

## Contrôle des valeurs saisies

Questions :

1. *Retrouver les valeurs aberrantes que l'utilisateur cherche à saisir.*
  - La date de début de session est antérieure à la date du jour
  - Le nombre de participants est négatif
  - Le coût d'hébergement est beaucoup trop élevé en comparaison des autres sessions
2. *Saisir les valeurs dans l'application.*
3. *Quels sont les contrôles effectués au moment de la saisie sur chacun des champs ?*

Champ	Contrôle
Activité	Pas de saisie, choix dans une liste
Action de l'activité	Pas de saisie, choix dans une liste
Date	Vérification du type de données et de la validité d'une date par le logiciel Access
Nombre de participants	Impossible de saisir une valeur < 0, les valeurs décimales sont automatiquement arrondies à la valeur la plus proche.
Ville	Pas de saisie, choix dans une liste
Coût d'hébergement prévu	Contrôle du type de données

Il existe donc deux types de contrôles : d'une part un contrôle sur le type de données saisies pris en charge par le logiciel et, d'autre part, un contrôle de vraisemblance pris en charge par le code de l'application.

Le premier permet de conserver l'intégrité et la cohérence de la base, le second de respecter les règles de gestion fixées à l'application.

4. *Quels sont les champs qui acceptent cependant des valeurs erronées ?*  
Tous les champs qui n'offrent pas un choix dans une liste déroulante.

5. *Donner des exemples de valeurs incorrectes que l'on peut saisir.*

Champ	Erreurs possibles
Date	Saisie d'une date antérieure à la date du jour saisie d'une date en dehors de la période autorisée (année civile)
Nombre de participants	Aucune limite maximale
Coût d'hébergement prévu	Aucun plafond

6. *Quels sont les contrôles de validité déjà pris en charge par le code associé à l'événement « Avant mise à jour » ?*

Ligne 3 : si le numéro est vide c'est une première saisie sinon c'est une modification, il ne s'agit donc pas d'un contrôle de validité  
Ligne 5 et 6 : compte le nombre d'inscrits actuels dans la table inscription  
Ligne 8 à 11 : vérifie s'il n'y a pas déjà davantage d'inscrit que le nouveau nombre maximum qu'on veut enregistrer

7. *Quels sont les autres contrôles que doit prendre en charge l'application sur chacun des champs du formulaire ?*

Contrôles	Contrôle de domaine de valeur	Contrôle de vraisemblance
dateSession	Vérifier type date	La date de sessions doit être supérieure à la date du jour et comprise dans l'année en cours
nombreMaxParticipants	Entier	Le nombre de participants doit être Compris entre 1 et 10
idLieu	Rien à faire	Rien à faire
coutPrevu	Montant décimal	Vérifier que le coût prévu par la création d'une nouvelle session ne va pas entraîner un dépassement du montant prévu par le budget de l'action

## Évolution de l'application pour prendre en charge de nouveaux besoins

Pour les besoins de cette partie les élèves doivent disposer des réponses de la première partie.

1. Rechercher dans l'aide la définition des deux premiers événements (touche F1 lorsque le curseur est positionné sur la ligne d'un événement).

Nom de l'événement	Définition
Avant Mise à jour	L'événement BeforeUpdate se produit avant que les données modifiées d'un <u>contrôle</u> ou d'un enregistrement soient <u>mises à jour</u> .
Après Mise à jour	L'événement AfterUpdate se produit après que les données modifiées d'un <u>contrôle</u> ou d'un enregistrement ont été <u>mises à jour</u> .

2. Parmi ces deux premiers événements quel est celui qui semble le plus approprié pour écrire le code qui permettra de tester la validité d'une date ?

On choisira le premier, sinon les données seront vérifiées après l'insertion des valeurs dans le champ de la table ce qui risque de générer un message de refus de la part du SGBD, message technique qui sera difficilement compréhensible par l'utilisateur.

Remarque : le nombre élevé d'événements proposés par l'environnement de développement permet d'envisager plusieurs solutions.

*Date saisie inférieure à la date du jour*

3. Rechercher dans l'aide s'il existe une fonction qui fournit la date du jour (date système)

Extrait de l'aide après avoir saisi les mots clefs « date système »

Date, fonction

Renvoie une valeur de type **Variant (Date)** contenant la date système actuelle.

4. Rédiger le message à afficher dans la boîte de dialogue pour avertir l'utilisateur de l'erreur.

Impossible cette date est déjà passée

5. En vous inspirant du code utilisé pour l'événement avant MAJ du contrôle `nombreMaxParticipant` `nombreMaxParticipant_BeforeUpdate(Cancel As Integer)`, rédiger le code qui permettra d'afficher le message d'erreur :

```
Private Sub dateSession_BeforeUpdate(Cancel As Integer)
If Me.dateSession.Value < Date Then
    MsgBox ("Impossible cette date " & Me.dateSession.Value & " est
d'jà passée ")
    Cancel = True
End If
```

6. Dans quel cas ce contrôle est-il utile ?

Ce contrôle est utile dans le cas d'une modification d'une session déjà créée.

7. Modifier le code de l'application pour prendre en charge cette nouvelle fonctionnalité.

Le code est ajouté en début de procédure car il s'applique aussi bien à une saisie qu'à une modification.

```
Private Sub nombreMaxParticipant_BeforeUpdate(Cancel As Integer)

    Dim nbInscrits As Integer

    If Me.nombreMaxParticipant < 1 Then
        MsgBox ("Le nombre de participants ne peut pas être négalatif ou
nul")
        Cancel = True
    End If
End Sub
```

8. Combien de participants une session peut-elle accueillir ?

Pour répondre, vous rédigerez la requête d'interrogation de la table session qui détermine le plus grand nombre de participants d'une session puis vous l'exécuterez.

```
SELECT max(nombreMaxParticipant) as plusGrandNombreParticipants
FROM session;
Réponse : 10
```

9. Modifier le code de l'application pour tenir compte de cette nouvelle contrainte.

Code à ajouter à la suite du précédent code inséré à la question 6 :

```
If Me.nombreMaxParticipant > 10 Then
    MsgBox ("Le nombre de participants excède le maximum
    autorisé (10)")
    Cancel = True
End If
```

10. Que faut-il modifier pour ne pas empêcher la saisie d'une valeur maximale erronée ?

Il suffit de supprimer la ligne :

```
Cancel = True
```

11. Rédiger le message à afficher.

Le nombre de participants excède le maximum autorisé (10).

Choisir OK pour conserver ce nombre ou Annuler pour saisir une autre valeur.

12. Rechercher dans l'aide du logiciel les paramètres à employer pour afficher plusieurs boutons et récupérer la valeur du bouton choisi par l'utilisateur.

Extrait de l'aide du logiciel

**Syntaxe**

**MsgBox(prompt[, buttons] [, title] [, helpfile, context])**

La syntaxe de la fonction **MsgBox** comprend les arguments nommés suivants :

Élément	Description
<b>prompt</b>	Expression de chaîne affichée comme message dans la boîte de dialogue. La longueur maximale de l'argument <b>prompt</b> est d'environ 1 024 caractères selon la largeur des caractères utilisés. Si l'argument <b>prompt</b> occupe plus d'une ligne, n'oubliez pas d'insérer un retour chariot ( <b>Chr(13)</b> ) ou un saut de ligne ( <b>Chr(10)</b> ) entre les lignes, ou une combinaison de caractères retour chariot-saut de ligne ( <b>Chr(13) &amp; Chr(10)</b> ).
<b>buttons</b>	Facultatif. Expression numérique qui représente la somme des valeurs indiquant le nombre et le type de boutons à afficher, le style d'icône à utiliser, l'identité du bouton par défaut, ainsi que la modalité du message. Si l'argument <b>buttons</b> est omis, sa valeur par défaut est 0.
<b>title</b>	Facultatif. Expression de chaîne affichée dans la barre de titre de la boîte de dialogue. Si l'argument <b>title</b> est omis, le nom de l'application est placé dans la barre de titre.

**Valeurs**

L'argument **buttons** prend les valeurs suivantes :

Constant	Value	Description
<b>vbOKOnly</b>	0	Affiche le bouton <b>OK</b> uniquement.
<b>vbOKCancel</b>	1	Affiche les boutons <b>OK</b> et <b>Annuler</b> .
<b>vbAbortRetryIgnore</b>	2	Affiche le bouton <b>Abandonner</b> , <b>Réessayer</b> et <b>Ignorer</b> .
<b>vbYesNoCancel</b>	3	Affiche les boutons <b>Oui</b> , <b>Non</b> et <b>Annuler</b> .
<b>vbYesNo</b>	4	Affiche les boutons <b>Oui</b> et <b>Non</b> .
<b>vbRetryCancel</b>	5	Affiche les boutons <b>Réessayer</b> et <b>Annuler</b> .
<b>vbCritical</b>	16	Affiche l'icône <b>Message critique</b> .
<b>vbQuestion</b>	32	Affiche l'icône <b>Requête d'avertissement</b> .
<b>vbExclamation</b>	48	Affiche l'icône <b>Message d'avertissement</b> .
<b>vbInformation</b>	64	Affiche l'icône <b>Message d'information</b> .
<b>vbDefaultButton1</b>	0	Le premier bouton est le bouton par défaut.
<b>vbDefaultButton2</b>	256	Le deuxième bouton est le bouton par défaut.
<b>vbDefaultButton3</b>	512	Le troisième bouton est le bouton par défaut.
<b>vbDefaultButton4</b>	768	Le quatrième bouton est le bouton par défaut.

<b>vbApplicationModal</b>	0	Boîte de dialogue modale. L'utilisateur doit répondre au message affiché dans la zone de message avant de pouvoir continuer de travailler dans l'application en cours.
<b>vbSystemModal</b>	4096	Modal système. Toutes les applications sont interrompues jusqu'à ce que l'utilisateur réponde au message affiché dans la zone de message.
<b>vbMsgBoxHelpButton</b>	16384	Ajoute le bouton Aide à la zone de message.
<b>VbMsgBoxSetForeground</b>	65536	Indique la fenêtre de zone de message comme fenêtre de premier plan.
<b>vbMsgBoxRight</b>	524288	Le texte est aligné à droite.
<b>vbMsgBoxRtlReading</b>	1048576	Indique que le texte doit apparaître de droite à gauche sur les systèmes hébraïques et arabes.

Le premier groupe de valeurs (0 à 5) décrit le nombre et le type de boutons affichés dans la boîte de dialogue. Le deuxième groupe (16, 32, 48 et 64) décrit le style d'icône. Le troisième groupe (0, 256 et 512) définit le bouton par défaut. Enfin, le quatrième groupe (0 et 4 096) détermine la modalité de la zone de message. Au moment d'additionner ces nombres pour obtenir la valeur finale de l'argument **buttons**, ne sélectionnez qu'un seul nombre dans chaque groupe.

**Note** Ces constantes sont indiquées par Visual Basic pour Applications. Vous pouvez par conséquent les utiliser n'importe où dans votre code à la place des valeurs réelles correspondantes.

#### Valeurs renvoyées

Constante	Valeur	Description
<b>vbOK</b>	1	<b>OK</b>
<b>vbCancel</b>	2	<b>Annuler</b>
<b>vbAbort</b>	3	<b>Abandonner</b>
<b>vbRetry</b>	4	<b>Réessayer</b>
<b>vbIgnore</b>	5	<b>Ignorer</b>
<b>vbYes</b>	6	<b>Oui</b>
<b>vbNo</b>	7	<b>Non</b>

#### 13. Modifier le code de l'application pour afficher ce message

```
reponse = MsgBox("Le nombre de participants excède le maximum autorisé
(10) " _
& vbCrLf & "Choisir OK pour conserver ce nombre ou Annuler pour saisir
une autre valeur", _
vbOKCancel + vbCritical, "Saisie d'un nombre de participants trop
élevé")
If reponse = 1 Then
    Cancel = False
Else
    Cancel = True
End If
```

#### Remarques :

- une réponse est attendue par l'utilisateur, il faut donc utiliser *MsgBox* comme une fonction qui renvoie une valeur.
- L'argument « *button* » peut être exprimé sous la forme d'un nombre composé de la somme de plusieurs arguments ou bien sous la forme de littéraux ce qui est plus facile à lire
- La constante *vbCrLf* remplace avantageusement les caractères *chr(13)* et *chr(10)*.

#### 14. Modifier l'application pour prendre en compte cette nouvelle contrainte.

#### Objet à contrôler : txt\_cout

**Vérification à effectuer** : si la somme des coûts prévus pour l'ensemble des sessions de formation d'une action donnée (y compris la session à créer) est supérieure au budget prévu pour une action alors un message d'erreur doit le signaler à l'utilisateur.

On en profitera pour vérifier que le coût prévu est positif.

**Requête à effectuer** pour obtenir le coût des sessions déjà programmées :

```
SELECT sum(coutReel)
FROM Action, Session
WHERE Action.code=Session.codeAction
And Action.code="E0335"
```

**Code à ajouter dans l'application :**

```
Private Sub coutPrevu_BeforeUpdate(Cancel As Integer)
Dim rsCout As DAO.Recordset
Dim coutTotal As Single
Dim coutAction As Single

' recherche de la somme des couts déjà engagés pour cette action de
formation
SQL = "SELECT Sum(Session.coutReel) AS SommeDeCoutReel"
SQL = SQL & " FROM Action, Session"
SQL = SQL & " WHERE Action.code = Session.codeAction"
SQL = SQL & " AND Action.code= '" & Forms!gestionSessions!ldr_action
& "';"

Set rsCout = CurrentDb.OpenRecordset(SQL)

' on ajoute le cout de la session en saisie
coutTotal = rsCout("SommeDecoutReel") + Me.coutPrevu.Value

' on récupère le cout du budget de la formation
coutAction = Forms!gestionSessions!txt_cout

' on compare les 2 couts si le cout actuel est supérieur on alerte
l'utilisateur
If coutTotal > coutAction Then
    reponse = MsgBox("La création de cette session va entraîner
un dépassement budgétaire" _
        & vbCrLf & "Choisir OK pour créer cette session ou Annuler
pour saisir une autre valeur", _
        vbOKCancel + vbCritical, "Saisie d'un cout d'hébergement
prévu")
    If reponse = 1 Then
        Cancel = False
    Else
        Cancel = True
    End If
End If

End Sub
```