tellora

Administration

SQL Server

2012

Tables et index

Les tables

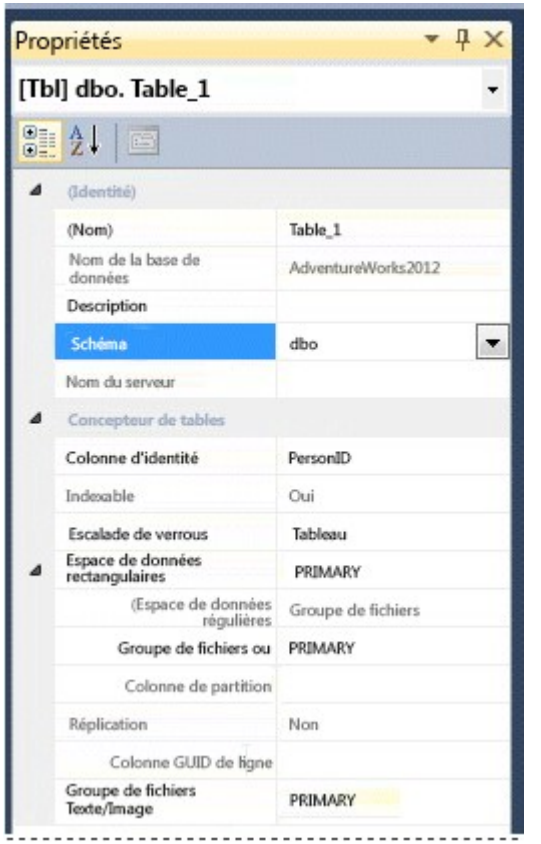
- La création d'une table dans SQL Server est délicate
  - le concepteur de tables de SQL Server Management Studio vous facilite la tâche
  - En plus des colonnes qui forment la table,
  - vous devez définir les contraintes d'intégrité
  - sans oublier de prendre en compte l'association des schémas,
  - le placement de groupes de fichiers et l'interclassement de caractères

Les tables

- Dans SQL Server Management Studio
  - connectez-vous à une instance SQL Server
  - développez le nœud **Databases**
  - développez la base de données à laquelle vous souhaitez ajouter la table
  - Cliquez avec le bouton droit sur le noeud **Tables**, puis sur **New Table**(Nouvelle table)
    - le concepteur de tables s'affiche

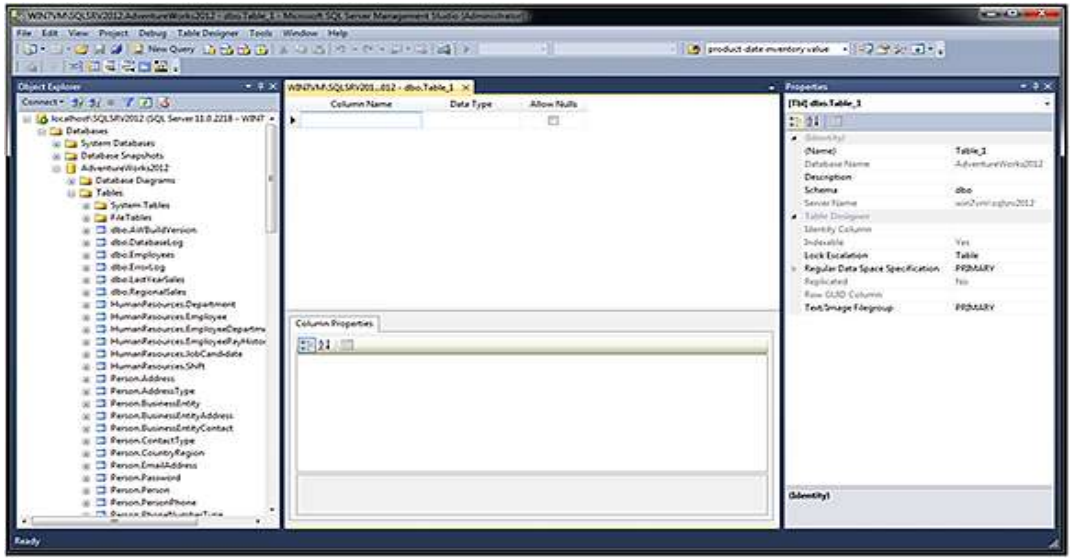
Les tables

- Par défaut, la table est contenue dans le schéma **dbo**.
  - Pour spécifier un schéma différent pour la table, cliquez avec le bouton droit dans le volet Concepteur de tables et sélectionnez **Propriétés**, comme indiqué dans l'illustration suivante.
    - Dans la liste déroulante **Schémas**, sélectionnez le schéma approprié
    - Dans la boîte de dialogue **Choisir un nom**, tapez un nom pour la table et cliquez sur **OK**



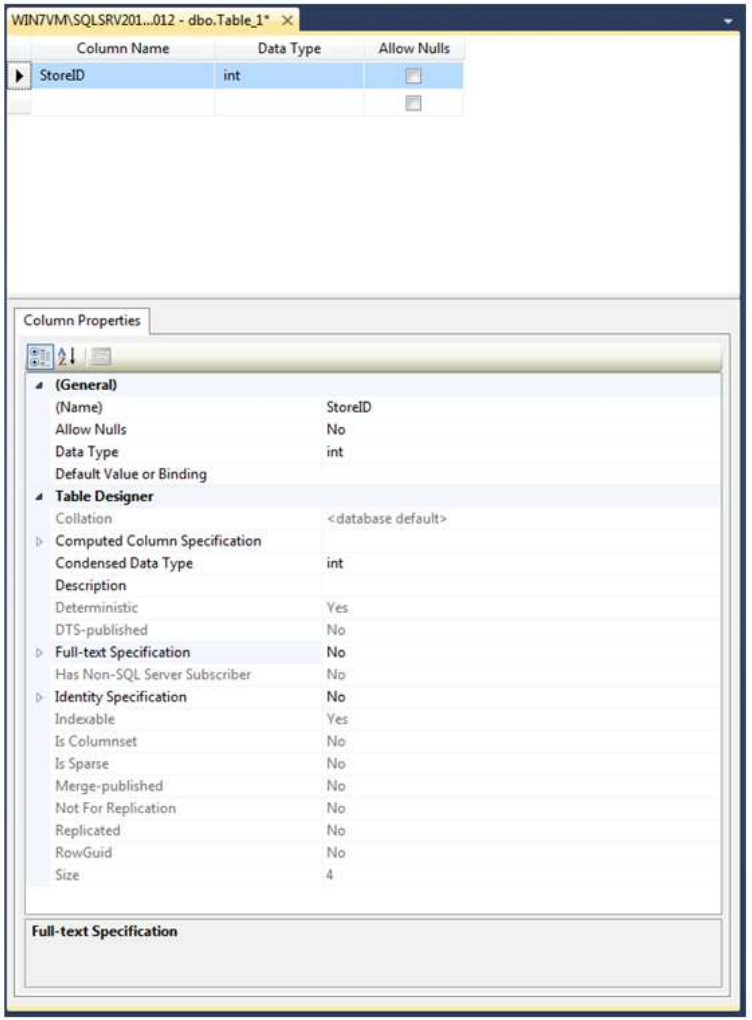
Les tables

- Le concepteur de tables de SQL Server Management Studio est le panneau central qui se subdivise en deux volets
  - Vous définissez vos colonnes de départ dans le volet du haut puis, dans le volet du bas,
  - vous configurez les propriétés supplémentaires pour chaque colonne
    - Les propriétés affichées dans le volet du bas sont associées aux colonnes sélectionnées dans le volet du haut
    - vous pouvez ouvrir la fenêtre **Properties** (Propriétés), à droite dans la figure ci-dessous



Les tables

- Ajouter des colonnes à une table
  - Saisissez le nom dans la première ligne de la première colonne de la grille dans le volet supérieur
  - sélectionnez **le type de données**
  - puis sélectionnez ou désélectionnez l'option **Allow Nulls** (Autoriser les valeurs Null)



Les tables

- Ajouter des colonnes à une table (suite)
  - Lorsque vous ajoutez une colonne dans la grille du haut, le volet du bas affiche les propriétés de cette colonne
    - Les propriétés **Name** (Nom), **AllowNulls** (Autoriser les valeurs Null) et **Data Type** (Type de données) apparaissent également dans le volet du bas
  - Vous pouvez modifier ces valeurs dans les deux volets
    - Vous pouvez également modifier les autres propriétés qui ne sont pas grisées
  - la propriété **Default Value or Binding** permet de définir une valeur par défaut

Les tables

- Les types de données
  - SQL Server vous permet de spécifier les types de données ci-dessous
    - Integer
      - `int -2,147,483,648 to 2,147,483,647`
    - `VARCHAR(100)`
      - `strings of characters`
    - `CHAR(50)`
      - `fixed width of 50 chars`
    - `DECIMAL(12,2)`
      - `precision number`
    - `Bigint`
      - `big integer`

Les tables

- Exemple

```
– CREATE TABLE dbo.MYTABLE
– (
–    my_integer   INTEGER PRIMARY KEY,
–    my_varchar        VARCHAR(100),
–    my_forcedwidth    CHAR(50),
–    my_yes_no         BIT,
–    my_decimal        DECIMAL(12,2),
–    my_bigint         Bigint
– );
```

Les tables

- Contraintes d'intégrité
  - **PRIMARY KEY** : précise que la ou les colonnes composent la clef de la table
    - ATTENTION : nécessite que chaque colonne participant à la clef soit NOT NULL.
  - **UNIQUE** : les valeurs de la ou les colonnes doivent être unique ou NULL, c'est à dire qu'à l'exception du marqueur NULL, il ne doit jamais y avoir plus d'une fois la même valeur (pas de doublon) au sein de l'ensemble de données formé par les valeurs des différentes colonnes composant la contrainte.
  - **CHECK** : permet de préciser un prédicat validant différentes colonnes de la table et qui accepterons les valeurs s'il est évalué à vrai.
  - **FOREIGN KEY** : permet, pour les valeurs de la ou les colonnes, de faire référence à des valeurs pré-existantes dans une ou plusieurs colonnes d'une autre table. Ce mécanisme s'appelle intégrité référentielle.
- Lorsqu'au cours d'un ordre SQL d'insertion, de modification ou de suppression, une contrainte n'est pas vérifiée on dit qu'il y a "violation" de la contrainte et les effets de l'ordre SQL sont totalement annulés (ROLLBACK)

Les tables

- Contraintes d'intégrité FK
  - **ON DELETE NO ACTION / ON UPDATE NO ACTION** : aucun traitement particulier n'est entrepris en cas de mise à jour ou suppression d'informations référencées.
    - Autrement dit, il y a blocage du traitement car le lien d'intégrité ne doit pas être brisé. Même effets que RESTRICT, mais post opératoire
  - **ON DELETE RESTRICT / ON UPDATE RESTRICT** : mêmes effets que NO ACTION, mais pré opératoire

Les tables

- Contraintes d'intégrité FK (suite)
  - **ON DELETE CASCADE / ON UPDATE CASCADE** : en cas de suppression d'un élément, les éléments qui le référence sont aux aussi supprimés
    - En cas de modification de la valeur de la clef, les valeurs des clefs étrangères qui le référence sont elles aussi modifiées afin de maintenir l'intégrité
    - Par exemple en cas de suppression d'un client les factures et commandes sont elles aussi supprimées
  - **NOTA** : ce mode est très tentant, mais son coût de traitement est très élevé et les performances peuvent très rapidement se dégrader fortement

Les tables

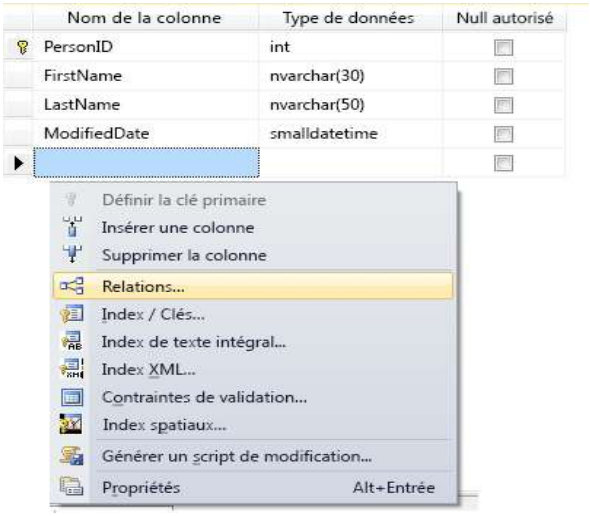
- Contraintes d'intégrité FK (suite)
  - **ON DELETE SET NULL / ON UPDATE SET NULL** : en cas de suppression d'un élément, les éléments qui le référence voit leur clef étrangère posséder le marqueur NULL
    - De même en cas de modification de la valeur de la clef. Le lien d'intégrité est alors brisé
  - L'intérêt d'une telle manoeuvre est de permettre la suppression des lignes devenues orphelines de manière différé,
    - par exemple dans un batch de nuit

Les tables

- Contraintes d'intégrité FK (suite)
  - **ON DELETE SET DEFAULT / ON UPDATE SET DEFAULT** : en cas de suppression comme en cas de mise à jour de la clef référencée, la référence passe à la valeur par défaut définie lors de la création de la table
    - Ce mode permet l'insertion d'un client générique, possédant un identifiant particulier (par exemple 0 ou -1) afin de ne jamais briser le lient d'intégrité référentielle
    - On veillera ensuite à rectifier la vrai valeur du lien au moment opportun si besoin

Les tables

- Définir une contraintes d'intégrité
  - Pour spécifier une colonne comme clé primaire, cliquez avec le bouton droit sur la colonne et sélectionnez **Définir la clé primaire**
  - Pour créer des relations de clé étrangère, des contraintes de validation ou des index, cliquez avec le bouton droit dans le volet Concepteur de tables et sélectionnez un objet dans la liste, comme indiqué ci-dessous



Les tables

- Exemples (suite)
  - **CREATE TABLE** T_PERSONNE7
  - (PRS_NOM **VARCHAR**(32),
  - PRS_PRENOM **VARCHAR**(32),
  - PRS_TELEPHONE **CHAR**(14) **UNIQUE**)
    - vous ne pouvez pas définir une contrainte d'unicité sur des colonnes de type BLOB

Les tables

- Pour créer une table dans l'éditeur de requête
  - Dans l'**Explorateur d'objets**, connectez-vous à une instance de Moteur de base de données.
  - Dans la barre d'outils standard, cliquez sur **Nouvelle requête**
  - Saisissez l'ordre SQL de création d'une table , puis cliquez sur **Exécuter**

```
CREATE TABLE T_CLIENT
(CLI_ID      INTEGER       NOT NULL PRIMARY KEY,
 CLI_NOM     CHAR(32)      NOT NULL,
 CLI_PRENOM VARCHAR(32));
```

Les tables

- Exemples
  - **CREATE TABLE** T_VOITURE
  - (VTR_ID **INTEGER** **NOT NULL PRIMARY KEY,**
  - VTR_MARQUE **CHAR**(32) **NOT NULL,**
  - VTR_MODELE **VARCHAR**(16),
  - VTR_IMMATRICULATION **CHAR**(10) **NOT NULL UNIQUE,**
  - VTR_COULEUR **CHAR**(16) **CHECK** (VALUE **IN** ('BLANC', 'NOIR', 'ROUGE', 'VERT', 'BLEU')))
    - Une table de voiture avec immatriculation unique et couleur limitée à 'BLANC', 'NOIR', 'ROUGE', 'VERT', 'BLEU‘
  - **CREATE TABLE** T_CLIENT
  - (CLI_NOM **CHAR**(32) **NOT NULL,**
  - CLI_PRENOM **VARCHAR**(32) **NOT NULL,**
  - **CONSTRAINT** PK_CLIENT **PRIMARY KEY** (CLI_NOM, CLI_PRENOM))
  - Une table de clients dont la clef est le couple de colonne NOM/PRENOM

Les tables

- Modification d'une table
  - **ALTER TABLE** nom_table
  - { **ADD** definition_colonne
  - | **ALTER** nom_colonne { **SET DEFAULT** valeur_défaut|**DROP DEFAULT**}
  - | **DROP** nom_colonne [CASCADE | RESTRICT]
  - | **ADD** définition_contrainte_ligne
  - | **DROP** **CONSTRAINT** nom_contrainte [CASCADE | RESTRICT] }

  - L'option CASCADE / RESTRICT permet de gérer l'intégrité de référence de la colonne ou la contrainte
  - Si RESTRICT, alors tout objet dépendant de cette colonne ou de cette contrainte provoquera l'annulation de l'opération de suppression
  - Si CASCADE, alors tous les objets dépendant de cette colonne ou contrainte seront supprimés

Les tables

- Suppression d'une table
  - **DROP** {**TABLE** | DOMAIN | ASSERTION | **VIEW** } nom_objet