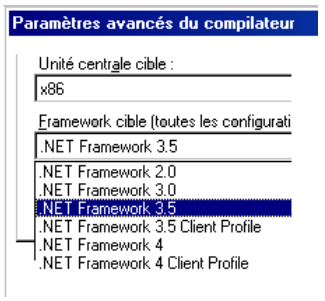
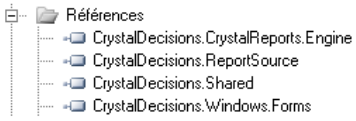
L'impression des données par CrystalReport

L'impression des données contenues dans un **`DataSet`** peut bien entendu être programmée par l'usage des outils d'impression étudiés précédemment. Mais il existe un autre composant qui mérite d'être abordé ici, même si ce n'est que de manière très sommaire. Il s'agit du *Crystal Report*, lequel se présente sous la forme du **`CrystalReportViewer`** grâce auquel l'utilisateur peut visualiser le document et en gérer l'impression. Ce document, quant à lui, est un **`CrystalReport`** présent sous la forme d'un fichier dont l'extension est **`.rpt`** ajouté à la solution en cours de développement de diverses manières dont voici la plus simple.
La première tentative d'insertion d'un *Composant Crystal Report* provoque le téléchargement et l'installation de l'outil à partir d'un site de SAP, le nouveau détenteur des droits sur ce produit.
Plusieurs réglages sont requis pour disposer des composants nécessaires au niveau du projet. Il faut :

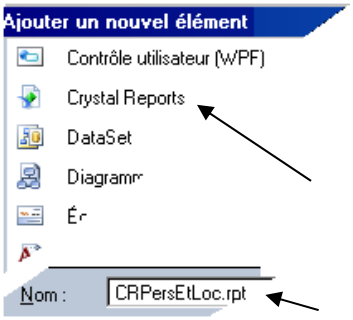
1. choisir un Framework cible dans Projet/Propriétés/Compiler/Options avancées …;
2. ajouter les références comme illustré ci-contre.

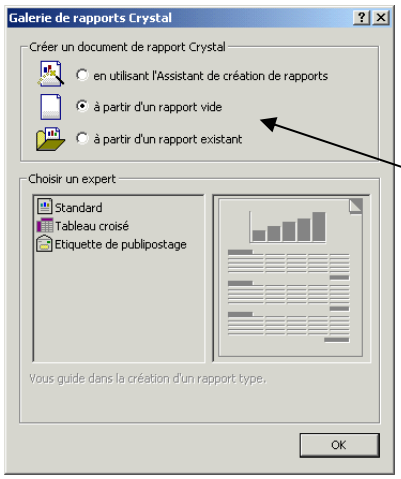




A noter qu'un Framework *Client* est limité et n'offre pas toutes les fonctionnalités nécessaires. Le composant **`CrystalReportViewer`** n'est pas disponible.

Le menu contextuel obtenu par un clic du bouton droit de la souris dans l'explorateur de solutions, permet notamment l'ajout d'un nouvel élément au projet.

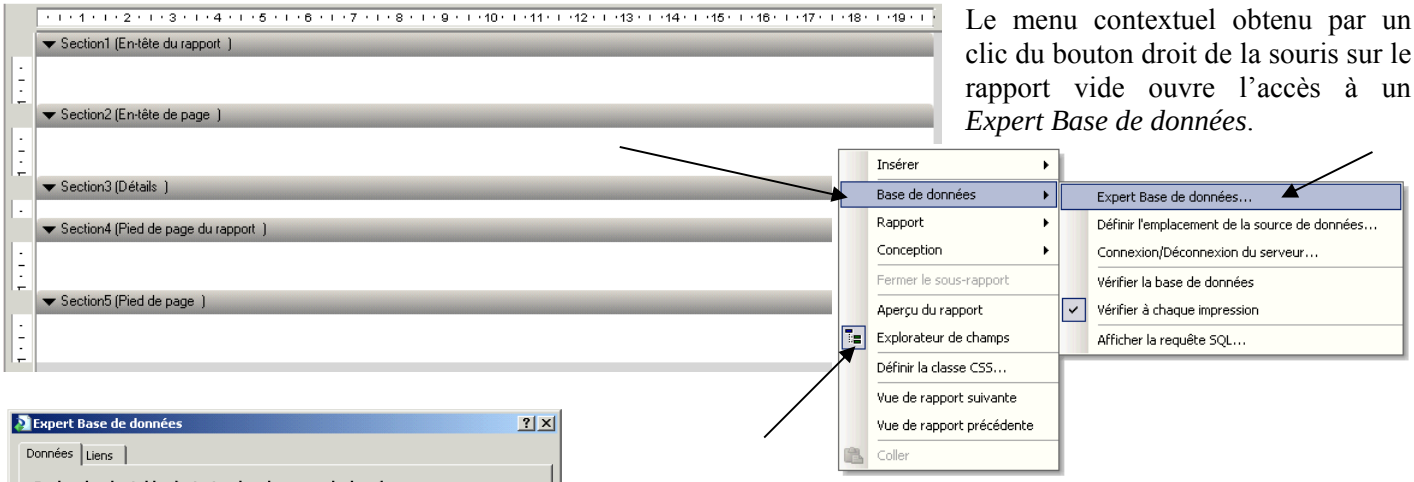




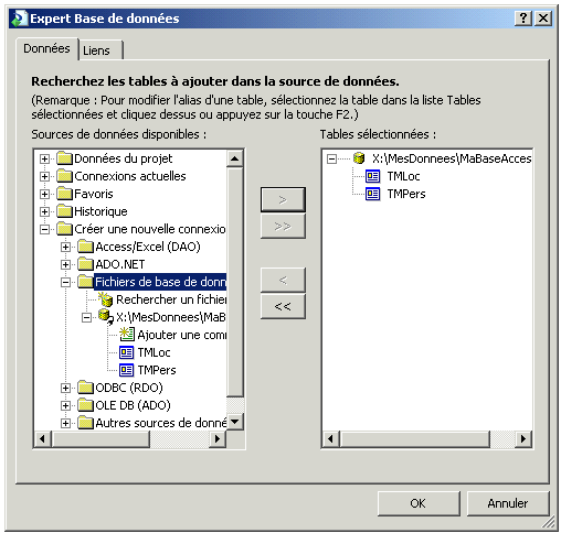
Il suffit de choisir dans la boîte dialogue illustrée ci contre l'élément de type *Rapport Crystal* et de le nommer de manière représentative de son rôle. Une application peut contenir de nombreux rapports.

La première boîte de dialogue qui s'ouvre alors offre le choix entre plusieurs modèles de rapport, ainsi que le choix de ne pas utiliser l'assistant de conception et de créer un rapport vide.

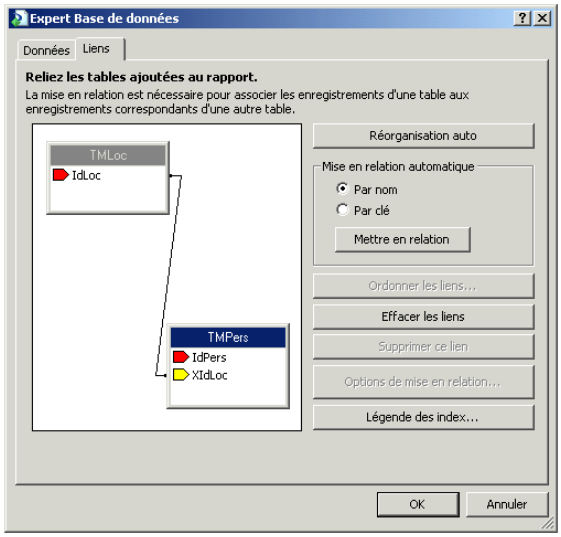
L'aide de l'assistant peut être utile pour l'élaboration de rapports plus évolués, mais pour un rapport standard, le plus simple est sans conteste de partir d'un rapport vide.



Le menu contextuel obtenu par un clic du bouton droit de la souris sur le rapport vide ouvre l'accès à un *Expert Base de données*.

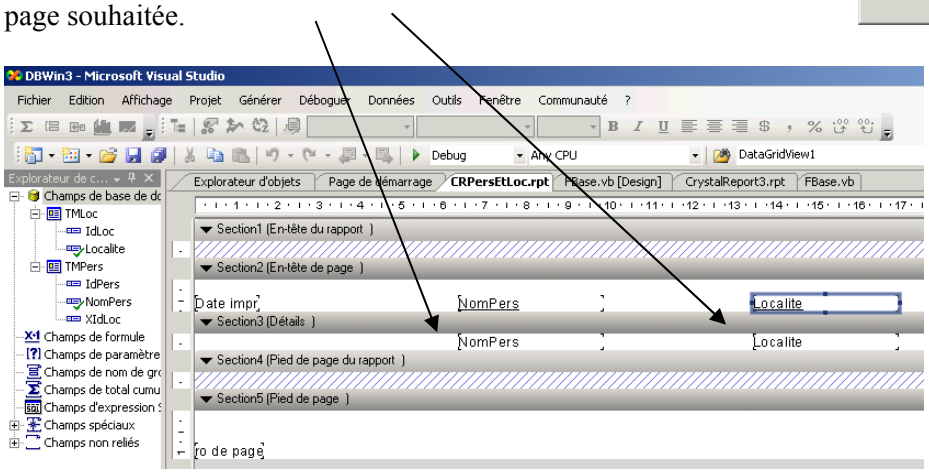


L'*Expert Base de données* ouvre une boîte de dialogue qui permet la sélection de diverses sources de données.
Pour l'exemple illustré ci contre, le choix s'est porté sur *Créer une nouvelle connexion/Fichiers de base de données* et a permis la désignation de la base des exemples précédents.
Les tables qui concernent le rapport envisagé sont ensuite glissées dans la liste étiquettée *Tables sélectionnées*.



L'onglet *Liens* de la boîte de dialogue permet d'établir les relations existant entre ces tables.

Ces opérations réalisées, l'*Explorateur de champs* accessible par le menu contextuel illustré ci-dessus propose les champs des tables sélectionnées ainsi que d'autres spécifiques à CrystalReport.
Ces champs peuvent être glissés sur le rapport vide selon la mise en page souhaitée.



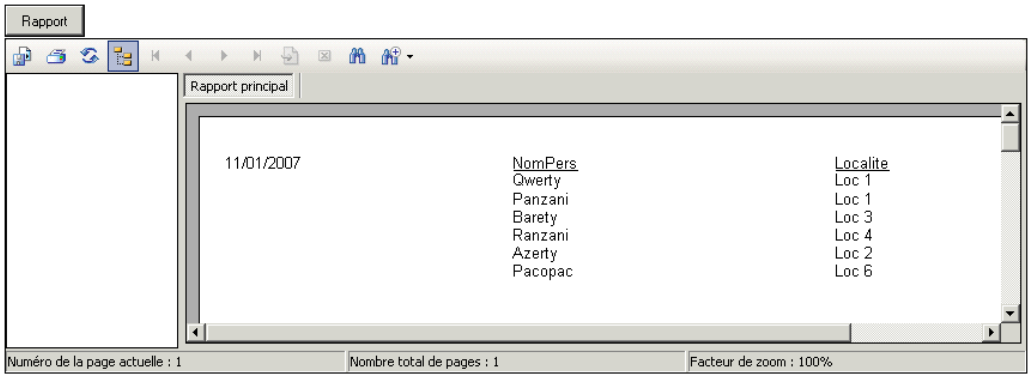
Le rapport est créé et il faut implémenter son exploitation dans l'application.

Le placement d'un composant visuel **`CrystalReportViewer`** sur un formulaire instancié à la demande offre l'avantage de fournir une prévisualisation à l'utilisateur et de dispenser le programmeur de la gestion de l'impression. Le travail du programmeur se résume alors à peu de choses comme l'illustre l'exemple suivant.

Un **CrystalReportViewer** nommé **MonCrystalVisu** a été déposé sur le formulaire principal de l'application. Un bouton *Rapport* nommé **BRapport** a également été placé sur ce formulaire et son rôle est d'activer le remplissage du rapport **CRPersEtLoc** et de le présenter dans **MonCrystalVisu**.
La seule programmation requise ici est celle de la procédure événementielle **BRapport_Click** suivante.

```
Private Sub BRapport_Click(ByVal sender As Object, ByVal e As System.EventArgs)
                                                          Handles BRapport.Click
  Dim UnRapport As New CRPersEtLoc              ' Instancier le rapport souhaité
  MonCrystalVisu.ReportSource = UnRapport       ' Lier le rapport au Viewer
End Sub
```

Et voici le résultat de son exécution.



Dans l'exemple précédent, l'objet **CrystalReport** lit toute la base de données et remplit le rapport conformément aux réglages réalisés à l'aide du concepteur.
Il est plus fréquemment utile de ne présenter qu'une partie des enregistrements, ceux correspondant à une sélection particulière. L'utilisateur de l'exemple précédent aurait pu souhaiter n'obtenir que la liste des personnes n'habitant pas la localité *Loc 1*.

Alors que la gestion d'une base de données par une application permet toutes les sélections souhaitables par l'utilisateur, les rapports intégrés dans cette application ne sont pas modifiables en cours d'exécution. L'édition de rapports partiels, à la demande de l'utilisateur, n'est donc pas possible si le programmeur ne met en place les dispositifs adéquats. Ces dispositifs sont une base de données *Rapports* et des procédures réalisant son remplissage et sa vidange au gré des besoins des utilisateurs. Les rapports sont ensuite conçus à partir de cette base.

Cette base de données *Rapports* doit contenir toutes les entités nécessaires pour satisfaire n'importe lequel des rapports intégrés et au moins, toutes les entités présentes dans la base de données réelle de l'exploitation. Elle peut aussi contenir l'une ou l'autre entité qui serait utile à certains rapports. Les entités qui reproduisent celles de la base de données réelle en sont des copies conformes, sauf qu'elles ne contiennent aucun enregistrement. Afin d'alléger le travail du S.G.B.D. sur cette base, les relations et contraintes ne sont pas reproduites. Par ailleurs, les relations nécessaires à un rapport sont spécifiées lors de sa conception. La base de données *Rapports* doit être stockée dans le dossier par défaut de l'application et la désignation de son nom au concepteur ne doit comporter aucun chemin.
Dans l'application, l'édition d'un rapport partiel doit réaliser le remplissage des entités de la base *Rapports* avant de désigner le rapport au **CrystalReportViewer**. Il convient que les entités soient vidées de leurs enregistrements après le traitement du rapport afin d'être disponibles pour un autre.