Quelques UDF (fonctions utilisateur) pour MS SQL Server.

Une collection d'UDF de toute nature pour satisfaire certains besoin. Le tout en vrac !.

Par Frédéric Brouard - MVP SQL Server
Expert SQL et SGBDR, Auteur de :

- SQL, Développement, Campus Press 2001
- SQL, collection Synthex, Pearson Education 2005, co écrit avec Christian Soutou
- http://sqlpro.developpez.com (site de ressources sur le langage SQL et les SGBDR)
- Enseigant aux Arts & Métiers et à l'ISEN Toulon


```
/*****************************************************************************/
-- mise en majuscule de la première lettre et minuscule de toutes les autres
/*****************************************************************************/
CREATE FUNCTION F_FIRST_CAP (@NAME VARCHAR(8000))
RETURNS VARCHAR(8000)
AS
BEGIN
   IF @NAME IS NULL
      RETURN NULL
   IF LEN(@NAME) = 1
      RETURN UPPER(@NAME)
   RETURN UPPER(SUBSTRING(@NAME, 1, 1)) +
          LOWER(SUBSTRING(@NAME, 2, LEN(@NAME) -1))
END
GO
```

```
/*****************************************************************************/
-- table des jours de semaine
/*****************************************************************************/
CREATE FUNCTION F_JOUR_SEMAINE ()
RETURNS TABLE
AS
RETURN (SELECT 1 AS N, 'Lundi' AS JOUR
        UNION
        SELECT 2 AS N, 'Mardi' AS JOUR
        UNION
        SELECT 3 AS N, 'Mercredi' AS JOUR
        UNION
        SELECT 4 AS N, 'Jeudi' AS JOUR
        UNION
        SELECT 5 AS N, 'Vendredi' AS JOUR
        UNION
        SELECT 6 AS N, 'Samedi' AS JOUR
        UNION
        SELECT 7 AS N, 'Dimanche' AS JOUR)
GO
```

```
/*****************************************************************************/
-- calcul de la date de Pâque
```

```
/*****************************************************************************/
CREATE FUNCTION F_PAQUE (@AN INT)
       RETURNS DATETIME
AS

BEGIN

IF @AN IS NULL
   RETURN NULL

DECLARE @G INT
DECLARE @I INT
DECLARE @J INT
DECLARE @C INT
DECLARE @H INT
DECLARE @L INT
DECLARE @JourPaque INT
DECLARE @MoisPaque INT
DECLARE @DimPaque DATETIME

SET @G = @AN % 19
SET @C = @AN / 100
SET @H = (@C - @C / 4 - (8 * @C + 13) / 25 + 19 * @G + 15) % 30
SET @I = @H - (@H / 28) * (1 - (@H / 28) * (29 / (@H + 1)) * ((21 - @G) / 11))
SET @J = (@AN + @AN / 4 + @I + 2 - @C + @C / 4) % 7

SET @L = @I - @J
SET @MoisPaque = 3 + (@L + 40) / 44
SET @JourPaque = @L + 28 - 31 * (@MoisPaque / 4)


SET @DimPaque = CAST(CAST(@AN AS VARCHAR(4)) +
                CASE
                   WHEN @MoisPaque < 10 THEN '0' + CAST(@MoisPaque AS CHAR(1))
                                        ELSE       CAST(@MoisPaque AS CHAR(2))
                END +
                CASE
                   WHEN @JourPaque < 10 THEN '0' + CAST(@JourPaque AS CHAR(1))
                                        ELSE       CAST(@JourPaque AS CHAR(2))
                END
                AS DATETIME)

RETURN @DimPaque

END
GO
```

```
/*****************************************************************************/
-- Transformation d'un entier en chaine binaire VARCHAR
/*****************************************************************************/
CREATE FUNCTION F_INT_TO_BIT (@I INT)
       RETURNS VARCHAR(512)
AS
BEGIN
-- effet de bord
   IF @I IS NULL
      RETURN NULL
-- valeur basique
   IF @I = 0
      RETURN '0'
-- signe négatif
   DECLARE @SIGNE VARCHAR(1)
   SET @SIGNE = ''
   IF @I < 0
      SET @SIGNE = '-'
-- conversion
   DECLARE @BIT_OUT VARCHAR(512)
   SET @BIT_OUT = ''
   WHILE NOT @I = 0
   BEGIN
      IF @I % 2 = 0
        SET @BIT_OUT = '0' +  @BIT_OUT
      ELSE
        SET @BIT_OUT = '1' +  @BIT_OUT
      SET @I = @I / 2
   END
   RETURN @SIGNE + @BIT_OUT
END
GO
```

```
/*****************************************************************************/
-- retourne le nom de l'utilisateur d'un id (userid) donné
/*****************************************************************************/
CREATE FUNCTION F_FORMATUSER(@USERID SMALLINT) RETURNS VARCHAR(32) AS
BEGIN
   RETURN(CAST(SUBSTRING(USER_NAME(@USERID), 1, 32) AS VARCHAR(32)))
```

```
END
GO
```

```
/*****************************************************************************/
-- retourne une chaine interprétable en chaine SQL
/*****************************************************************************/
CREATE FUNCTION F_QUOTESTR(@S VARCHAR(8000))
RETURNS VARCHAR(8000)
AS
BEGIN
   DECLARE @OUT VARCHAR(8000)
   SET @OUT = CASE
                WHEN @S IS NULL
                     THEN 'NULL'
                ELSE ''''+REPLACE(@S, '''', '''''')+''''
              END
   RETURN (@OUT)
END
GO
```

```
/*****************************************************************************/
-- retourne une chaine de caractère représentant une date au format ISO
/*****************************************************************************/
CREATE FUNCTION F_DATEISO(@D DATETIME)
RETURNS VARCHAR(8000)
AS
BEGIN
   DECLARE @OUT VARCHAR(12)
   SET @OUT = CASE
                WHEN @D IS NULL
                     THEN 'NULL'
                ELSE ''''+CONVERT(CHAR(10), @D, 121)+''''
              END
   RETURN (@OUT)
END
GO
```

```
/*****************************************************************************/
-- obtient la liste des colonnes d'une table
/*****************************************************************************/
CREATE FUNCTION F_LISTCOLS (@NOM_TABLE VARCHAR(128))
RETURNS VARCHAR(8000) AS
BEGIN
   DECLARE @RETVAL VARCHAR(8000)
   SET @RETVAL = ''
   SELECT @RETVAL = @RETVAL +COLUMN_NAME+', '
   FROM   INFORMATION_SCHEMA.COLUMNS
   WHERE  TABLE_NAME = @NOM_TABLE
   IF @RETVAL IS NULL
      RETURN NULL
   IF  @RETVAL = ''
      RETURN NULL
   SET @RETVAL = SUBSTRING(@RETVAL, 1, LEN(@RETVAL) -1)
   RETURN @RETVAL
END
GO
```

```
/*****************************************************************************/
-- purge de caractères indésirables
/*****************************************************************************/
-- exemple : F_RESTRICT('à Paris...?', 'abcdefghijklmnopqrstuvwxyz') => 'aris'
CREATE FUNCTION F_RESTRICT (@IN VARCHAR (8000),
                            @CHARSOK VARCHAR(256))
RETURNS VARCHAR (8000)
AS
BEGIN
-- effets de bord
   IF @IN IS NULL
      RETURN NULL
   IF @CHARSOK IS NULL
      RETURN NULL
   IF LEN(@IN) = 0
      RETURN @IN
-- initialisation
   DECLARE @I INTEGER
   DECLARE @OUT VARCHAR(8000)
   SET @OUT = ''
-- lecture caractère par caractère
   SET @I =1
   WHILE @I <= LEN(@IN)
   BEGIN
      IF PATINDEX('%' + SUBSTRING(@IN, @I, 1)+ '%', @CHARSOK) > 0
         SET @OUT = @OUT + SUBSTRING(@IN, @I, 1)
      SET @I = @I + 1
   END
```

```
   RETURN @OUT
END
GO
```

```
/*************************************************************************/
-- conversion d'heure minute seconde littérale en heure décimale
/*************************************************************************/
CREATE FUNCTION F_CONVERT_HMS_HD (@HMS CHAR(8))
RETURNS FLOAT
AS

BEGIN

DECLARE @H FLOAT
DECLARE @M FLOAT
DECLARE @S FLOAT
DECLARE @RETVAL FLOAT

-- cas trivial
IF @HMS IS NULL
   RETURN NULL

-- voir si saisie erronée (pas de chiffres)
SET @HMS = REPLACE(@HMS, ':', '')
IF LEN(@HMS) <> 6
   RETURN NULL

DECLARE @I INTEGER
SET @I = 1
WHILE @I < 7
BEGIN
   IF SUBSTRING(@HMS, @I, 1) NOT BETWEEN '0' AND '9'
      RETURN NULL
   SET @I = @I + 1
END

-- la saisie est correcte
SET @H = CAST(SUBSTRING(@HMS, 1, 2) AS FLOAT)
SET @M = CAST(SUBSTRING(@HMS, 3, 2) AS FLOAT) / 60.0
SET @S = CAST(SUBSTRING(@HMS, 5, 2) AS FLOAT) / 3600.0
SET @RETVAL = @H + @M + @S

RETURN @RETVAL

END
GO
```

```
/*************************************************************************/
-- Compte le nombre d'occurences d'une sous chaine dans une chaine
/*************************************************************************/
CREATE FUNCTION dbo.F_COUNTSTR (@STR VARCHAR(8000), @PATTERN VARCHAR(8000))
RETURNS INTEGER
AS
BEGIN

DECLARE @I INTEGER

-- cas trivial données en entrée NULL
IF @STR IS NULL OR @PATTERN IS NULL
BEGIN
   SET @I = NULL
   RETURN @I
END

-- cas trivial données en entrée vide
IF @STR ='' OR @PATTERN =''
BEGIN
   SET @I = 0
   RETURN @I
END

-- cas général

DECLARE @STR2 VARCHAR(8000)

SET @STR2 = @STR
SET @I = 0

WHILE PATINDEX('%'+@PATTERN+'%', @STR2) > 0
BEGIN
   SET @I = @I +1
   IF LEN(@STR2) > PATINDEX('%'+@PATTERN+'%', @STR2) + LEN(@PATTERN)
      SET @STR2 = SUBSTRING(@STR2, PATINDEX('%'+@PATTERN+'%', @STR2)
                            + LEN(@PATTERN), LEN(@STR2) - PATINDEX('%'+@PATTERN+'%', @STR2)
                            -  LEN(@PATTERN)+1)
   ELSE
```

```
      SET @STR2 = ''
END

RETURN @I

END
GO
```

```
/*****************************************************************************/
-- remplace un datetime par une datetime avec heure à zero
/*****************************************************************************/
CREATE FUNCTION F_DATETIME_AS_DATE (@DT DATETIME)
RETURNS DATETIME AS
BEGIN
   RETURN CAST(FLOOR(CAST(@DT AS FLOAT)) AS DATETIME)
END
GO
```

```
/*****************************************************************************/
-- soundex 2 (Celko SQL Avancé p 86)
/*****************************************************************************/
CREATE FUNCTION F_SOUNDEX2 (@NAME VARCHAR (128))
RETURNS CHAR (4)
AS
BEGIN
   DECLARE @SNDX2 CHAR(4)
   SET @SNDX2 = '    '
   IF @NAME IS NULL
      RETURN @SNDX2
   DECLARE @FIRSTLET CHAR(1)
   DECLARE @XGRAM VARCHAR(4)
-- mise en majuscule
   SET @NAME = UPPER(@NAME)
-- dé diacritisation
   SET @NAME = dbo.F_TRANSLATE(@NAME, 'ÀÂÄÉÈÊËÎÏÔÖÙÛÜÇ', 'AAAEEEEIIOOUUUC')
-- restriction aux letttres
   SET @NAME = dbo.F_RESTRICT(@NAME, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ')
   IF @NAME = ''
      RETURN @SNDX2
-- substitution des préfixes
   IF LEN(@NAME) >= 3
   BEGIN
      SET @XGRAM = SUBSTRING(@NAME, 1, 3)
      IF @XGRAM = 'MAC'
         IF LEN(@NAME) = 3
            SET @NAME = 'MCC'
         ELSE
            SET @NAME = 'MCC'+ SUBSTRING(@NAME, 4, LEN(@NAME) -3)
      IF @XGRAM = 'SCH'
         IF LEN(@NAME) = 3
            SET @NAME = 'SSS'
         ELSE
            SET @NAME = 'SSS'+ SUBSTRING(@NAME, 4, LEN(@NAME) -3)
   END
   IF LEN(@NAME) >= 2
   BEGIN
      SET @XGRAM = SUBSTRING(@NAME, 1, 2)
      IF @XGRAM = 'KN'
         IF LEN(@NAME) = 2
            SET @NAME = 'NN'
         ELSE
            SET @NAME = 'NN'+ SUBSTRING(@NAME, 3, LEN(@NAME) -2)
      IF @XGRAM = 'PF'
         IF LEN(@NAME) = 2
            SET @NAME = 'FF'
         ELSE
            SET @NAME = 'FF'+ SUBSTRING(@NAME, 3, LEN(@NAME) -2)
      IF @XGRAM = 'PH'
         IF LEN(@NAME) = 2
            SET @NAME = 'FF'
         ELSE
            SET @NAME = 'FF'+ SUBSTRING(@NAME, 3, LEN(@NAME) -2)
   END
   IF LEN(@NAME) >= 1
   BEGIN
      SET @XGRAM = SUBSTRING(@NAME, 1, 1)
      IF @XGRAM = 'K'
         IF LEN(@NAME) = 1
            SET @NAME = 'C'
         ELSE
            SET @NAME = 'C'+ SUBSTRING(@NAME, 2, LEN(@NAME) -1)
   END

-- transformations sauf première lettre                                      ''
   SET @FIRSTLET = SUBSTRING(@NAME, 1, 1)
   IF LEN(@NAME) > 1
```

```
      SET @NAME = dbo.F_TRANSLATE(SUBSTRING(@NAME, 2, LEN(@NAME)-1), 'AEIOUY', 'AAAAAA')
   ELSE
      SET @NAME = ''
   SET @NAME = REPLACE(@NAME, 'DG',   'GG')
   SET @NAME = REPLACE(@NAME, 'CAAN', 'TAAN')
   SET @NAME = REPLACE(@NAME, 'D',    'T')
   SET @NAME = REPLACE(@NAME, 'NST',  'NSS')
   SET @NAME = REPLACE(@NAME, 'AV',   'AF')
   SET @NAME = REPLACE(@NAME, 'Q',    'G')
   SET @NAME = REPLACE(@NAME, 'Z',    'S')
   SET @NAME = REPLACE(@NAME, 'M',    'N')
   SET @NAME = REPLACE(@NAME, 'KN',   'NN')
   SET @NAME = REPLACE(@NAME, 'K',    'C')
-- remplacement des H par A sauf suivi et précédé par A (exemple '...AHA...')
   SET @NAME = REPLACE(@NAME, 'AHA',  'AhA')
   SET @NAME = REPLACE(@NAME, 'H',    'A')
   SET @NAME = REPLACE(@NAME, 'AhA',  'AHA')
-- remplacements divers
   SET @NAME = REPLACE(@NAME, 'AW',   'A')
   SET @NAME = REPLACE(@NAME, 'PH',   'FF')
   SET @NAME = REPLACE(@NAME, 'SCH',  'SSS')
-- suppression A en fin de mot
   WHILE SUBSTRING(@NAME, LEN(@NAME), 1) = 'A'
      IF LEN(@NAME) > 1
         SET @NAME = SUBSTRING(@NAME, 1, LEN(@NAME)-1)
      ELSE
         SET @NAME = ''
-- suppression S en fin
   WHILE SUBSTRING(@NAME, LEN(@NAME), 1) = 'S'
      IF LEN(@NAME) > 1
         SET @NAME = SUBSTRING(@NAME, 1, LEN(@NAME)-1)
      ELSE
         SET @NAME = ''
-- supression NT en fin
   IF LEN(@NAME) >= 2
     IF SUBSTRING(@NAME, LEN(@NAME)-1, 2) = 'NT'
        IF LEN(@NAME) > 2
           SET @NAME = SUBSTRING(@NAME, 1, LEN(@NAME) -2)
-- suppression des A
   SET @NAME = REPLACE(@NAME, 'A', '')
-- test effet de bord : @NAME vide
   IF @NAME = ''
   BEGIN
      SET @SNDX2 = @FIRSTLET +'   '
      RETURN @SNDX2
   END
-- suppression des répétitions
   DECLARE @OUT VARCHAR(4)
   SET @OUT = @FIRSTLET
   DECLARE @I INTEGER
   DECLARE @C CHAR(1)
   DECLARE @CC CHAR(1)
   SET @I = 1
   SET @CC = ''
   WHILE @I <= LEN(@NAME)
   BEGIN
      SET @C = SUBSTRING(@NAME, @I, 1)
      IF @C <> @CC
      BEGIN
         IF LEN(@OUT) < 4
            SET @OUT = @OUT + @C
         SET @CC = @C
      END
      IF LEN(@OUT) = 4
         BREAK
      SET @I = @I + 1
   END
   SET @SNDX2 = @OUT
   RETURN @SNDX2
END
GO
```

```
/*****************************************************************************/
-- remplacement par substitution de caractères
/*****************************************************************************/
-- exemple : F_TRANSLATE('à Paris...', 'à.', 'a') => 'a paris'
CREATE FUNCTION F_TRANSLATE (@VALIN VARCHAR (8000),
                             @FROM VARCHAR(256), @TO VARCHAR(256))
RETURNS VARCHAR (8000)
AS
BEGIN
-- effets de bord
   IF @VALIN IS NULL
      RETURN NULL
   IF @FROM IS NULL OR @TO IS NULL
      RETURN NULL
```

```
   IF LEN(@VALIN) = 0
      RETURN @VALIN
-- initialisation
   DECLARE @I INTEGER
   DECLARE @OUT VARCHAR(8000)
   SET @OUT = ''
-- lecture caractère par caractère
   SET @I =1
   WHILE @I <= LEN(@VALIN)
   BEGIN
      IF PATINDEX('%' + SUBSTRING(@VALIN, @I, 1)+ '%', @FROM) > 0
      BEGIN
         IF LEN(@TO) >= PATINDEX('%' + SUBSTRING(@VALIN, @I, 1)+ '%', @FROM)
            SET @OUT = @OUT + SUBSTRING(@TO, PATINDEX('%' + SUBSTRING(@VALIN, @I, 1)+ '%',
@FROM), 1)
      END
      ELSE
         SET @OUT = @OUT + SUBSTRING(@VALIN, @I, 1)
      SET @I = @I + 1
   END
   RETURN @OUT
END
GO
```

```
/*****************************************************************************/
-- conversion d'heure décimale en heure minute seconde littérale
/*****************************************************************************/
CREATE FUNCTION F_CONVERT_HD_HMS (@HD FLOAT)
RETURNS VARCHAR(8)
AS

BEGIN

DECLARE @H INTEGER
DECLARE @M INTEGER
DECLARE @S INTEGER
DECLARE @RETVAL VARCHAR(8)

-- cas trivial
IF @HD IS NULL
   RETURN NULL

-- récupération des heures, minutes, secondes
SET @H  = FLOOR(@HD)
SET @HD = @HD - @H
SET @HD = @HD * 60
SET @M  = FLOOR(@HD)
SET @HD = @HD - @M
SET @HD = @HD * 60
SET @S  = FLOOR(@HD)

IF @H < 10
   SET @RETVAL = '0'+CAST(@H AS CHAR(1))+':'
ELSE
   SET @RETVAL = CAST(@H AS CHAR(2))+':'
IF @M < 10
   SET @RETVAL = @RETVAL + '0' + CAST(@M AS CHAR(1))+':'
ELSE
   SET @RETVAL = @RETVAL + CAST(@M AS CHAR(2))+':'
IF @S < 10
   SET @RETVAL = @RETVAL + '0' + CAST(@S AS CHAR(1))
ELSE
   SET @RETVAL = @RETVAL + CAST(@S AS CHAR(2))+':'

RETURN @RETVAL

END
GO
```

```
/*****************************************************************************/
-- affichage heure et minute extrait d'une date
/*****************************************************************************/
CREATE FUNCTION F_DATETIME_AS_HM (@DT DATETIME)
RETURNS CHAR(5) AS
BEGIN
   IF @DT IS NULL RETURN NULL
   DECLARE @H INT
   DECLARE @M INT
   SET @H = DATEPART(HOUR, @DT)
   SET @M = DATEPART(MINUTE, @DT)
   DECLARE @RETVAL VARCHAR(5)
   IF @H < 10
      SET @RETVAL = '0' + CAST(@H AS CHAR(1))+':'
   ELSE
      SET @RETVAL = CAST(@H AS CHAR(2))+':'
   IF @M < 10
```

```
      SET @RETVAL = @RETVAL + '0' + CAST(@M AS CHAR(1))
   ELSE
      SET @RETVAL = @RETVAL + CAST(@M AS CHAR(2))
   RETURN CAST(@RETVAL AS CHAR(5))
END
GO
```

```
/*****************************************************************************/
-- remplace un datetime par une datetime avec date à zero
/*****************************************************************************/
CREATE FUNCTION F_DATETIME_AS_HOUR (@DT DATETIME)
RETURNS DATETIME AS
BEGIN
   RETURN CAST(CAST(@DT AS FLOAT) - FLOOR(CAST(@DT AS FLOAT)) AS DATETIME)
END
GO
```

```
/*****************************************************************************/
-- compléte avec des zéros un nombre converti en chaine de caractères
/*****************************************************************************/
CREATE FUNCTION dbo.F_PAD_ZERO (@INT INTEGER, @NBR_ZERO INTEGER)
RETURNS VARCHAR(8000)
AS
BEGIN

DECLARE @OUT VARCHAR(8000)

-- cas trivial @NBR_ZERO est NULL
IF @NBR_ZERO IS NULL
   RETURN @OUT

-- cas trivial @NBR_ZERO < longueur de @INT
IF @INT IS NOT NULL
   IF @NBR_ZERO < LEN(CAST(@INT AS VARCHAR(8000)))
      RETURN @OUT

-- cas général
SET @OUT = CAST(@INT AS VARCHAR(8000))
WHILE LEN(@OUT) < @NBR_ZERO
BEGIN
   SET @OUT = '0'+@OUT
END

   RETURN @OUT

END
GO
```

```
/*****************************************************************************/
/* conversion code hexa en décimal                                           */
/*****************************************************************************/
CREATE FUNCTION F_HEX_TO_DEC (@HEX VARCHAR(16))
       RETURNS BIGINT

AS
BEGIN
-- effet de bord
   IF @HEX IS NULL
      RETURN NULL
-- valeur basique
   SET @HEX = RTRIM(LTRIM(UPPER(@HEX)))
   IF @HEX = ''
      RETURN NULL
-- signe négatif
   DECLARE @SIGNE VARCHAR(1)
   SET @SIGNE = ''
   IF SUBSTRING(@HEX, 1, 1) = '-'
   BEGIN
      SET @SIGNE = '-'
      SET @HEX = SUBSTRING(@HEX, 2, LEN(@HEX) - 1)
   END
   IF SUBSTRING(@HEX, 1, 1) = '+'
   BEGIN
      SET @SIGNE = ''
      SET @HEX = SUBSTRING(@HEX, 2, LEN(@HEX) - 1)
   END
   SET @HEX = RTRIM(@HEX)
-- conversion
   DECLARE @INT_OUT BIGINT
   DECLARE @CHR CHAR(1)
   DECLARE @I INT
   SET @INT_OUT = 0
   SET @I = 0
   WHILE @I < LEN(@HEX)
   BEGIN
```

```
      SET @CHR = SUBSTRING(@HEX, LEN(@HEX) - @I, 1)
      SET @INT_OUT = @INT_OUT + POWER(16, @I) * CASE @CHR
                                                 WHEN '0' THEN 0
                                                 WHEN '1' THEN 1
                                                 WHEN '2' THEN 2
                                                 WHEN '3' THEN 3
                                                 WHEN '4' THEN 4
                                                 WHEN '5' THEN 5
                                                 WHEN '6' THEN 6
                                                 WHEN '7' THEN 7
                                                 WHEN '8' THEN 8
                                                 WHEN '9' THEN 9
                                                 WHEN 'A' THEN 10
                                                 WHEN 'B' THEN 11
                                                 WHEN 'C' THEN 12
                                                 WHEN 'D' THEN 13
                                                 WHEN 'E' THEN 14
                                                 WHEN 'F' THEN 15
                                                 ELSE NULL
                                               END
      SET @I = @I + 1
   END
   RETURN @INT_OUT * CASE @SIGNE WHEN '-' THEN -1 ELSE 1 END
END
GO
-- liste toutes les colonnes d'une table
CREATE FUNCTION F_LISTE_COLS (@TABLE_NAME VARCHAR(128))
   RETURNS VARCHAR(8000)
AS

BEGIN

-- liste des noms de colonnes dans
   DECLARE @ColumnList VARCHAR(8000)
   SET @ColumnList = ''

-- obtention de la liste des colonnes pour la requête de recherche
   SELECT @ColumnList = @ColumnList + COLUMN_NAME +', '
   FROM   INFORMATION_SCHEMA.COLUMNS
   WHERE  TABLE_NAME = @TABLE_NAME

   RETURN SUBSTRING(@ColumnList, 1, LEN(@ColumnList) -1)

END
GO

-- exemple d'utilisation :
SELECT dbo.F_LISTE_COLS('T_CONTACT_CTC')
```

```
/*************************************************************
  CALCUL DES SAISONS - basé sur l'algorithmes de Meeus
 ("Astronomical Algorithms", 1991) modifié par Simon Cassidy
-------------------------------------------------------------
 Frédéric BROUARD - SQLpro - 2004-06-27 - Transact SQL
*************************************************************/

CREATE FUNCTION F_SUB_SEASONS (@JDME FLOAT(50))
   RETURNS DATETIME
AS
-- sous fonction utilisée par les 4 calculs de saisons
BEGIN

IF @JDME IS NULL
   RETURN NULL

DECLARE @T FLOAT(50)
DECLARE @S FLOAT(50)
DECLARE @W FLOAT(50)
DECLARE @L FLOAT(50)
DECLARE @JD FLOAT(50)
DECLARE @D DATETIME

SET @T = (@JDME - 2451545.0) / 36525

SET @S =
  485 * COS(0.43563418129778464 +   33.757041381353048   * @T) +
  203 * COS(0.64978608051748876 +  575.33848531501758    * @T) +
  199 * COS(0.73443454923921381 +    0.35231216280757538 * @T) +
  182 * COS(0.48607419668042079 + 7771.3771552463541     * @T) +
  156 * COS(1.2765338149086527  +  786.04194554533876    * @T) +
  136 * COS(1.2482594810263443  +  393.02097277266938    * @T) +
   77 * COS(0.39339721339952183 + 1150.6769706300352     * @T) +
   74 * COS(1.6880824525289155  +   52.969102188531025   * @T) +
   70 * COS(0.76061448801912879 +  157.7343580417903     * @T) +
   58 * COS(0.34574972482007665 +  588.4926828214484     * @T) +
   52 * COS(1.69593643416289    +    2.6298272103200158  * @T) +
```

```
   50 * COS(0.366868208769208   +   39.81490468210017    * @T) +
   45 * COS(0.82972952639810416 +  522.36940057977904    * @T) +
   44 * COS(0.43895030687657388 +  550.75533081445974    * @T) +
   29 * COS(1.063429113240145   +   77.552256689088878   * @T) +
   18 * COS(0.96202548369927443 + 1179.0629008647159     * @T) +
   17 * COS(1.5496778428457652  +   79.629809364200341   * @T) +
   16 * COS(1.7111207986552408  + 1097.7078858947966     * @T) +
   14 * COS(1.7411404617895434  +  548.67777813934822    * @T) +
   12 * COS(1.6648695734773908  +  254.43144545527034    * @T) +
   12 * COS(1.5203563114122605  +  557.31427814345443    * @T) +
   12 * COS(0.36320301734001997 +  606.97767436883066    * @T) +
    9 * COS(0.48397980157802756 +   21.32991313471798    * @T) +
    8 * COS(0.2696533694331239  +  294.24635013737048    * @T)

SET @W = ( 35999.373*@T - 2.47 ) * PI() / 180

SET @L = 1 + 0.0334*COS(@W) + 0.0007*COS(2*@W)

SET @JD = @JDME + (0.00001*@S/@L)

SET @JD = @JD - 2415020.50208142228

SET @D = CAST(FLOOR(@JD) AS DATETIME)

RETURN @D

END

GO

CREATE FUNCTION F_WINTER_DATE (@Y INT)
   RETURNS DATETIME
AS

-- calcul de la date de début d'hiver

BEGIN

-- @Y est l'année considérée

-- effets de bord : année absente
IF @Y IS NULL
   RETURN NULL

-- limites de calculs de l'an 1000 à l'an 3000
IF NOT @Y BETWEEN 1000 AND 3000
   RETURN NULL

DECLARE @M FLOAT(50)
DECLARE @JDME FLOAT(50)

SET @M = (CAST(@Y AS FLOAT(50)) - 2000.0) / 1000.0
SET @JDME = 2451900.05952
           + 365242.74049 * @M
           -      0.06223 * SQUARE(@M)
           -      0.00823 * POWER (@M, 3)
           +      0.00032 * POWER (@M, 4)

RETURN dbo.F_SUB_SEASONS (@JDME)

END
GO

CREATE FUNCTION F_AUTUMN_DATE (@Y INT)
   RETURNS DATETIME
AS

-- calcul de la date de début d'automne

BEGIN

-- @Y est l'année considérée

-- effets de bord : année absente
IF @Y IS NULL
   RETURN NULL

-- limites de calculs de l'an 1000 à l'an 3000
IF NOT @Y BETWEEN 1000 AND 3000
   RETURN NULL

DECLARE @M FLOAT(50)
DECLARE @JDME FLOAT(50)

SET @M = (CAST(@Y AS FLOAT(50)) - 2000.0) / 1000.0
SET @JDME = 2451810.21715
```

```
           + 365242.01767 * @M
           -      0.11575 * SQUARE(@M)
           +      0.00337 * POWER (@M, 3)
           +      0.00078 * POWER (@M, 4)

RETURN dbo.F_SUB_SEASONS (@JDME)

END
GO

CREATE FUNCTION F_SUMMER_DATE (@Y INT)
   RETURNS DATETIME
AS

-- calcul de la date de début de l'été

BEGIN

-- @Y est l'année considérée

-- effets de bord : année absente
IF @Y IS NULL
   RETURN NULL

-- limites de calculs de l'an 1000 à l'an 3000
IF NOT @Y BETWEEN 1000 AND 3000
   RETURN NULL

DECLARE @M FLOAT(50)
DECLARE @JDME FLOAT(50)

SET @M = (CAST(@Y AS FLOAT(50)) - 2000.0) / 1000.0
SET @JDME = 2451716.56767
           + 365241.62603 * @M
           +      0.00325 * SQUARE(@M)
           +      0.00888 * POWER (@M, 3)
           -      0.00030 * POWER (@M, 4)

RETURN dbo.F_SUB_SEASONS (@JDME)

END
GO

CREATE FUNCTION F_SPRING_DATE (@Y INT)
   RETURNS DATETIME
AS

-- calcul de la date de début du printemps

BEGIN

-- @Y est l'année considérée

-- effets de bord : année absente
IF @Y IS NULL
   RETURN NULL

-- limites de calculs de l'an 1000 à l'an 3000
IF NOT @Y BETWEEN 1000 AND 3000
   RETURN NULL

DECLARE @M FLOAT(50)
DECLARE @JDME FLOAT(50)
DECLARE @T FLOAT(50)
DECLARE @S FLOAT(50)
DECLARE @W FLOAT(50)
DECLARE @L FLOAT(50)
DECLARE @JD FLOAT(50)
DECLARE @D DATETIME

SET @M = (CAST(@Y AS FLOAT(50)) - 2000.0) / 1000.0
SET @JDME = 2451623.80984
           + 365242.37404 * @M
           +      0.05169 * SQUARE(@M)
           -      0.00411 * POWER (@M, 3)
           -      0.00057 * POWER (@M, 4)

RETURN dbo.F_SUB_SEASONS (@JDME)

END
GO

/***********************
 EXEMPLE D'UTILISATION
***********************/
```

```
SELECT 'PRINTEMPS' AS SAISON, dbo.F_SPRING_DATE(2000) AS DATE_DEBUT
UNION
SELECT 'ÉTÉ'       AS SAISON, dbo.F_SUMMER_DATE(2000) AS DATE_DEBUT
UNION
SELECT 'AUTOMNE'   AS SAISON, dbo.F_AUTUMN_DATE(2000) AS DATE_DEBUT
UNION
SELECT 'HIVER'     AS SAISON, dbo.F_WINTER_DATE(2000) AS DATE_DEBUT
ORDER BY 2
```

```
/*************************************************************
  CONVERSION DE NOMBRE EN LITTERAUX :
  sous fonctions :
     F_NEC_20    : transformation des nombres de 1 à 19 en littéraux
     F_NEC_100   : transformation des nombres de 20 à 99 en littéraux
     F_NEC_0_100 : transformation des nombres de 0 à 100 en littéraux
  fonction principale :
     F_NOMBRE_EN_CHIFFRE : transformation de n'importe quel nombre
     entier de l'intervalle [- 2 147 483 648 ; 2 147 483 647]
-------------------------------------------------------------
 Frédéric BROUARD - SQLpro - 2004-08-07 - Transact SQL
*************************************************************/

CREATE FUNCTION F_NEC_20 (@I INT)
   RETURNS VARCHAR(16)
AS

/*****************************************************************************
* Frédéric BROUARD - SQLpro - 2004-08-07                                     *
* Sous procédure de transformation des nombres de 1 à 19 en littéraux        *
*****************************************************************************/

BEGIN

DECLARE @RETVAL VARCHAR(256)

SET @RETVAL =
CASE
   WHEN @I=1 THEN 'UN'
   WHEN @I=2 THEN 'DEUX'
   WHEN @I=3 THEN 'TROIS'
   WHEN @I=4 THEN 'QUATRE'
   WHEN @I=5 THEN 'CINQ'
   WHEN @I=6 THEN 'SIX'
   WHEN @I=7 THEN 'SEPT'
   WHEN @I=8 THEN 'HUIT'
   WHEN @I=9 THEN 'NEUF'
   WHEN @I=10 THEN 'DIX'
   WHEN @I=11 THEN 'ONZE'
   WHEN @I=12 THEN 'DOUZE'
   WHEN @I=13 THEN 'TREIZE'
   WHEN @I=14 THEN 'QUATORZE'
   WHEN @I=15 THEN 'QUINZE'
   WHEN @I=16 THEN 'SEIZE'
   WHEN @I=17 THEN 'DIX-SEPT'
   WHEN @I=18 THEN 'DIX-HUIT'
   WHEN @I=19 THEN 'DIX-NEUF'
END

RETURN @RETVAL

END

GO



CREATE FUNCTION F_NEC_100 (@I INT)
   RETURNS VARCHAR(32)
AS

/*****************************************************************************
* Frédéric BROUARD - SQLpro - 2004-08-07                                     *
* Sous procédure de transformation des nombres de 20 à 99 en littéraux       *
*****************************************************************************/

BEGIN

DECLARE @RETVAL VARCHAR(256)

IF NOT(@I BETWEEN 20 AND 99)
   RETURN @RETVAL

DECLARE @U  CHAR(1)

SET @U = SUBSTRING(CAST(@I AS CHAR(2)), 2, 1)
```

```
SET @RETVAL =

CASE
   WHEN @I = 20 THEN 'VINGT'
   WHEN @I = 21 THEN 'VINGT ET UN'
   WHEN @I BETWEEN 22 AND 29 THEN 'VING-' + dbo.F_NEC_20(CAST(@U AS INTEGER))

   WHEN @I = 30 THEN 'TRENTE'
   WHEN @I = 31 THEN 'TRENTE ET UN'
   WHEN @I BETWEEN 32 AND 39 THEN 'TRENTE-' + dbo.F_NEC_20(CAST(@U AS INTEGER))

   WHEN @I = 40 THEN 'QUARANTE'
   WHEN @I = 41 THEN 'QUARANTE ET UN'
   WHEN @I BETWEEN 42 AND 39 THEN 'QUARANTE-' + dbo.F_NEC_20(CAST(@U AS INTEGER))

   WHEN @I = 50 THEN 'CINQUANTE'
   WHEN @I = 51 THEN 'CINQUANTE ET UN'
   WHEN @I BETWEEN 52 AND 59 THEN 'CINQUANTE-' + dbo.F_NEC_20(CAST(@U AS INTEGER))

   WHEN @I = 60 THEN 'SOIXANTE'
   WHEN @I = 61 THEN 'SOIXANTE ET UN'
   WHEN @I BETWEEN 62 AND 69 THEN 'SOIXANTE-' + dbo.F_NEC_20(CAST(@U AS INTEGER))

   WHEN @I = 70 THEN 'SOIXANTE-DIX'
   WHEN @I = 71 THEN 'SOIXANTE ET ONZE'
   WHEN @I BETWEEN 72 AND 79 THEN 'SOIXANTE-' + dbo.F_NEC_20(CAST(@U AS INTEGER)+10)

   WHEN @I = 80 THEN 'QUATRE-VINGT'
   WHEN @I BETWEEN 81 AND 89 THEN 'QUATRE-VINGT-' + dbo.F_NEC_20(CAST(@U AS INTEGER))
   WHEN @I BETWEEN 90 AND 99 THEN 'QUATRE-VINGT-' + dbo.F_NEC_20(CAST(@U AS INTEGER)+10)

END

RETURN @RETVAL

END

GO

CREATE FUNCTION F_NEC_0_100 (@I INT)
   RETURNS VARCHAR(256)
AS

/*************************************************************************
* Frédéric BROUARD - SQlpro - 2004-08-07                                 *
* Sous procédure de transformation des nombres de 0 à 100 en littéraux   *
*************************************************************************/

BEGIN

IF @I = 0 RETURN 'ZÉRO'

IF @I BETWEEN 1 AND 19 RETURN dbo.F_NEC_20 (@I)

IF @I BETWEEN 20 AND 99 RETURN dbo.F_NEC_100 (@I)

IF @I = 100 RETURN 'CENT'

RETURN ''

END

GO


CREATE FUNCTION F_NOMBRE_EN_CHIFFRE (@I INTEGER)
   RETURNS VARCHAR(256)
AS

/*************************************************************************
* Frédéric BROUARD - SQLpro - 2004-08-07                                 *
* Fonction de transformation des nombres de 0 à 2 147 483 647 en littéraux *
*************************************************************************/

BEGIN

IF @I IS NULL RETURN NULL

DECLARE @SIGN VARCHAR(5)

IF @I < 0
BEGIN
   SET @SIGN = 'MOINS'
   SET @I = -1 * @I
END
```

```
IF @I BETWEEN 0 AND 100 RETURN COALESCE(@SIGN + ' ', '') + dbo.F_NEC_0_100 (@I)

-- le nombre est supérieur à 100

DECLARE @IS VARCHAR(10)
SET @IS = CAST(@I AS VARCHAR(10))

WHILE LEN(@IS) < 10
   SET @IS = '0'+@IS

DECLARE @D11 INT              -- chiffres des unité et dizaine
DECLARE @D100 INT             -- chiffre des centaines
DECLARE @D1000 INT            -- chiffre des milliers
DECLARE @D10000 INT           -- chiffre des dix-milliers
DECLARE @D100000 INT          -- chiffre des cent-milliers
DECLARE @D1000000 INT         -- chiffre des millions
DECLARE @D10000000 INT        -- chiffre des dix-millions
DECLARE @D100000000 INT       -- chiffre des cent-millions
DECLARE @D1000000000 INT      -- chiffre des milliards

SET @D11 = CAST(SUBSTRING(@IS, 9, 2) AS INTEGER)
SET @D100 = CAST(SUBSTRING(@IS, 8, 1) AS INTEGER)
SET @D1000 = CAST(SUBSTRING(@IS, 7, 1) AS INTEGER)
SET @D10000 = CAST(SUBSTRING(@IS, 6, 1) AS INTEGER)
SET @D100000 = CAST(SUBSTRING(@IS, 5, 1) AS INTEGER)
SET @D1000000 = CAST(SUBSTRING(@IS, 4, 1) AS INTEGER)
SET @D10000000 = CAST(SUBSTRING(@IS, 3, 1) AS INTEGER)
SET @D100000000 = CAST(SUBSTRING(@IS, 2, 1) AS INTEGER)
SET @D1000000000 = CAST(SUBSTRING(@IS, 1, 1) AS INTEGER)

DECLARE @RETVAL VARCHAR(256)
SET @RETVAL = ''

-- traitement des milliards
IF @D1000000000 <> 0
   SET @RETVAL = @RETVAL + dbo.F_NEC_20 (@D1000000000)+' MILLIARD'


-- traitement des millions
IF @D100000000 = 1
   SET @RETVAL = @RETVAL + ' CENT'
IF @D100000000 > 1
   SET @RETVAL = @RETVAL + ' ' + dbo.F_NEC_20 (@D100000000) + ' CENT'
-- exception de l's à la centaine pure
IF @D100000000 > 1 AND @D10000000 + @D1000000 = 0
   SET @RETVAL = @RETVAL + 'S'
IF @D10000000 * 10 + @D1000000 <> 0
   SET @RETVAL =  @RETVAL + ' ' + dbo.F_NEC_0_100 (@D10000000 * 10 + @D1000000)
IF @D100000000 * 100 + @D10000000 * 10 + @D1000000 <> 0
   SET @RETVAL = @RETVAL + ' MILLION'

-- traitement des milliers
IF @D100000 = 1
   SET @RETVAL = @RETVAL + ' CENT'
IF @D100000 > 1
   SET @RETVAL = @RETVAL + ' ' + dbo.F_NEC_20 (@D100000) + ' CENT'
-- exception de l's à la centaine pure
IF @D100000 > 1 AND @D10000 = 0 AND @D1000 = 0
   SET @RETVAL = @RETVAL + 'S'
IF @D10000 * 10 + @D1000 <> 0
   SET @RETVAL = @RETVAL + ' ' + dbo.F_NEC_0_100 (@D10000 * 10 + @D1000)
IF @D100000 * 100 + @D10000 * 10 + @D1000 <> 0
   SET @RETVAL = ' ' + @RETVAL + ' MILLE'

-- traitement des centaines
IF @D100 > 1
   SET @RETVAL = @RETVAL + ' ' +  dbo.F_NEC_20 (@D100)
IF @D100 > 0
   SET @RETVAL = @RETVAL + ' CENT'
-- exception de l's à la centaine pure
IF @D100 > 1 AND @D11 = 0
   SET @RETVAL = @RETVAL + 'S'
IF @D11 <> 0
   SET @RETVAL = @RETVAL + ' ' + dbo.F_NEC_0_100(@D11)

RETURN COALESCE(@SIGN + ' ', '') + LTRIM(@RETVAL)

END

GO

-- exemple d'utilisation
SELECT dbo.F_NOMBRE_EN_CHIFFRE(-2111623500) AS NOMBRE_EN_LETTRE
```

```
CREATE FUNCTION F_STORAGE_SIZE_KB (@TABLE_NAME VARCHAR(128), @NATURE CHAR(1))
   RETURNS BIGINT
```

```
AS

/*****************************************************************************
  Obtention de la taille des informations stockées pour une table spécifique
*****************************************************************************
  Frédéric BROUARD - SQLpro - 2004-08-11
*****************************************************************************
  ATTENTION, ces données sont basées sur les informations contenues dans les
  tables systèmes de descriptions des fichiers. Ces tables ne sont pas
  forcément toujours à jour au moment de l'exécution de la fonction.
  Pour s'assurer de l'exactitude des informations, veuillez procéder
  préalablement à l'exécution de la commande DBCC UPDATEUSAGE pour la table
  spécifiée
*****************************************************************************
  FONCTIONNEMENT : (exemples)
  1)  SELECT dbo.F_STORAGE_SIZE_KB ('MaTable', 'I')
       => donne le cout de stockage en kilo octets des index de la table
          MaTable
  2)  SELECT dbo.F_STORAGE_SIZE_KB ('MaTable', 'D')
       => donne le cout de stockage en kilo octets des données de la table
          MaTable
  3)  SELECT dbo.F_STORAGE_SIZE_KB ('MaTable', '')
      => donne le cout de stockage en kilo octets des données et index de la
         table MaTable
*****************************************************************************/

BEGIN

IF @TABLE_NAME IS NULL
   RETURN NULL

IF NOT EXISTS(SELECT *
              FROM INFORMATION_SCHEMA.TABLES
              WHERE TABLE_NAME = @TABLE_NAME
                AND TABLE_TYPE = 'BASE TABLE')
   RETURN NULL

DECLARE @id       INT
declare @DATA_SIZE BIGINT
declare @INDX_SIZE BIGINT

SELECT @id = id
FROM   sysobjects
WHERE  name = @TABLE_NAME

IF @id IS NULL
   RETURN NULL

SET @DATA_SIZE = 0
SET @INDX_SIZE = 0

-- la taille des données d'un objet
   set @DATA_SIZE = (SELECT SUM(dpages)
                     FROM   sysindexes
                     WHERE  indid < 2
                       AND  id = @id)
                  + (SELECT COALESCE(sum(used), 0)
                     FROM   sysindexes
                     WHERE  indid = 255
                       AND  id = @id)

-- la taille des index d'un objet
   set @INDX_SIZE = (SELECT sum(used)
                     FROM   sysindexes
                     WHERE  indid in (0, 1, 255)
                     AND  id = @id)
                  - @DATA_SIZE

IF @NATURE = 'I' SET @DATA_SIZE = 0
IF @NATURE = 'D' SET @INDX_SIZE = 0

-- cooroboration avec les unités de mesure physique de stockage du système
SELECT @DATA_SIZE = (@DATA_SIZE + @INDX_SIZE) * low / 1024.0
FROM   master.dbo.spt_values
WHERE  number = 1
  AND  type = 'E'

RETURN @DATA_SIZE

END

GO
```

```
CREATE FUNCTION F_DROP_CHARS (@DATA VARCHAR(128), @CHARS_TO_DROP VARCHAR(220))
   RETURNS VARCHAR(128)
AS
```

```
/****************************************************************************
  Suppression de caractères dans une chaine de caractères
****************************************************************************
  Frédéric BROUARD - SQlpro - 2004-08-11
****************************************************************************
  La fonction F_DROP_CHARS supprime tous les caractères contenus dans la
  chaine de caractères @CHARS_TO_DROP au sein de la chaine @DATA.
  Exemple : SELECT dbo.F_DROP_CHARS('Locomotive', 'lot')
            => Lcmive
****************************************************************************/

BEGIN

IF @DATA IS NULL OR @CHARS_TO_DROP IS NULL
   RETURN NULL

IF @DATA = ''
   RETURN ''

IF @CHARS_TO_DROP = ''
   RETURN @DATA

DECLARE @NEW_DATA VARCHAR(128)
SET @NEW_DATA = ''
DECLARE @I INT
SET @I = 1
DECLARE @C CHAR(1)

WHILE @I <= LEN(@DATA)
BEGIN
   SET @C = SUBSTRING(@DATA, @I, 1)
   IF CHARINDEX(@C, @CHARS_TO_DROP) = 0
      SET @NEW_DATA = @NEW_DATA + @C
   SET @I = @I +1
END

RETURN @NEW_DATA

END

GO
```

```
CREATE VIEW V_DATEHEURE_COURANTE
AS
   SELECT CURRENT_TIMESTAMP AS DATEHEURE_COURANTE
GO
```

```
CREATE FUNCTION F_DATE_HEURE_FORMAT_COMPACT ()
   RETURNS CHAR(16)
AS

/****************************************************************************
  Obtention d'une dateheure au format compact AAAAMMJJHHMMSS
****************************************************************************
  Frédéric BROUARD - sqlpro - 2004-08-11
****************************************************************************
  NOTA : cette fonction est basée sur la vue V_DATEHEURE_COURANTE implémentée
  ci dessus.
****************************************************************************/

BEGIN
DECLARE @DH CHAR(16)

SELECT @DH = CONVERT(CHAR(8), DATEHEURE_COURANTE, 112)
           + REPLACE(CONVERT(CHAR(8), DATEHEURE_COURANTE, 108), ':', '')
FROM V_DATEHEURE_COURANTE

RETURN @DH
END

GO
```

```
/****************************************************************************
  Recherche de correspondance de chaîne basé sur les expressions régulières
****************************************************************************
  Frédéric BROUARD - sqlpro - 2004-08-11
****************************************************************************
  NOTA : cette se comporte à la manière du prédicat SIMILAR de la norme
         SQL:1999 qui simule un REGEX. Elle utilise des appels OLE à un REGEX
         écrit en VB script et disponible sur toutes les éditions de Windows
****************************************************************************/

CREATE FUNCTION dbo.F_REGEXMATCH (@MOTIF   VARCHAR(256),
                                  @VALEUR  VARCHAR(8000),
```

```
                                        @NOCASSE BIT)

RETURNS int
AS
BEGIN
       declare @OLE_OBJECT int, @RETVAL int, @MATCH bit
       set @MATCH=0
       exec @RETVAL=sp_OACreate 'VBScript.RegExp',@OLE_OBJECT OUT
       IF (@RETVAL <> 0) RETURN NULL
       exec @RETVAL=sp_OASetProperty @OLE_OBJECT, 'Pattern', @MOTIF
       IF (@RETVAL <> 0) RETURN NULL
       exec @RETVAL=sp_OASetProperty @OLE_OBJECT, 'IgnoreCase', @NOCASSE
       IF (@RETVAL <> 0) RETURN NULL
       exec @RETVAL=sp_OAMethod @OLE_OBJECT, 'Test',@MATCH OUT, @VALEUR
       IF (@RETVAL <> 0) RETURN NULL
       exec @RETVAL=sp_OADestroy @OLE_OBJECT
       return @MATCH
END

GO

-- exemple : liste de tous les clients dont le nom commence avec la lettre "g"
-- avec respect de la casse

SELECT *
FROM   DB_HOTEL.dbo.T_CLIENT
WHERE  dbo.F_REGEXMATCH('g.*', CLI_NOM, 0) <> 0

-- sans respect de la casse
SELECT *
FROM   DB_HOTEL.dbo.T_CLIENT
WHERE  dbo.F_REGEXMATCH('g.*', CLI_NOM, 1) <> 0
```