End With.....	296
Mail.Body.....	321, 339		
Mail.Send.....	321, 339		
Mail.Subject	321, 338		
Mail.To	321, 339		
Méthodes.....	83		
Modules	86, 103		
MsgBox.....	114, 263, 288, 294, 304		
ObjectApplication.....	209, 223		
Objet et classe.....	82		
Options Base.....	260		
Page, MultiPage	263		
PivotCache.....	205, 215		
PivotTable.....	205, 215		
PivotTable DataSource.....	206, 216		
PivotTable Filter	207		
Position	265		
PowerPoint : présentation,			
slide	209, 223		
Procédures	86, 101, 103, 227		
Propriétés	82		
Protéger le code.....	140, 152		
Quit	210		
Raccourcis-clavier	14		
Range	204		
Redim, Preserve	259		
Relier événements et procédures	129		
Save, SaveAs.....	210		
Select Case	245, 290, 303		
Selection.....	204, 215		
Shape.....	207, 216		

Apprenez le langage VBA et devenez un expert sur Excel

Visual Basic for Application (**VBA**) est un langage de programmation mis en place afin d'apporter des fonctionnalités supplémentaires aux outils de la suite Microsoft Office et plus particulièrement à Excel en permettant l'automatisation des calculs dans le tableur mais également :

- La création de fonctions gérées comme les fonctions Excel natives.
- La création de formulaire permettant à l'utilisateur d'interagir avec l'application.
- La possibilité d'implémenter de nombreuses fonctionnalités permettant par exemple : d'envoyer un e-mail, de créer un rapport Power-Point, d'imprimer un document, de lancer une application, d'ouvrir un fichier, de modifier des paramètres Windows...

VBA est un langage accessible c'est-à-dire qu'il ne requiert pas de connaissances poussées en programmation.

L'objectif de ce livre est de vous apprendre à utiliser le **langage VBA** et à développer vos compétences sur **Excel**. Il a été rédigé avec la version 2016 d'Excel.

Après une brève introduction à VBA, vous commencerez par utiliser des **fonctions avancées d'Excel** (validation des données, calculs sur les dates, fonctions conditionnelles, calcul matriciel, mise en forme conditionnelle...), vous serez amené à créer un formulaire de saisie des ventes puis à interagir avec Excel grâce au langage VBA. Vous utiliserez ensuite les **tableaux et graphiques croisés dynamiques** via Excel et VBA pour gérer le suivi d'une campagne de test d'une application de vente en ligne. L'exercice suivant se base sur les **fonctions de date** pour calculer des durées et le coût de chaque tâche d'un projet informatique. Dans le dernier chapitre, l'exemple traité vous permettra de **consolider des données**, d'automatiser la saisie de données en ligne et l'envoi de données mensuelles par e-mail pour plusieurs agences immobilières.

L'approche utilisée est basée principalement sur des exemples extraits de la vie professionnelle. Chaque chapitre correspond à un **cas métier** et se décompose en **notions de cours** et explications permettant de réaliser l'**exercice**. L'objectif est d'être guidé le plus possible et de mettre tout de suite en pratique les notions de cours.

Les classeurs nécessaires à la réalisation des exercices (énoncé) et les versions corrigées sont disponibles en téléchargement sur le site des Editions ENI www.editions-eni.fr.

Solutions Business



www.editions-eni.fr



Jean-Emmanuel CHAPARTEGUI

Après obtention d'un Master en Sciences de Gestion suivi d'une spécialisation en Gestion des Systèmes d'Information, **Jean-Emmanuel CHAPARTEGUI** a enseigné pendant 4 ans à l'université Paris Dauphine un cours portant sur Excel/VBA pour les contrôleurs de gestion. Consultant en Système d'Information, il travaille principalement dans la gestion de projet, la conception fonctionnelle d'application et dans les tests. Ses nombreuses missions lui ont permis d'utiliser Excel/VBA dans des contextes très variés lui offrant aujourd'hui la possibilité d'écrire ce livre et vous transmettre ainsi son expérience.



sur www.editions-eni.fr :

→ Classeurs des énoncés
et corrigés des exercices

Pour plus
d'informations :



ISBN : 978-2-409-00326-4

21,95 €



9 782409 003264