Les « collations » dans SQL Server

Microsoft Communautés

Souvent ignorées, les collations sont un outil unique et indispensable aux manipulations des chaînes de caractères. Leur usage se répercute sur le traitement des données et les performances. SQL Server 2000 propose le choix de plus de 700 collations.

Frédéric Brouard
Spécialiste SQL et SGBDR,
a écrit le livre «SQL, développement»
et le site sqlpro.developpez.com.
Il est MVP (Most Valuable
Profesional) sur SQL.

Une collation consiste en une énumération de données et de leurs équivalences afin d'en déduire le comportement dans le traitement des chaînes de caractères (pour opérations de comparaison et de tri notamment).

Les chaînes de caractères sont des compositions de lettres pour lesquelles il existe un ordre, dit alphabétique. Dans la plupart des langues latines, les lettres «basiques» de «a» à «z» peuvent être modifiées par l'ajout d'un symbole présent sur la lettre même : accent, cédille, tilde... On parle alors de caractère diacritique.

Il faut donc intercaler à l'ordre des 26 lettres basiques, celui des lettres diacritiques. Si tel n'était pas le cas, lors d'un tri de chaîne de caractères, les lettres accentuées apparaîtraient après le z, du fait que leur code informatique est supérieur à ce dernier.

La collation consiste donc à ajouter au code ASCII ou UNICODE, un «poids», sorte de code d'équivalence. Par exemple la lettre «e», sera dotée du poids 5 tout comme é, è, ê, ë... Les collations permettent aussi de confondre majuscules et minuscules, par ce même mécanisme. Enfin, dans le cas des ligatures («œ» de cœur, æ de Laetitia...), la collation permet d'introduire une équivalence avec des compositions de lettres basiques.

Choisir sa collation...

SQL Server propose de spécifier sa collation à l'installation du serveur, à la création d'une table ou encore à la définition d'une colonne. Attention, le choix à l'installation est définitif et pesant !

Les caractéristiques des collations de SQL Server 2000 sont :

- L'ordre induit par la langue ou le code de caractères
- L'influence de la casse (CS => Case Sensitive, CI => Case Insensitive)
- L'influence des caractères diacritique (AS => Accent Sensitive, AI => Accent Insensitive)
- La sensibilité à la largeur de stockage : ASCII et EBCDIC sont codés sur 2 octets, UNICODE sur 4 (WS => Wide Sensitive)
- Le respect des caractères japonais Kana : distinction phonétique des syllabes du katakana et du hiragana (KS => Kana Sensitive, KI => Kana Insensitive).

Enfin, les collations binaires sont suffixées BIN. Par exemple la collation : French_CI_AI_KS_WS se base sur l'ordre du dictionnaire français, n'est ni sensible à la casse ni aux caractères diacritiques, mais est sensible aux caractères Kana et à la largeur du code de représentation des caractères.

Conséquences

Le choix d'une collation impacte les insertions, mises à jour, les comparaisons et les tris.

Le piège serait de considérer qu'il vaut mieux utiliser une collation insensible aux accents et à la casse (m majuscule/minuscule). Mais cette manière de procéder possède deux inconvénients majeurs : des données comme des mots de passe deviennent insensibles aux accents et majuscules, et sont plus faciles à trouver, de plus la performance d'une collation lâche est largement moindre que celle d'une collation binaire.

Par défaut SQL Server s'installe avec une collation lâche. Si le serveur destinataire du développement a été installé avec une collation forte alors il se peut que certaines requêtes, comme certains codes ne fonctionnent plus, car tous les noms d'objet (tables, colonnes) deviennent eux aussi sensibles à la casse.

On spécifiera au niveau du serveur, la collation la plus adaptée pour le développement (si possible sensible à tout, voire binaire afin de booster les performances). Par exception on précisera quelles colonnes auront une collation différente en fonction de besoins de traitements particuliers.

NOTA : La requête suivante donne la liste dés collations de SQL Server :

```
SELECT * FROM ::fn_helpcollations()
```

■

Trucs et astuces

> Pour s'affranchir d'une collation sensible à la largeur, utiliser l'opérateur CAST :

```
SELECT *
FROM maTable
WHERE CAST(ColUnicode AS
VARCHAR(32)) = ColASCII
```

> Pour comparer deux colonnes dotées d'une collation différente, il faut utiliser "COLLATE" :

```
SELECT *
FROM MaTable
WHERE GreekCol = LatinCol
COLLATE greek_ci_as
```

> Pour distinguer "maçoné" de "Macone" en cas de collation faible, il faut transtyper en binaire :

```
WHERE CAST('maçoné' AS
VARBINARY(16)) =
    CAST('macone' AS
VARBINARY(16))
```