

Les fonctions

Une fonction est un ensemble de lignes de code stocké dans le système, qui exécute à la demande, une tâche pour l'utilisateur et renvoie un résultat. Une fonction demande souvent que **des paramètres soient fournis** en entrée.

Le résultat renvoyé doit ensuite être affiché ou inclus dans une expression ou une requête. Une fonction peut bien sûr utiliser le résultat d'une autre fonction

- Le nom de la fonction est ***toujours suivi de parenthèses***, même si aucun paramètre n'est fourni ou attendu
- Les fonctions renvoient des valeurs de types divers. Le tableau suivant classe les fonctions présentées dans cette formation, selon le type de valeur qu'elles renvoient

Type retourné	Noms des fonctions
numérique	datepart, charindex, len, abs, modulo
chaîne de caractères	substring, upper, lower, replace, trim
datetime	getdate()

Les fonctions : **CONVERT**

CONVERT (NOUVEAU_TYPE, valeur_à_convertir)

CONVERT (TYPE_CHAÎNE_DE_CARACTÈRES, date_à_convertir, format_date)

La fonction « **CONVERT** » attend 2 paramètres en entrée (éventuellement 3 lorsque l'élément à convertir est une date) et **renvoie une valeur correspondant au deuxième paramètre dans le type spécifié par le premier** (et dans le format précisé par le troisième, le cas échéant **[100-114]**)

```
SELECT CONVERT (varchar, birth_date, 110)
      as [Date de naissance]
FROM student
```

Date de naissance
05-17-1944
05-31-1930
08-25-1930
08-17-1942

Les fonctions : **GETDATE()**

```
SELECT GETDATE ()  
      , CONVERT (varchar, GETDATE (), 109)  
      , CONVERT (date, GETDATE ())  
      , CONVERT (time, GETDATE ())
```

Sous SQL-Server, la fonction « **GETDATE()** » renvoie la date et l'heure actuelles

Type retourné : DATETIME

Date du jour	Date du jour formatée	Date uniquement	Heure uniquement
2014-05-14 14:21:09.210	May 14 2014 2:21:09:210PM	2014-05-14	14:21:09.2130000

Les fonctions : **DATEPART**

DATEPART (*partie_de_date_à_extraire, date_traitée*)

La fonction « **DATEPART** » extrait une partie d'une date donnée

Type retourné : NOMBRE

```
SELECT DATEPART (mm, GETDATE ( ) )  
      , DATEPART (dy, GETDATE ( ) )  
      , DATEPART (ns, GETDATE ( ) )
```

Mois	Jour de l'année	Nanosecondes
5	134	153000000

<http://msdn.microsoft.com/fr-be/library/ms174420.aspx>

Les fonctions : **CHARINDEX**

CHARINDEX (*chaine_de_caractères_recherchée, valeur_à_évaluer*)

La fonction « **CHARINDEX** » renvoie la position du début de l'occurrence d'une chaîne de caractère dans une autre

Type retourné : NOMBRE

```
SELECT CHARINDEX('i', 'Kim Basinger')
       , CHARINDEX('08', 'Basinger 08/12/1953')
       , CHARINDEX('y', 'Kim Basinger')
       , CHARINDEX('', 'Kim Basinger')
```

position du premier i	position du 08	pas de y	chaîne vide
2	10	0	0

Les fonctions : **LEN**

LEN (*chaine_de_caractères_à_mesurer*)

La fonction « **LEN** » renvoie le nombre de lettres composant une chaine de caractères donnée, espaces blancs compris

Type retourné : NOMBRE

```
SELECT LEN('Kim Basinger')
```

Longueur de la chaine de caractères

12

Les fonctions : **ABS**

ABS (*nombre*)

La fonction « **ABS** » renvoie la valeur absolue du nombre passé en paramètre

Type retourné : NOMBRE

```
SELECT ABS (-1.0), ABS (0.0), ABS (1.0)  
SELECT ABS (-2147483648)
```

val1	val2	val3
1.0	0.0	1.0

```
Msg 8115, Level 16, State 2, Line 1  
Arithmetic overflow error converting expression to data type int.
```

Les fonctions : Modulo

dividende % diviseur

Le « % », qui représente la fonction « **modulo** » que l'on rencontre fréquemment dans d'autres langages également, **renvoie le reste de la division ENTIÈRE du premier nombre (dividende) par le second (diviseur)**. Cette fonction permet de savoir si le premier chiffre est multiple du second

Type retourné : NOMBRE

```
SELECT 38 / 5, 38 % 5
```

$$\begin{array}{r} 38,0 \\ 35 \\ \hline 30 \\ 30 \\ \hline 0 \end{array} \quad \begin{array}{r} 5 \\ 7,6 \end{array}$$

Division entière	Reste
7	3

Les fonctions : **SUBSTRING**

SUBSTRING (*chaine_de_caractères, position_départ, nombre_caractères*)

La fonction « **SUBSTRING** » renvoie une chaîne de caractère d'une longueur souhaitée, à partir d'une position donnée, à l'intérieur d'une chaîne de caractères passée en paramètre

Type retourné : CHAÎNE DE CARACTÈRES

```
SELECT SUBSTRING ('Basinger', 4, 3)
SELECT last_name, SUBSTRING(first_name, 1, 1)
FROM student
```

Caractères 4, 5 et 6
ing

last_name	Initiale du prenom
Lucas	G
Eastwood	C
Connery	S
De Niro	R
Bacon	K

Les fonctions : **UPPER** et **LOWER**

UPPER (*chaine_de_caractères*)

LOWER (*chaine_de_caractères*)

Les fonctions « **UPPER** » et « **LOWER** » renvoient la chaîne de caractères passée en paramètres, respectivement en majuscules ou en minuscules

Type retourné : CHAÎNE DE CARACTÈRES

```
SELECT UPPER(last_name), LOWER(first_name)
FROM student
WHERE UPPER(first_name) LIKE 'TOM'
```

Nom de famille	Prénom
CRUISE	tom
HANKS	tom

Les fonctions : **REPLACE**

REPLACE (*chaine_de_caractères_traitée, caract_à_remplacer, nouveau_caract*)

La fonction « **REPLACE** » remplace les caractères demandés par d'autres

Type retourné : CHAÎNE DE CARACTÈRES

```
SELECT REPLACE(' Kim Basinger ', ' ', '')  
      , REPLACE('11110000101010', '1', '0')
```

Sans espaces	Sans 1
KimBasinger	0000000000000000

Les fonctions : LTRIM et RTRIM

LTRIM (*chaine_de_caractères*)

RTRIM (*chaine_de_caractères*)

Les fonctions « **LTRIM** » et « **RTRIM** » renvoient la chaîne de caractères passée en paramètres épurée des espaces blancs éventuellement présents en début ou en fin de chaîne, respectivement

Type retourné : CHAÎNE DE CARACTÈRES

```
SELECT LTRIM('    Kim Basinger    ')  
      , RTRIM('    Kim Basinger    ')  
      , LTRIM(RTRIM('    Kim Basinger    '))
```

LTRIM	RTRIM	LTRIM de RTRIM
Kim Basinger	Kim Basinger	Kim Basinger

Les fonctions : Agrégations

Une fonction d'agrégation est une fonction particulière qui attend comme paramètres **un ensemble de valeurs** (une colonne) et qui présente en retour **un seul** résultat agrégé (regroupé) sur ces valeurs. Sauf exceptions, les valeurs **NULL** ne sont pas prises en compte

Fonctions d'agrégation principales

Fonction	Description
COUNT	Nombre total de valeurs contenues dans la table/colonne
MAX	Valeur numérique la plus élevée
MIN	Plus petite valeur numérique disponible
SUM	Somme de l'ensemble des valeurs de la colonne
AVG	Moyenne de l'ensemble des valeurs de la colonne

Les fonctions : **COUNT**

COUNT (*)
COUNT (colonne)
COUNT (DISTINCT colonne)

La fonction d'agrégation « **COUNT** » renvoie le nombre total de valeur contenues dans la table ou la colonne à laquelle on applique la fonction. Les valeurs « **NULL** » ne sont prises en compte que dans l'utilisation du « **COUNT(*)** »

Type retourné : NOMBRE (INTEGER)

```
SELECT COUNT(*), COUNT(first_name), COUNT(DISTINCT first_name)  
FROM student
```

Total des lignes	Total des prénoms	Total des prénoms sans doublons
25	25	23

Les fonctions : **MAX** et **MIN**

MAX (*colonne*)

MIN (*colonne*)

Les fonctions d'agrégation « **MAX** » et « **MIN** » renvoient respectivement la plus grande ou la plus petite des valeurs contenues dans une colonne donnée

Type retourné : NOMBRE

```
SELECT MAX (year_result), MIN (year_result*5)  
      , MAX (LEN(last_name))  
FROM student
```

Résultat le plus élevé	Pourcentage le plus faible	Taille du nom le plus long
19	10	15

Les fonctions : **SUM**

SUM (*colonne*)

La fonction « **SUM** » renvoie la somme des valeurs d'une colonne

Type retourné : NOMBRE

```
SELECT SUM (year_result), SUM (year_result) / COUNT (*)  
FROM student
```

Somme des résultats annuels	Moyenne générale
219	8

Les fonctions : **AVG**

AVG (colonne)

La fonction « **AVG** » renvoie la moyenne de l'ensemble des valeurs contenues dans une colonne

Type retourné : NOMBRE

```
SELECT AVG (year_result)
      , AVG (DATEPART(yy,GETDATE()) - DATEPART(yy,birth_date))
FROM student
```

Moyenne générale	Moyenne d'âge
8	53

Les fonctions : **CASE**

```
CASE  
  WHEN expression1 THEN valeur1  
  WHEN expression2 THEN valeur2  
  ...  
  WHEN expressionN THEN valeurN  
  ELSE valeur_par_défaut  
END
```

- **L'instruction « CASE »** peut être utilisée afin de modifier l'affichage des éléments d'une colonne selon ce que l'on souhaite
- Dès qu'une expression contenue dans l'une des clauses « **WHEN** » est évaluée à « **TRUE** », la valeur contenue après la clause « **THEN** » est affichée dans la colonne et **l'instruction « CASE » se termine** et est réévaluée en totalité pour la ligne suivante
- Si aucune des expressions évaluées après les clauses « **WHEN** » n'est validée, la valeur affichée dans la colonne correspond à la valeur présentée dans la clause « **ELSE** »

Les fonctions : CASE

```
SELECT last_name, first_name, year_result,  
CASE  
    WHEN year_result BETWEEN 18 AND 20 THEN 'Excellent'  
    WHEN year_result BETWEEN 16 AND 17 THEN 'Très Bien'  
    WHEN year_result BETWEEN 14 AND 15 THEN 'Bien'  
    WHEN year_result BETWEEN 12 AND 13 THEN 'Suffisant'  
    WHEN year_result BETWEEN 10 AND 11 THEN 'Faible'  
    WHEN year_result BETWEEN 8 AND 9 THEN 'Insuffisant'  
    ELSE 'Insuffisance Grave'  
END AS [Note globale]  
FROM student
```

last_name	first_name	year_result	Note globale
Lucas	Georges	10	Faible
Eastwood	Clint	4	Insuffisance Grave
Connery	Sean	12	Suffisant
De Niro	Robert	3	Insuffisance Grave
Bacon	Kevin	16	Très Bien
Basinger	Kim	19	Excellent
Depp	Johnny	11	Faible

Les fonctions : CASE

```
CASE colonne_à_évaluer  
  WHEN valeur_de_comparaison1 THEN valeur1  
  ...  
  ELSE valeur_par_défaut  
END
```

Lorsque les expressions à évaluer sont des **égalités strictes**, il est possible de simplifier l'écriture du « **CASE** » en reprenant le nom de la colonne à évaluer directement après le mot-clé « **CASE** »

```
SELECT student_id, first_name,  
CASE section_id  
  WHEN 1010 THEN 'BSc Management'  
  WHEN 1320 THEN 'MA Sociology'  
  ELSE NULL  
END AS [Nom de section section]  
FROM student
```

last_name	first_name	Nom de section section
Lucas	Georges	MA Sociology
Eastwood	Clint	BSc Management
De Niro	Robert	NULL
Depp	Johnny	NULL
Portman	Natalie	BSc Management
Garcia	Andy	NULL
Willie	Bruce	BSc Management

Les fonctions : **NULLIF**

NULLIF (*colonne_considerée*, *valeur_à_mettre_à_NULL*)

La fonction « **NULLIF** » est un cas particulier du « **CASE** » qui renvoie les mêmes valeurs que la colonne passée en paramètre, sauf pour les valeurs équivalentes au deuxième paramètre fourni, pour lesquelles la valeur **NULL** sera affichée

```
CASE colonne_considerée  
  WHEN valeur_à_mettre_à_NULL THEN NULL  
  ELSE valeur_colonne_considerée  
END
```

Les fonctions : NULLIF

```
SELECT last_name, first_name, year_result  
      , NULLIF (year_result, 7)  
FROM student
```

```
SELECT last_name, first_name, year_result  
      , CASE year_result  
          WHEN 7 THEN NULL  
          ELSE year_result  
        END AS [Résultats sauf les 4/20]  
FROM student
```

last_name	first_name	year_result	Résultats sauf les 7/20
Clooney	Georges	4	4
Garcia	Andy	19	19
Willis	Bruce	6	6
Cruise	Tom	4	4
Witherspoon	Reese	7	NULL
Marceau	Sophie	6	6
Michelle Gellar	Sarah	7	NULL
Milano	Alyssa	7	NULL
Cameron	Jennifer	18	18

Les fonctions : **COALESCE**

COALESCE (*colonne1, colonne2, ..., colonneN*)

La fonction « **COALESCE** » est un autre cas particulier du « **CASE** » qui renvoie la première valeur non NULL rencontrée parmi les différentes colonnes fournies en paramètres

CASE

WHEN *colonne1 IS NOT NULL THEN colonne1*

WHEN *colonne2 IS NOT NULL THEN colonne2*

...

WHEN *colonneN-1 IS NOT NULL THEN colonneN-1*

ELSE *colonneN*

END

Les fonctions : COALESCE

Table
« WAGES »

emp_id	hourly_wage	salary	commission	num_sales
1	10	NULL	NULL	NULL
2	20	NULL	NULL	NULL
3	30	NULL	NULL	NULL
4	40	NULL	NULL	NULL
5	NULL	10000	NULL	NULL
6	NULL	20000	NULL	NULL
7	NULL	30000	NULL	NULL
8	NULL	40000	NULL	NULL
9	NULL	NULL	15000	3
10	NULL	NULL	25000	2
11	NULL	NULL	20000	6
12	NULL	NULL	14000	4

Résultat
du COALESCE

Total Salary
10000,00
20000,00
20800,00
30000,00
40000,00
41600,00
45000,00
50000,00
56000,00
62400,00
83200,00
120000,00

```
SELECT CAST(COALESCE(hourly_wage * 40 * 52
                      , salary
                      , commission * num_sales)
           AS money) AS 'Total Salary'
FROM wages
ORDER BY 'Total Salary'
```


Les fonctions : Imbrication

Table
« BUDGETS »

dept	current_year	previous_year
1	100000	150000
2	NULL	300000
3	0	100000
4	NULL	150000
5	300000	250000

NULL = même budget que l'année précédente

0 = budget non défini (valeurs à ne pas considérer dans la moyenne)

« current_year » et « previous_year » sont de type DECIMAL



```
SELECT AVG(NULLIF(COALESCE(current_year, previous_year), 0.00)) AS 'Average Budget'  
FROM budgets
```

Average Budget

212500.000000

Les fonctions : Microsoft vs Oracle

T-SQL	Oracle	Utilité
AVG()	AVG()	Faire une moyenne
COUNT()	COUNT()	Compte le nombre d'enregistrement
MAX() & MIN()	MAX() & MIN()	Plus grande/petite valeur
SUM()	SUM()	Faire la somme
CAST() & CONVERT()	CAST(<donnée> AS <type>)	Conversion de données
DATEADD() & DATEDIFF()	+ & -	Additionner/soustraire des dates
COALESCE()	COALESCE()	Retourne la valeur reprise si elle est NON NULL
GETDATE()	CURRENT_DATE	Retourne la date et heure actuelle
DATEPART(<extraction> , <date>)	EXTRACT (<extraction> FROM <date>)	Retourne un entier Datepart, ex: yy,yyyy, mm...

Les fonctions : Microsoft vs Oracle

T-SQL	Oracle	Utilité
DAY(), MONTH(), YEAR()		Retourne le jour, le mois, l'année
NULLIF()	NULLIF()	Retourne NULL si <nom_colonne> est égale à <valeur_à_éliminer>
ABS()	ABS()	Valeur absolue
<dividende> % <diviseur>	MOD(<dividende> , <diviseur>)	Modulo
RAND()	dbms_random.random	Génère un nombre aléatoire
LEN()	LENGTH()	Nombre de caractères
LOWER() & UPPER()	LOWER() & UPPER()	Mettre en minuscule/majuscule
LTRIM(RTRIM())	TRIM	Supprime les espaces à gauche et à droite
CHARINDEX()	INSTR()	Retourne la position d'une chaîne de caractères dans une autre

Les fonctions : Microsoft vs Oracle

T-SQL	Oracle	Utilité
SUBSTRING()	SUBSTR ()	Extraire une chaîne
+ ou CONCAT(exp1, exp2)		Concaténation

Auto-Evaluation

N'oubliez pas de prendre le temps d'évaluer le niveau de maîtrise que vous estimez avoir acquis personnellement concernant les notions abordées dans ce module !

Rappel de la signification des lettres dans les tableaux d'auto-évaluation :

- **Parfait (P)** : vous avez parfaitement compris cette notion et vous vous sentez à votre aise
- **Satisfaisant (S)** : vous avez compris de quoi il s'agit mais la pratique vous manque
- **Vague (V)** : vous savez de quoi il s'agit, mais cela reste un peu vague dans votre esprit.
Une explication supplémentaire du formateur ou une bonne révision de votre part s'impose
- **Insatisfaisant (I)** : Vous n'avez pas du tout compris la notion abordée, il faut tout faire pour y remédier !

Auto-Evaluation

Notions à évaluer

Notions	P	S	V	I
Fonction (fonctionnement interne, utilité, mise en pratique)				
Imbrication de fonctions				
Fonctions d'agrégation				
Expression « CASE »				
Fonctions « NULLIF » et « COALESCE »				