



Dotnet France  
Technologies Sharepoint, SQL Server & .NET

Association Dotnet France

# Requêter un service de données ADO .NET

*Version 1.0*



James RAVAILLE

<http://blogs.dotnet-france.com/jamesr>

# Sommaire

---

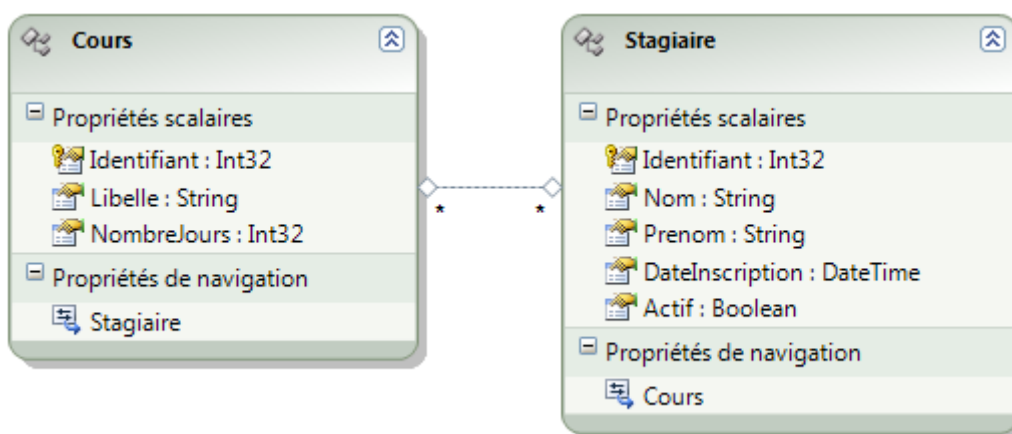
1	Introduction.....	3
1.1	Présentation .....	3
1.2	Désactivation des lecteurs de flux de données RSS dans les navigateurs.....	3
2	Ecriture d'URL.....	5
2.1	L'URL de base .....	5
2.2	Accéder à un ensemble d'entités .....	5
2.3	Accéder à une entité particulière.....	6
2.4	Les opérateurs de requête .....	6
2.4.1	Filtrer les entités.....	7
2.4.2	Trier les entités.....	8
2.4.3	Sélectionner les n premières ou n dernières entités .....	9
2.4.4	Etendre le chargement des données.....	9
2.1	Les fonctions.....	10
2.1.1	Les fonctions de traitement de chaînes de caractères.....	10
2.1.2	Les fonctions de traitement de dates.....	11
2.1.3	Les fonctions de traitement de données numériques .....	11
2.1.4	Les fonctions de type.....	12
3	Codes retours HTTP .....	13
4	Conclusion .....	14

## 1 Introduction

### 1.1 Présentation

Dans ce cours, nous allons étudier comment écrire des URLs, permettant de requêter un service de données ADO .NET, afin de consulter et gérer des données.

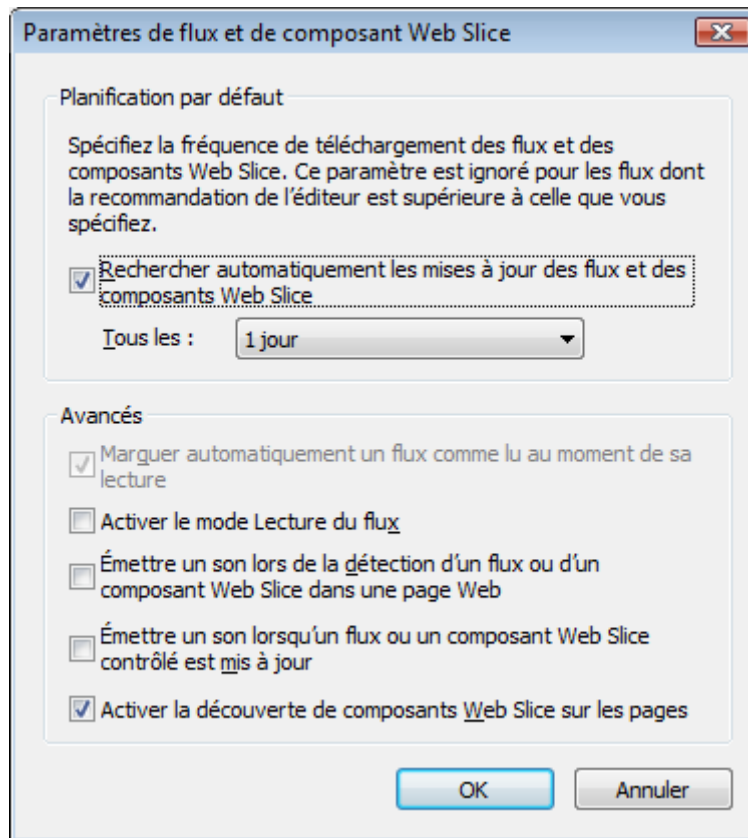
Pour comprendre le contenu de ce cours, nous vous conseillons vivement de lire le chapitre « Premiers pas avec ADO .NET Data Services » publié sur Dotnet-France, qui permet de réaliser un cas pratique, dans lequel nous construisons le service de données ADO .NET s'appuyant sur le Framework Entity, que nous allons requêter dans ce cours. Attention, pour les besoins liés à ce cours, le modèle de données à été modifié. Voici celui sur lequel nous nous appuyerons :



Dans ce cours, nous nous intéresseront uniquement aux requêtes de types HTTP Get, permettant de lire ces données. Son but de comprendre le langage de requêtage permettant d'accéder aux données, afin de les consulter.

### 1.2 Désactivation des lecteurs de flux de données RSS dans les navigateurs

Afin de pouvoir consulter directement les flux de données, renvoyés par le serveur à l'issue de l'exécution de requêtes HTTP, il est recommandé de désactiver les lecteurs de flux de données RSS dans les navigateurs. Dans Internet Explorer 8, dans les Options Internet, se positionner sur l'onglet Contenu, puis sur la section « flux et composants Web Slice », et cliquer sur le bouton « Paramètres ». La fenêtre suivante apparaît :



Vérifiez que la case à cocher « Activer le mode lecture du flux » n'est pas cochée. Si c'est le cas, décochez-la.

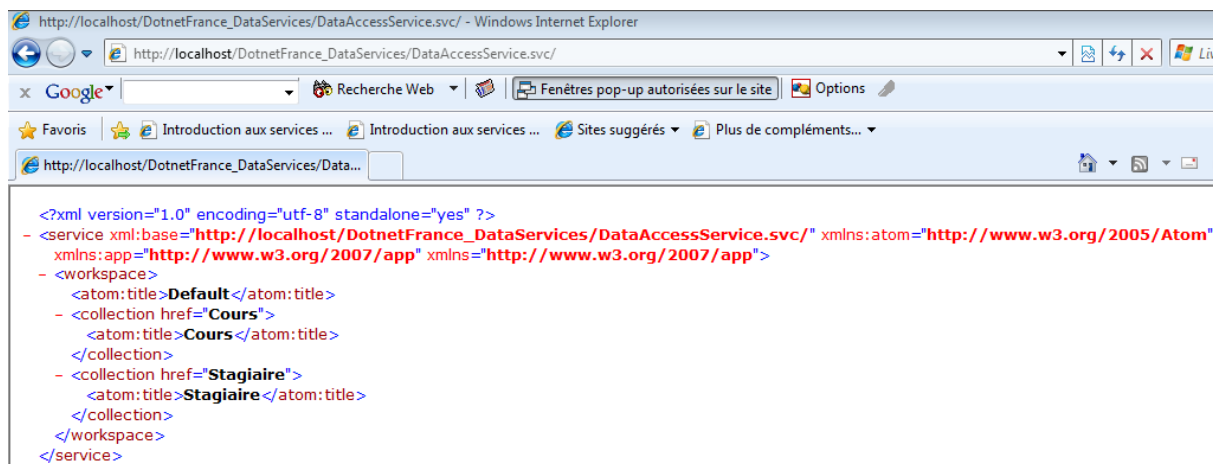
## 2 Ecriture d'URL

### 2.1 L'URL de base

L'URL de base de notre service de données ADO .NET est la suivante :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc
```

Cette URL permet d'accéder aux entités et services proposés par le service d'accès aux données. Lors de l'exécution de cette URL dans un navigateur Web, voici le résultat obtenu :



```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
- <service xmlns:base="http://localhost/DotnetFrance_DataServices/DataAccessService.svc/" xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:app="http://www.w3.org/2007/app" xmlns="http://www.w3.org/2007/app">
- <workspace>
  <atom:title>Default</atom:title>
  - <collection href="Cours">
    <atom:title>Cours</atom:title>
  </collection>
  - <collection href="Stagiaire">
    <atom:title>Stagiaire</atom:title>
  </collection>
</workspace>
</service>
```

Ce flux présente les noms des entités à utiliser dans l'URL, ainsi que les noms des classes d'entités, au travers desquelles les données contenues dans la base de données sont exposées.

### 2.2 Accéder à un ensemble d'entités

Pour accéder aux entités du modèle de données, exposées au travers du service de données ADO .NET, il suffit de préciser le nom de l'entité à la suite de l'URL de base. Voici une requête permettant d'obtenir la liste des stagiaires :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc/Stagiaire
```

Comme aucun filtre ni tri n'est précisé, une lecture linéaire des données contenues dans la base de données est réalisée. L'ordre des entités affichées est alors celui des entités dans leur ordre de lecture.

Voici le résultat d'exécution de cette requête :

```

<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
- <feed xml:base="http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/"
  xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
  xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
  xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Stagiaire</title>
  <id>http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/Stagiaire</id>
  <updated>2009-03-11T22:10:48Z</updated>
  <link rel="self" title="Stagiaire" href="Stagiaire" />
+ <entry>
+ <entry>
+ <entry>
- <entry>
  <id>http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/Stagiaire(4)</id>
  <title type="text" />
  <updated>2009-03-11T22:10:48Z</updated>
- <author>
  <name />
</author>
  <link rel="edit" title="Stagiaire" href="Stagiaire(4)" />
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Cours"
    type="application/atom+xml;type=feed" title="Cours" href="Stagiaire(4)/Cours" />
  <category term="DotnetFranceModel.Stagiaire"
    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme" />
- <content type="application/xml">
- <m:properties>
  <d:Identifiant m:type="Edm.Int32">4</d:Identifiant>
  <d:Nom>EMATO</d:Nom>
  <d:Prenom>Julie</d:Prenom>
  <d:DateInscription m:type="Edm.DateTime">2008-07-07T00:00:00</d:DateInscription>
  <d:Actif m:type="Edm.Boolean">>false</d:Actif>
</m:properties>
</content>
</entry>
</feed>

```

Ce flux de données constitue un flux de données au format ATOM. Dans le cas où le nom de l'entité précisé est inconnu, une erreur HTTP 404 est renvoyée.

### 2.3 Accéder à une entité particulière

Il est possible d'accéder à une entité particulière au travers d'un indexeur. Autrement dit, dans la liste des stagiaires présentés ci-dessus, un indexeur permet d'accéder à un stagiaire particulier via son indice dans la liste. Voici l'exemple d'une URL permettant d'accéder au second stagiaire de mon flux de données :

[http://localhost/DotnetFrance\\_DataServices/DataAccessService.svc/Stagiaire\(2\)](http://localhost/DotnetFrance_DataServices/DataAccessService.svc/Stagiaire(2))

Comme vous pouvez le remarquer, l'indice du premier stagiaire est 1.

### 2.4 Les opérateurs de requête

Les opérateurs de requêtes utilisables dans les URL sont les suivants :

Opérateur de requête	Signification
<i>filter</i>	Permet de filtrer les entités à partir d'une valeur ou d'une expression
<i>orderby</i>	Permet de trier les entités
<i>expand</i>	Permet de charger des entités liées à d'autres entités au travers d'une relation dans le modèle de données, sur lequel s'appuie le service de données

<i>top</i>	Permet d'obtenir les n premiers éléments d'une liste d'entités
<i>skip</i>	Permet d'obtenir les n derniers éléments d'une liste d'entités

Pour les utiliser, il est nécessaire de les préfixer par le caractère « \$ ». Il est aussi possible de les combiner. Voici quelques recommandations à connaître, lors de l'utilisation de leur utilisation :

- L'ordre des opérateurs de requête n'a pas d'importance.
- Les opérateurs de requête sont sensibles à la casse.
- Les noms des entités et leurs propriétés sont aussi sensibles à la casse.

### 2.4.1 Filtrer les entités

Les URLs d'accès aux entités acceptent les filtres sous la forme suivante :

```
<URL du service>/Ressource<Filtre>
```

Où le filtre peut s'exprimer de la manière suivante :

```
?$filter=<expression>&$filter=<expression>
```

Nous pouvons observer l'utilisation de paramètres de requête, tels qu'on les utilise dans les applications Web. L'opérateur de requête *filter* permet de filtrer les entités, au travers d'expressions booléennes constituées d'opérateurs :

- Logiques, présentés dans le tableau ci-dessous :

Opérateur logique	Signification
<i>eq</i>	Egalité
<i>ne</i>	Différent de
<i>gt</i>	Strictement plus grand que
<i>lt</i>	Strictement plus petit
<i>and</i>	Et logique
<i>or</i>	Ou logique
<i>not</i>	Négation

- Arithmétiques, aussi présentés dans le tableau ci-dessous :

Opérateur arithmétique	Signification
<i>add</i>	Addition
<i>sub</i>	Soustraction
<i>mul</i>	Multiplication
<i>div</i>	Division
<i>mod</i>	Modulo (reste entier de la division)

Voici un exemple d'URL, permettant de sélectionner un stagiaire par rapport à son nom. Vous remarquerez que les valeurs, telles que le nom du stagiaire, sont comprises entre deux quotes :

```
http://localhost/DotnetFrance_DataServices/DataAccessService.svc/Stagiaire?$filter=Nom eq 'EMATO'
```

Voici un autre exemple d'URL, permettant d'obtenir la liste des stagiaires, pour lesquels l'identifiant est strictement supérieur à 2 :

```
http://localhost/DotnetFrance_DataServices/DataAccessService.svc/Stagiaire?$filter=Identifiant gt 2
```

Voici un autre exemple d'URL, permettant d'obtenir la liste des stagiaires, pour lesquels l'identifiant est pair :

```
http://localhost/DotnetFrance_DataServices/DataAccessService.svc/Stagiaire?$filter=Identifiant mod 2 eq 0
```

- L'URL ci-dessous permet d'afficher la liste des stagiaires actifs :

```
http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/Stagiaire?$filter=Actif eq true
```

### 2.4.2 Trier les entités

L'opérateur de requête *orderby* permet de trier un jeu de d'entités sur une de leur propriété. Il s'utilise de la manière suivante :

```
$orderby=NomPropriété [asc/desc]
```

L'option *asc* (par défaut) permet d'effectuer un tri alphabétique ou numéraire croissant, et *desc* permet d'effectuer un tri alphabétique inversé ou numéraire décroissant.

Voici un premier exemple d'URL, permettant d'obtenir la liste des stagiaires, triés sur leur nom dans l'ordre alphabétique :

```
http://localhost/DotnetFrance_DataServices/DataAccessService.svc/Stagiaire?$orderby=Nom
```

Voici un second exemple d'URL, permettant d'obtenir la liste des stagiaires, triés sur leur prénom dans l'ordre alphabétique inversé :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc/Stagiaire?\$orderby=Prenom desc
```

Et voici un troisième exemple d'URL, permettant d'obtenir la liste des stagiaires, triés sur leur nom dans l'ordre alphabétique et sur leur prénom dans l'ordre alphabétique inversé :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc/Stagiaire?\$orderby=Nom asc, Prenom desc
```

### 2.4.3 Sélectionner les *n* premières ou *n* dernières entités

L'opérateur de requête *top* permet d'obtenir les *n* premières entités, résultant de l'exécution d'une requête. Il s'utilise de la manière suivante :

```
$Top=n
```

Voici un exemple d'URL, permettant d'obtenir les deux premiers stagiaires de la liste des stagiaires :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc/Stagiaire?\$top=2
```

A l'inverse, l'opérateur de requête *skip* permet d'obtenir les *m* dernières entités, en esquivant les *n* premières entités. Il s'utilise de la manière suivante :

```
$Skip=n
```

Voici un exemple d'URL, permettant d'obtenir une liste des stagiaires, à l'exception des trois premiers stagiaires :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc/Stagiaire?\$skip=3
```

### 2.4.4 Étendre le chargement des données

L'opérateur de requête *expand* permet d'étendre le chargement des données, en permettant d'inclure des entités en relation avec les entités visées par l'URL.

Par exemple, l'URL suivante permet d'obtenir la liste des stagiaires, avec pour chacun d'entre eux, la liste des cours auxquels ils sont inscrits :

```
http://localhost/DotnetFrance\_DataServices/DataAccessService.svc/Stagiaire?\$expand=Cours
```

Aussi, voici une URL permettant d'accéder au premier stagiaire, avec la liste des cours auxquels il est inscrit :

[http://localhost/DotnetFrance\\_DataServices/DataAccessService.svc/Stagiaire\(1\)?\\$expand=Cours](http://localhost/DotnetFrance_DataServices/DataAccessService.svc/Stagiaire(1)?$expand=Cours)

## 2.1 Les fonctions

En plus des opérateurs présentés ci-dessus, il est possible d'utiliser les fonctions, qui se répartissent dans quatre catégories distinctes :

- Les fonctions de traitement de chaînes de caractères.
- Les fonctions de traitement de dates.
- Les fonctions de traitement de données numériques.
- Les fonctions de type.

### 2.1.1 Les fonctions de traitement de chaînes de caractères

Voici les fonctions de traitement de chaînes de caractères, qu'il est possible d'utiliser dans les URL (s0 et s1 sont des chaînes de caractères) :

Fonction	Signification
bool substringof(s0, s1)	Renvoie <i>Vrai</i> si <i>s1</i> contient au moins une occurrence de <i>s0</i>
bool endswith(s0, s1)	Renvoie <i>Vrai</i> si <i>s0</i> se termine par <i>s1</i>
bool startswith(s0, s1)	Renvoie <i>Vrai</i> si <i>s0</i> commence par <i>s1</i>
int length(s0)	Retourne le nombre de caractères de <i>s0</i>
string insert(s0, int pos, s1)	Permet d'insérer la chaîne de caractères <i>s1</i> dans <i>s0</i> à la position précisée.
string remove(p0, int pos)	Permet de supprimer une partie d'une chaîne de caractères, à partir d'une position.
string remove(s0, int pos, int longueur)	Permet de supprimer une partie d'une chaîne de caractères, à partir d'une position et d'une longueur.
string replace(s0, chaîne rechercher, chaîne remplacer)	Permet de remplacer une partie d'une chaîne de caractères par une autre
string substring(s0, int pos)	Permet d'obtenir un sous-ensemble d'une chaîne de caractères. Par exemple, <i>substring('TRAIN', 2)</i> retourne <i>'AIN'</i>
string substring(s0, int pos, int NbCaracteres)	Permet d'obtenir un sous-ensemble d'une chaîne de caractères. Par exemple, <i>substring('TRAIN', 0, 2)</i> retourne <i>'TR'</i>
string tolower(s0)	Permet d'obtenir <i>s0</i> en caractères minuscules
string toupper(s0)	Permet d'obtenir <i>s0</i> en caractères majuscules
string trim(s0)	Permet d'obtenir une nouvelle chaîne de caractères, ayant le même contenu que <i>s1</i> sans les espaces avant la première lettre et après la dernière lettre
string concat(s0, s1)	Permet d'obtenir une nouvelle chaîne de caractères, résultant de la concaténation de <i>s0</i> et <i>s1</i>

Voici un exemple d'URL, permettant d'obtenir les informations sur le stagiaire 'RAVILLE James' :

[http://localhost/DotnetFrance\\_DataServices\\_Requetes/DataAccessService.svc/Stagiaire?\\$filter='RAVILLE James' eq concat\(concat\(Nom, ' '\), Prenom\)](http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/Stagiaire?$filter='RAVILLE James' eq concat(concat(Nom, ' '), Prenom))

### 2.1.2 Les fonctions de traitement de dates

Voici les fonctions de traitement de dates, qu'il est possible d'utiliser dans les URL (*d0* est de type DateTime) :

Fonction	Signification
int year(d0)	Permet d'obtenir l'année d'une date
int month(d0)	Permet d'obtenir le mois d'une date
int day(d0)	Permet d'obtenir l'indice du jour de la semaine d'une date
int hour(d0)	Permet d'obtenir l'heure d'une date / heure
int minute(d0)	Permet d'obtenir les minutes d'une date / heure
int second(d0)	Permet d'obtenir les secondes d'une date / heure

Voici quelques exemples d'URL :

- L'URL ci-dessous permet d'afficher la liste des stagiaires, qui se sont inscrits en 2007 :

[http://localhost/DotnetFrance\\_DataServices\\_Requetes/DataAccessService.svc/Stagiaire?\\$filter=year\(DateInscription\) eq 2007](http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/Stagiaire?$filter=year(DateInscription) eq 2007)

- L'URL ci-dessous permet d'afficher la liste des stagiaires, qui se sont inscrits avant 2008 et qui sont actifs :

[http://localhost/DotnetFrance\\_DataServices\\_Requetes/DataAccessService.svc/Stagiaire?\\$filter=year\(DateInscription\) lt 2008 and Actif eq true](http://localhost/DotnetFrance_DataServices_Requetes/DataAccessService.svc/Stagiaire?$filter=year(DateInscription) lt 2008 and Actif eq true)

### 2.1.3 Les fonctions de traitement de données numériques

Voici les fonctions de traitement de données numériques, qu'il est possible d'utiliser dans les URL :

Fonction	Signification
decimal round(decimal p0)	Permet d'arrondir un nombre décimal, à la valeur entière la plus proche. Par exemple <i>round(12.22) vaut 12</i> et <i>round(12.52) vaut 13</i> .
decimal floor(decimal p0)	Permet de tronquer un nombre décimal, de manière à ne conserver que la partie entière. Par exemple, <i>floor(12.32) vaut 12</i> .
decimal ceiling(decimal p0)	Permet d'obtenir le plus petit nombre entier supérieur ou égal au nombre décimal. Par exemple <i>ceiling(12.32) vaut 13</i> .

Remarque : Les fonctions d'arrondi ne sont pas prises en charge, lors de la création de services de données ADO .NET, qui utilisent LINQ to SQL.

#### 2.1.4 Les fonctions de type

Voici les fonctions de type, qu'il est possible d'utiliser dans les URL :

Fonction	Signification
bool isof(type p0)	Permet si un élément est du type passé en paramètre
bool isof(expression p0, type p1)	Renvoie <i>Vrai</i> si <i>p0</i> est de type <i>p1</i>
<p1> cast(expression p0, type p1)	Permet de convertir <i>p0</i> suivant le type <i>p1</i>

### 3 Codes retours HTTP

L'exécution de requêtes HTTP, visant à exécuter un service de données ADO .NET peut rencontrer une erreur. Le tableau ci-dessous présente quelques codes retours HTTP désignant des erreurs, principalement rencontrées :

Code	Libelle	Signification
400	Demande incorrecte	La syntaxe de l'URL n'est pas conforme
404	Service introuvable	L'URL pointe vers un service de données ADO .NET qui n'a pas été trouvé
406	Requête non acceptable	Erreur dans le format de la requête. Le serveur fournit en retour le langage et les types d'encodages disponibles
422	Impossible de traiter l'entité	L'exécution d'une requête HTTP visant à modifier l'état une entité entraîne une violation du schéma ADO.NET Data Services
405	Verbe HTTP non pris en charge	La requête utilise un verbe HTTP qui n'est pas prise en charge par la ressource identifiée par l'URL
500	Erreur d'exécution	Une erreur s'est produite lors de l'exécution du service

## 4 Conclusion

Ce cours vous a permis d'étudier comment écrire des URLs, vous permettant de consommer un service de données ADO .NET.

Aussi, dans le cours sur la consommation de services de données ADO .NET publié sur Dotnet-France, nous verrons qu'il est possible de consommer sans avoir à écrire d'URL. Le Framework .NET vous permet de générer des classes, dont les objets qu'elles permettent de créer, se chargent elles-mêmes de créer les URLs, et de les exécuter.