

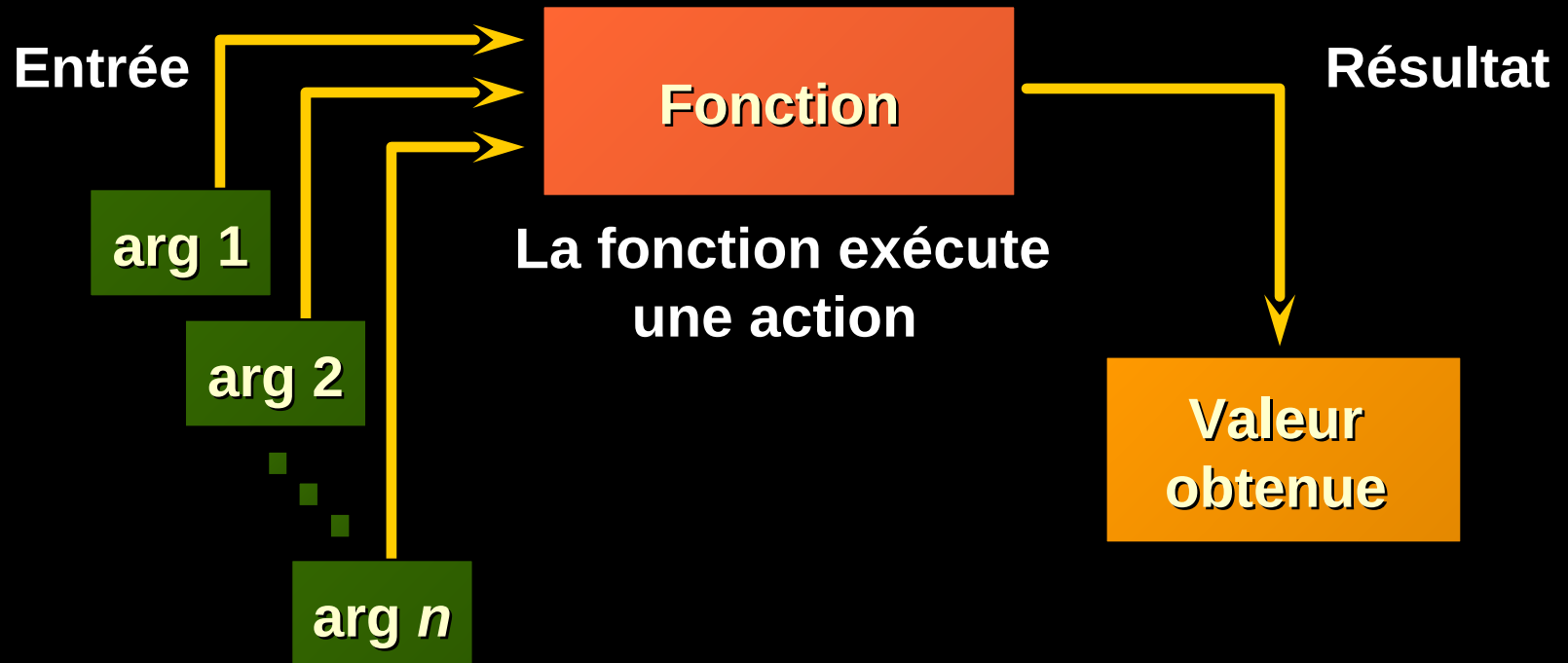
3 Fonctions monoligne

Objectifs

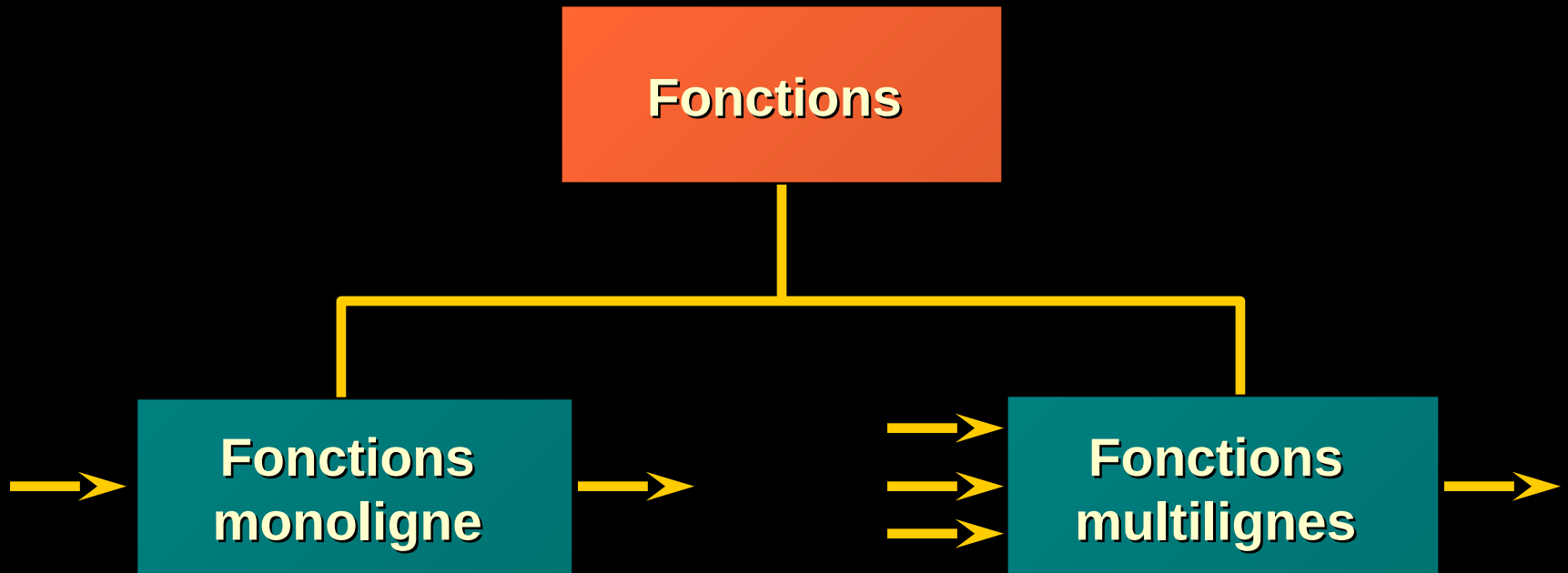
A la fin de ce chapitre, vous pourrez :

- **décrire différents types de fonction SQL**
- **utiliser des fonctions alphanumériques, numériques et de date dans les instructions SELECT**
- **décrire l'utilisation des fonctions de conversion**

Fonctions SQL



Deux types de fonction SQL



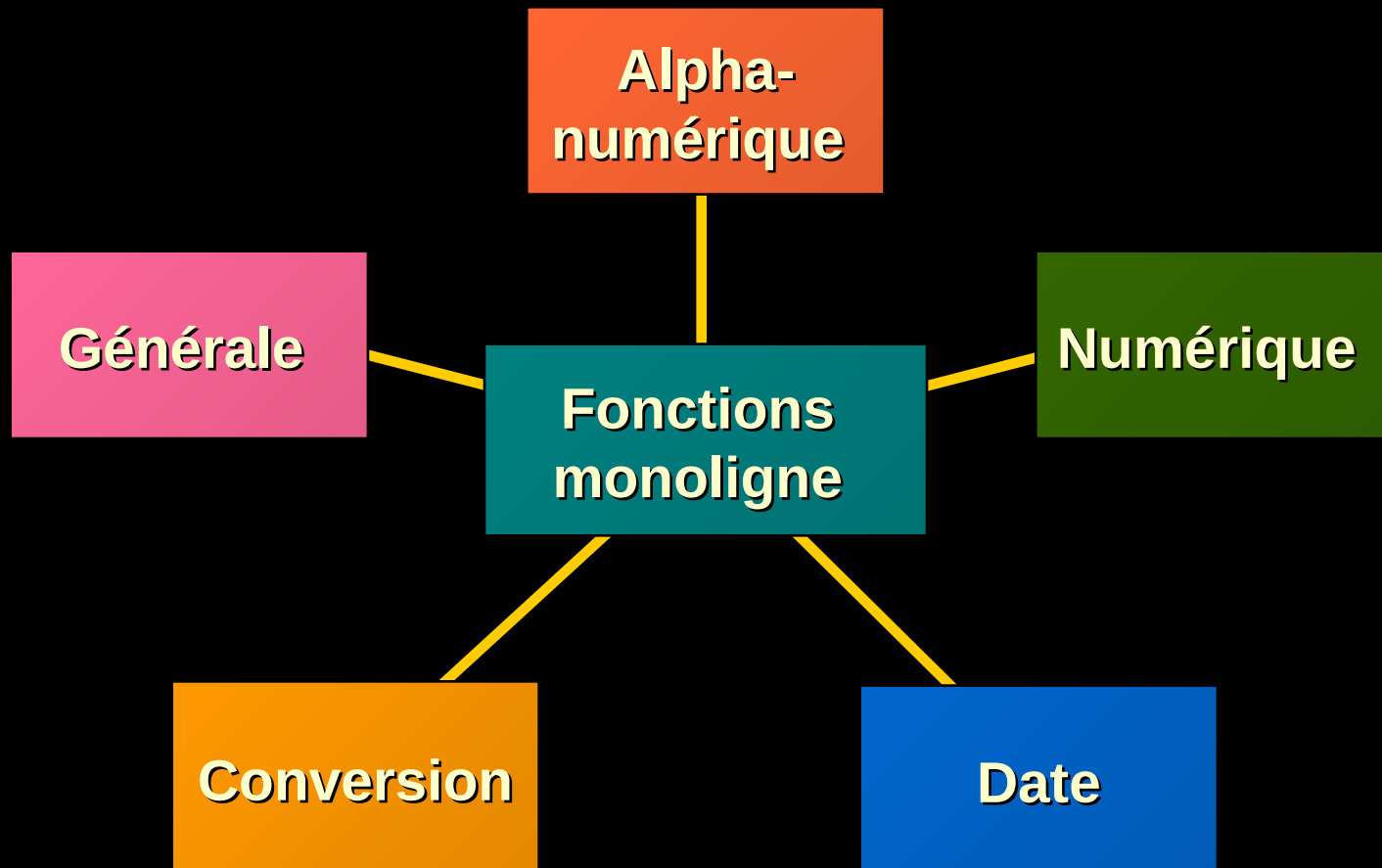
Fonctions monoligne

Les fonctions monoligne :

- manipulent des éléments de données,
- acceptent des arguments et renvoient une valeur,
- agissent sur chacune des lignes renvoyées,
- renvoient un résultat par ligne,
- peuvent modifier le type de données,
- peuvent être imbriquées,
- acceptent des arguments qui peuvent être une colonne ou une expression.

```
function_name [(arg1, arg2, ...)]
```

Fonctions monoligne



Fonctions alphanumériques

Fonctions alphanumériques

```
graph TD; A[Fonctions alphanumériques] --> B[Fonctions de manipulation des majuscules/minuscules]; A --> C[Fonctions de manipulation des caractères]; B --> B1[LOWER]; B --> B2[UPPER]; B --> B3[INITCAP]; C --> C1[CONCAT]; C --> C2[SUBSTR]; C --> C3[LENGTH]; C --> C4[INSTR]; C --> C5[LPAD | RPAD]; C --> C6[TRIM]; C --> C7[REPLACE];
```

Fonctions de manipulation des majuscules/minuscules

LOWER
UPPER
INITCAP

Fonctions de manipulation des caractères

CONCAT
SUBSTR
LENGTH
INSTR
LPAD | RPAD
TRIM
REPLACE

Fonctions de manipulation des majuscules/minuscules

Ces fonctions permettent de modifier la casse des caractères dans des chaînes.

Fonction	Résultat
<code>LOWER('SQL Course')</code>	sql course
<code>UPPER('SQL Course')</code>	SQL COURSE
<code>INITCAP('SQL Course')</code>	Sql Course

Utiliser les fonctions de manipulation des majuscules/minuscules

Affichez le numéro, le nom et le numéro de service de l'employé Higgins :

```
SELECT employee_id, last_name, department_id
FROM   employees
WHERE  last_name = 'higgins';
no rows selected
```

```
SELECT employee_id, last_name, department_id
FROM   employees
WHERE  LOWER(last_name) = 'higgins';
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
205	Higgins	110

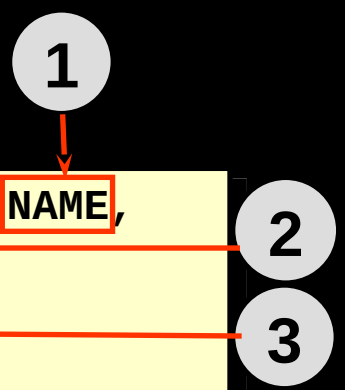
Fonctions de manipulation des caractères

Ces fonctions permettent de manipuler des chaînes de caractères :

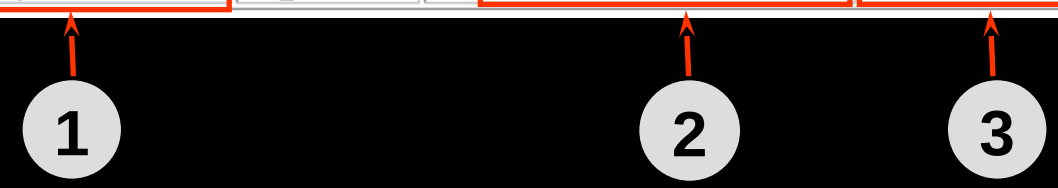
Fonction	Résultat
<code>CONCAT('Hello', 'World')</code>	HelloWorld
<code>SUBSTR('HelloWorld',1,5)</code>	Hello
<code>LENGTH('HelloWorld')</code>	10
<code>INSTR('HelloWorld', 'W')</code>	6
<code>LPAD(salary,10,'*')</code>	*****24000
<code>RPAD(salary, 10, '*')</code>	24000*****
<code>TRIM('H' FROM 'HelloWorld')</code>	elloWorld

Utiliser les fonctions de manipulation des caractères

```
SELECT employee_id, CONCAT(first_name, last_name) NAME,  
        job_id, LENGTH (last name),  
        INSTR(last_name, 'a') "Contains 'a'?"  
FROM employees  
WHERE SUBSTR(job_id, 4) = 'REP';
```



EMPLOYEE_ID	NAME	JOB_ID	LENGTH(LAST_NAME)	Contains 'a'?
174	EllenAbel	SA_REP	4	0
176	JonathonTaylor	SA_REP	6	2
178	KimberelyGrant	SA_REP	5	3
202	PatFay	MK_REP	3	2



Fonctions numériques

- **ROUND** : Arrondit la valeur à la décimale spécifiée

ROUND (45.926, 2) → 45.93

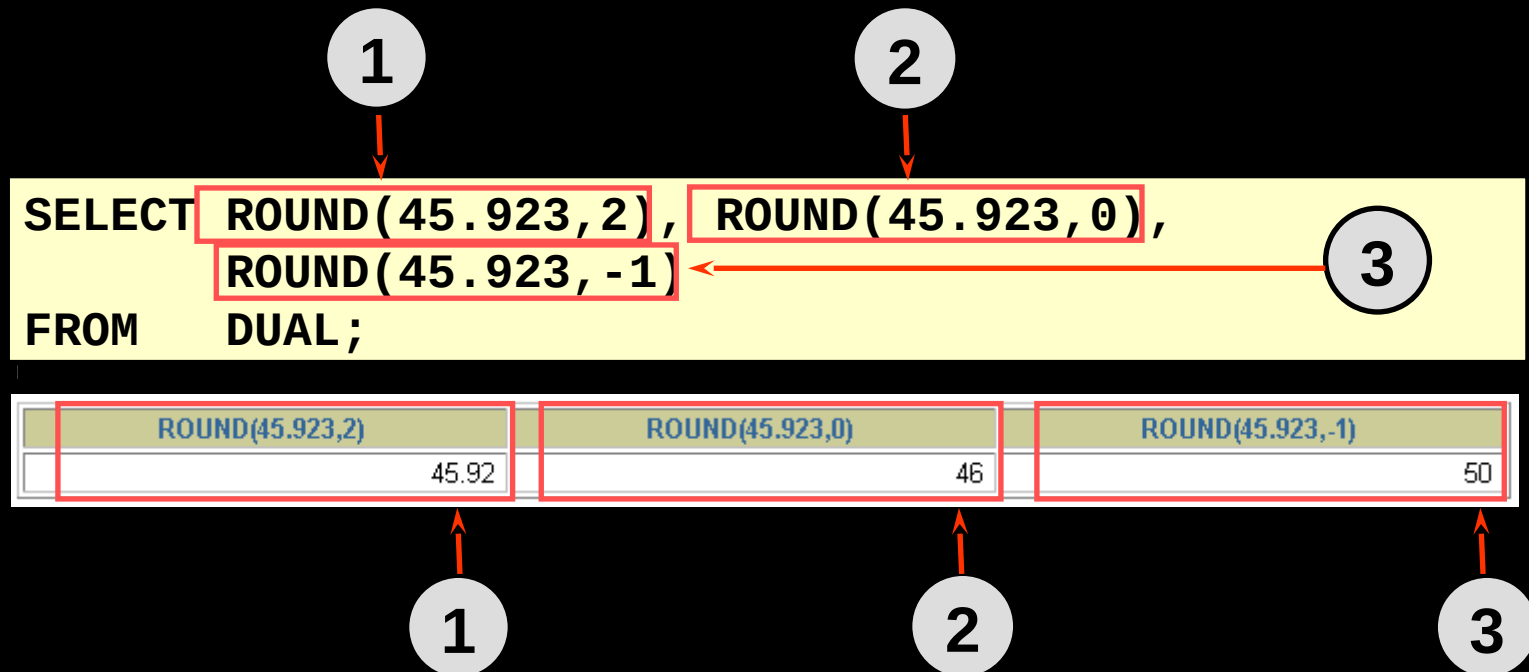
- **TRUNC** : Tronque la valeur à la décimale spécifiée

TRUNC (45.926, 2) → 45.92

- **MOD** : Renvoie le reste d'une division

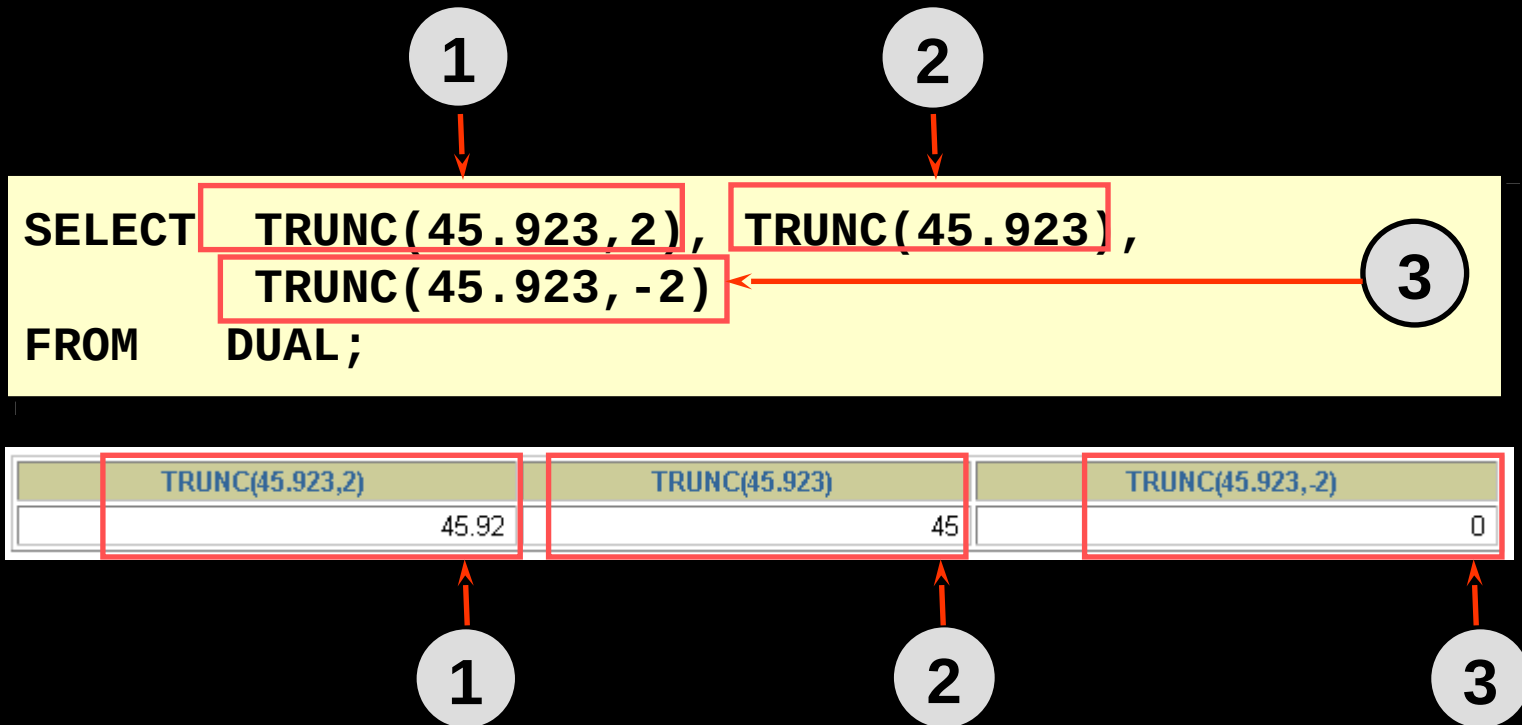
MOD (1600, 300) → 100

Utiliser la fonction ROUND



DUAL est une table factice dans laquelle vous pouvez visualiser les résultats des fonctions et des calculs.

Utiliser la fonction TRUNC



Utiliser la fonction MOD

Calculez le reste d'un salaire après division par 5000 pour tous les employés occupant le poste de représentant.

```
SELECT last_name, salary, MOD(salary, 5000)
FROM employees
WHERE job_id = 'SA_REP';
```

LAST_NAME	SALARY	MOD(SALARY,5000)
Abel	11000	1000
Taylor	8600	3600
Grant	7000	2000

Utiliser les dates

- Dans la base de données Oracle, les dates sont stockées sous un format numérique interne : siècle, année, mois, jour, heures, minutes, secondes.
- Le format de date par défaut est DD-MON-RR.
 - Vous pouvez ainsi enregistrer des dates du 21e siècle au 20e siècle en indiquant uniquement les deux derniers chiffres de l'année.
 - Inversement, vous pouvez enregistrer des dates du 20e siècle au 21e siècle.

```
SELECT last_name, hire_date
FROM employees
WHERE last_name like 'G%';
```

LAST_NAME	HIRE_DATE
Gietz	07-JUN-94
Grant	24-MAY-99

Utiliser les dates

La fonction **SYSDATE** renvoie :

- la date,
- l'heure.

Opérations arithmétiques sur les dates

- Ajout ou soustraction d'un nombre à une date pour obtenir un résultat de type date.
- Soustraction de deux dates afin de déterminer le nombre de jours entre ces deux dates.
- Ajout d'un nombre d'heures à une date en divisant le nombre d'heures par 24.

Utiliser des opérateurs arithmétiques avec les dates

```
SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS  
FROM employees  
WHERE department_id = 90;
```

LAST_NAME	WEEKS
King	744.245395
Kochhar	626.102538
De Haan	453.245395

Fonctions de date

Fonction	Description
MONTHS_BETWEEN	Nombre de mois entre deux dates
ADD_MONTHS	Ajout de mois calendaires à une date
NEXT_DAY	Jour suivant la date indiquée
LAST_DAY	Dernier jour du mois
ROUND	Arrondi d'une date
TRUNC	Troncature d'une date

Utiliser les fonctions de date

- **MONTHS_BETWEEN ('01-SEP-95' , '11-JAN-94')**
→ 19.6774194
- **ADD_MONTHS ('11-JAN-94' , 6)** → '11-JUL-94'
- **NEXT_DAY ('01-SEP-95' , 'FRIDAY')**
→ '08-SEP-95'
- **LAST_DAY('01-FEB-95')** → '28-FEB-95'

Utiliser les fonctions de date

Supposons que `SYSDATE` = '25-JUL-95':

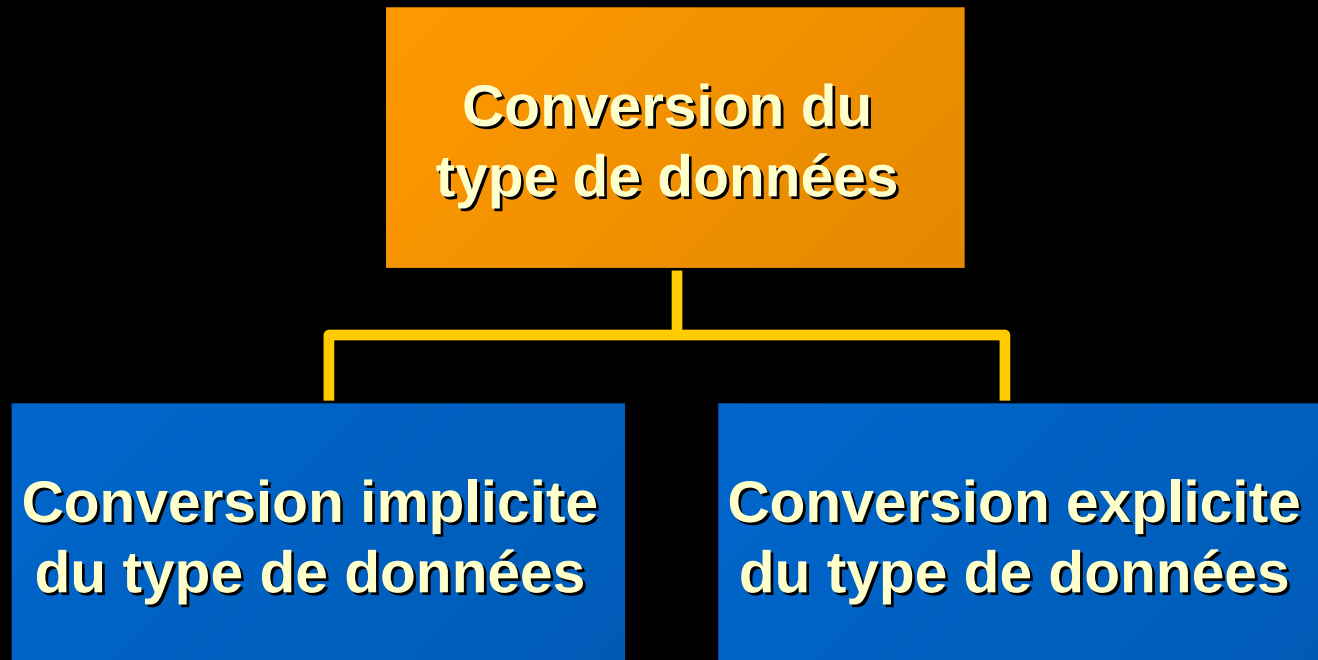
- `ROUND(SYSDATE, 'MONTH')` → 01-AUG-95
- `ROUND(SYSDATE, 'YEAR')` → 01-JAN-96
- `TRUNC(SYSDATE, 'MONTH')` → 01-JUL-95
- `TRUNC(SYSDATE, 'YEAR')` → 01-JAN-95

Présentation de l'exercice 3, 1^{ère} partie

Dans cet exercice, vous allez :

- **écrire une interrogation permettant d'afficher la date du jour**
- **créer des interrogations utilisant des fonctions numériques, alphanumériques et de date**
- **effectuer des calculs relatifs aux années et aux mois d'ancienneté d'un employé**

Fonctions de conversion



Conversion implicite des types de données

Pour les affectations, le serveur Oracle peut convertir automatiquement les types de données suivants :

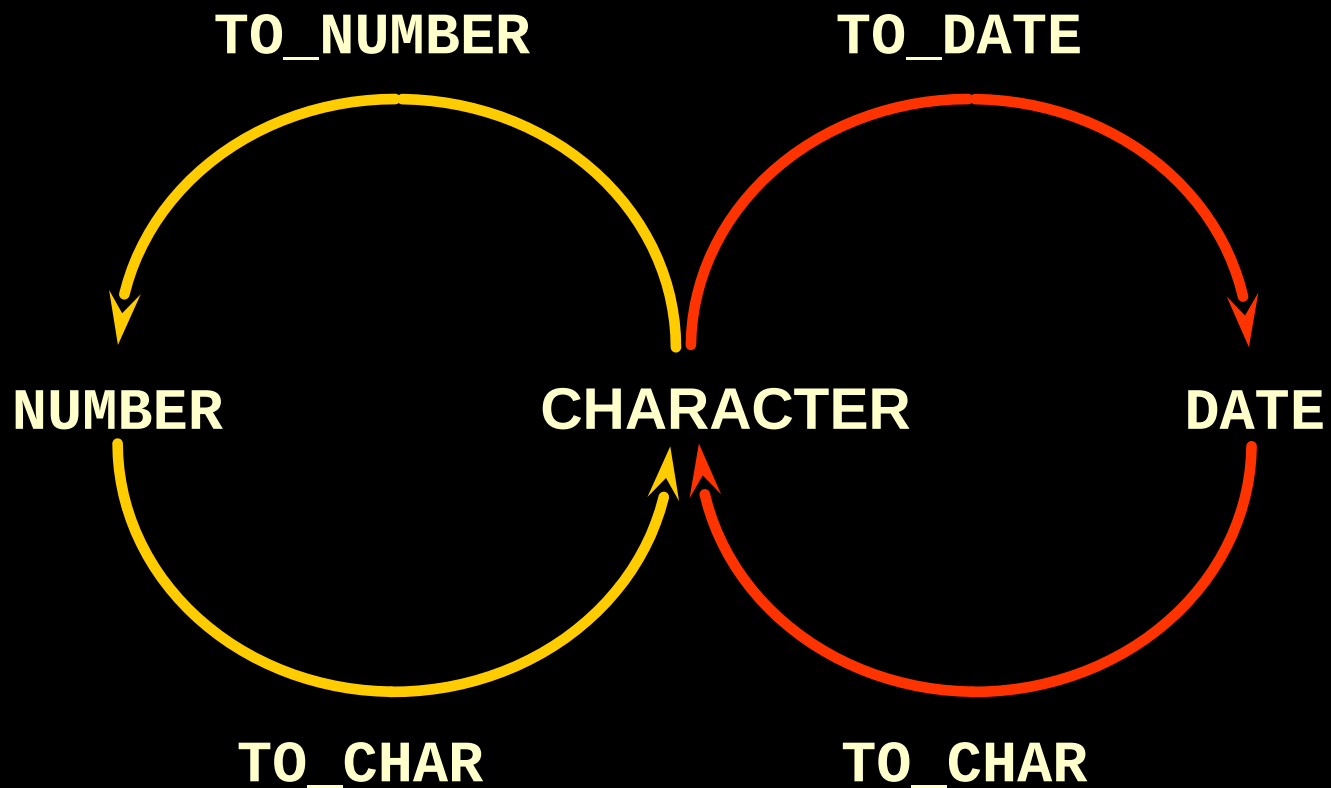
Type de données source	Type de données cible
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

Conversion implicite des types de données

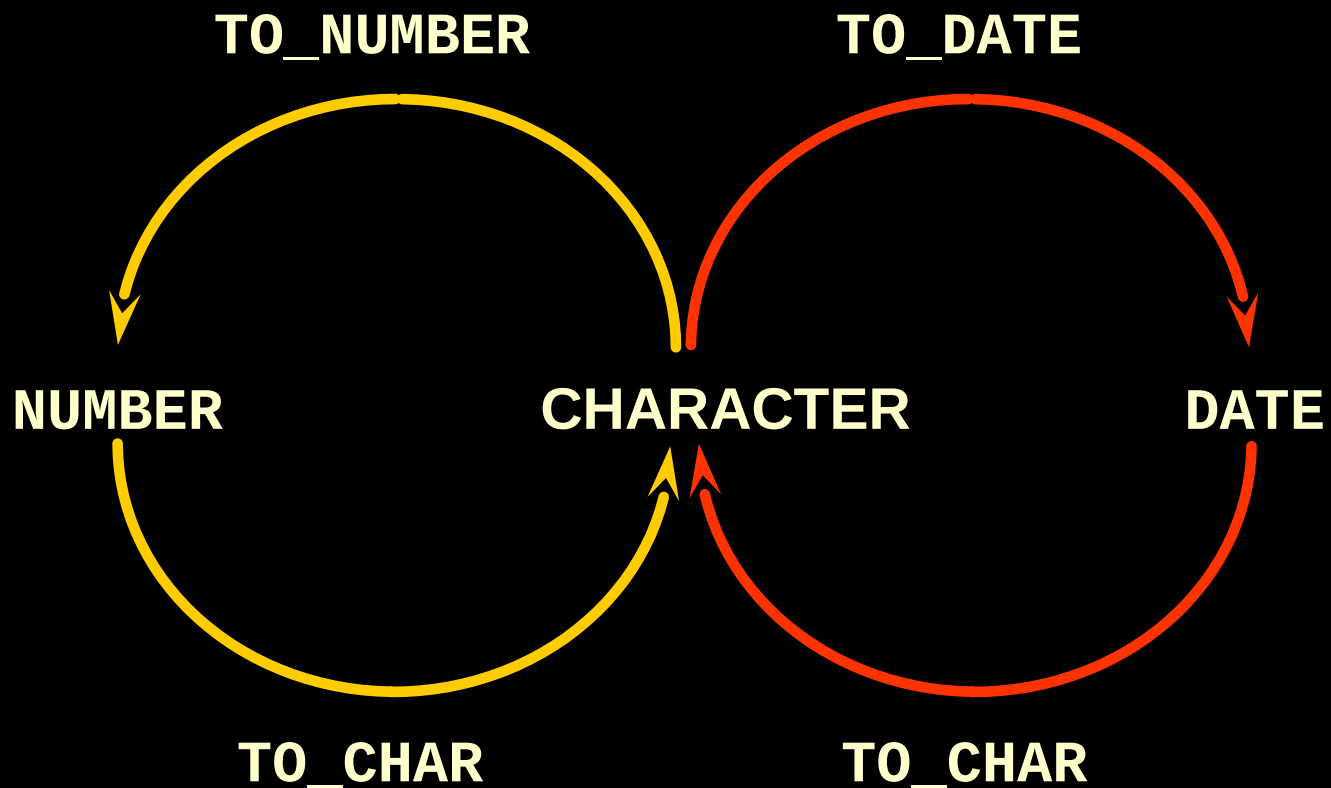
Pour l'évaluation d'expressions, le serveur Oracle peut convertir automatiquement les valeurs suivantes :

Type de données source	Type de données cible
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE

Conversion explicite des types de données



Conversion explicite des types de données



Utiliser la fonction TO_CHAR avec les dates

```
TO_CHAR(date, 'format_model') 
```

Le modèle de format :

- doit être placé entre apostrophes et différencie les majuscules et minuscules,
- peut inclure tout élément valide de format de date,
- comporte un élément *fm* qui supprime les espaces de remplissage ou les zéros de tête,
- est séparé de la valeur de date par une virgule.

Éléments du modèle de format de date

YYYY	4 chiffres de l'année
YEAR	Année exprimée en toutes lettres
MM	Mois représenté par 2 chiffres
MONTH	Mois exprimé en toutes lettres
MON	Mois abrégé en trois lettres
DY	Jour abrégé en trois lettres
DAY	Jour exprimé en toutes lettres
DD	Valeur numérique correspondant au jour du mois

Éléments du modèle de format de date

- Les éléments horaires permettent de formater la partie horaire de la date.

HH24:MI:SS AM	15:45:32 PM
---------------	-------------

- Vous pouvez ajouter des chaînes de caractères placées entre guillemets.

DD "of" MONTH	12 of OCTOBER
---------------	---------------

- Les suffixes numériques permettent d'indiquer les nombres en toutes lettres.

ddspth	fourteenth
--------	------------

Utiliser la fonction TO_CHAR avec les dates

```
SELECT last_name,  
       TO_CHAR(hire_date, 'fmDD Month YYYY')  
       AS HIREDATE  
FROM   employees;
```

LAST_NAME	HIREDATE
King	17 June 1987
Kochhar	21 September 1989
De Haan	13 January 1993
Hunold	3 January 1990
Ernst	21 May 1991
Lorentz	7 February 1999
Mourgos	16 November 1999

...

20 rows selected.

Utiliser la fonction TO_CHAR avec les nombres

TO_CHAR(*number*, '*format_model*')

Vous pouvez utiliser certains éléments de format avec la fonction TO_CHAR pour afficher une valeur numérique sous forme de caractère.

9	Représente un nombre
0	Force l'affichage d'un zéro
\$	Place un signe dollar flottant
L	Utilise le symbole monétaire local flottant
.	Affiche un séparateur décimal
,	Affiche un indicateur de milliers

Utiliser la fonction TO_CHAR avec les nombres

```
SELECT TO_CHAR(salary, '$99,999.00') SALARY  
FROM employees  
WHERE last_name = 'Ernst';
```

SALARY
\$6,000.00

Utiliser les fonctions TO_NUMBER et TO_DATE

- Convertissez une chaîne de caractères en un format numérique à l'aide de la fonction TO_NUMBER :

```
TO_NUMBER(char[, 'format_model'])
```

- Convertissez une chaîne de caractères en un format de date à l'aide de la fonction TO_DATE :

```
TO_DATE(char[, 'format_model'])
```

- Ces fonctions possèdent un modificateur fx qui indique la correspondance exacte de l'argument alphanumérique et du modèle de format de date d'une fonction TO_DATE.

Using the TO_NUMBER and TO_DATE Functions

- Convert a character string to a number format using the TO_NUMBER function:

```
TO_NUMBER(char[, 'format_model'])
```

- Convert a character string to a date format using the TO_DATE function:

```
TO_DATE(char[, 'format_model'])
```

- These functions have an fx modifier. This modifier specifies the exact matching for the character argument and date format model of a TO_DATE function

Format de date RR

Année en cours	Date indiquée	Format RR	Format YY
1995	27-OCT-95	1995	1995
1995	27-OCT-17	2017	1917
2001	27-OCT-17	2017	2017
2001	27-OCT-95	1995	2095

		Si l'année à deux chiffres indiquée est la suivante :	
		0–49	50–99
Si l'année en cours comporte deux chiffres parmi les suivants :	0–49	La date renvoyée appartient au siècle en cours	La date renvoyée appartient au siècle qui précède le siècle en cours
	50–99	La date renvoyée appartient au siècle qui suit le siècle en cours	La date renvoyée appartient au siècle en cours

Exemple de format de date RR

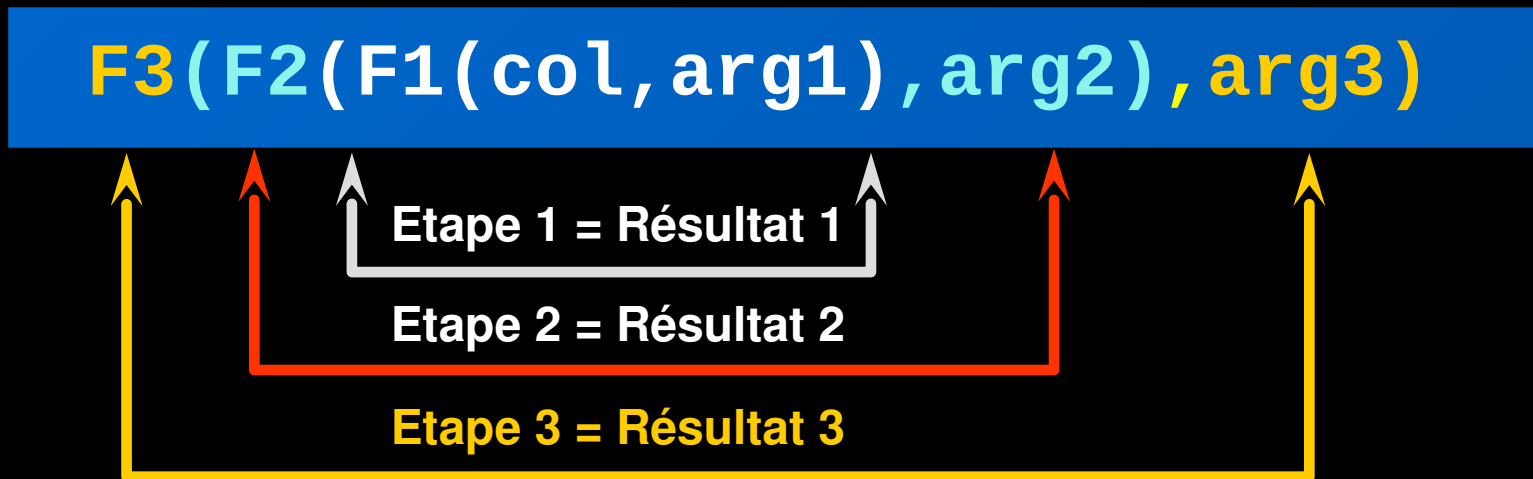
Pour rechercher des employés embauchés avant 1990, utilisez le format RR, qui produit les mêmes résultats que la commande soit exécutée en 1999 ou à présent :

```
SELECT last_name, TO_CHAR(hire_date, 'DD-Mon-YYYY')  
FROM employees  
WHERE hire_date < TO_DATE('01-Jan-90', 'DD-Mon-RR');
```

LAST_NAME	TO_CHAR(HIR
King	17-Jun-1987
Kochhar	21-Sep-1989
Whalen	17-Sep-1987

Imbriquer des fonctions

- Les fonctions monoligne peuvent être imbriquées à tous les niveaux.
- Les fonctions imbriquées sont évaluées de l'intérieur vers l'extérieur.



Imbriquer des fonctions

```
SELECT last_name,  
       NVL(TO_CHAR(manager_id), 'No Manager')  
FROM   employees  
WHERE  manager_id IS NULL;
```

LAST_NAME	NVL(TO_CHAR(MANAGER_ID), 'NOMANAGER')
King	No Manager

Fonctions générales

Ces fonctions, qui se rapportent à l'utilisation de valeurs NULL, s'utilisent avec tous les types de données.

- NVL (expr1, expr2)
- NVL2 (expr1, expr2, expr3)
- NULLIF (expr1, expr2)
- COALESCE (expr1, expr2, ..., exprn)

Fonction NVL

Cette fonction convertit une valeur NULL en valeur réelle.

- **Les types de données admis sont DATE, CHARACTER et NUMBER. .**
- **Les types de données doivent correspondre :**
 - **NVL(commission_pct,0)**
 - **NVL(hire_date, '01-JAN-97')**
 - **NVL(job_id, 'No Job Yet')**

Utiliser la fonction NVL

```
SELECT last_name, salary, NVL(commission_pct, 0),  
       (salary*12) + (salary*12*NVL(commission_pct, 0)) AN_SAL  
FROM employees;
```

LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
King	24000	0	288000
Kochhar	17000	0	204000
De Haan	17000	0	204000
Hunold	9000	0	108000
Ernst	6000	0	72000
Lorentz	4200	0	50400
Mourgos	5800	0	69600
Rajs	3500	0	42000

...

20 rows selected.

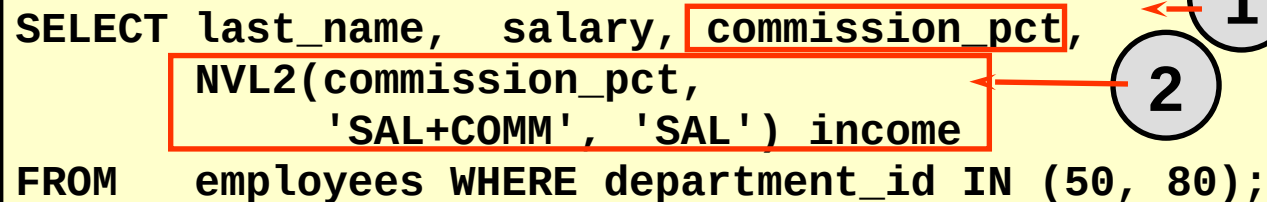
1

2

ORACLE

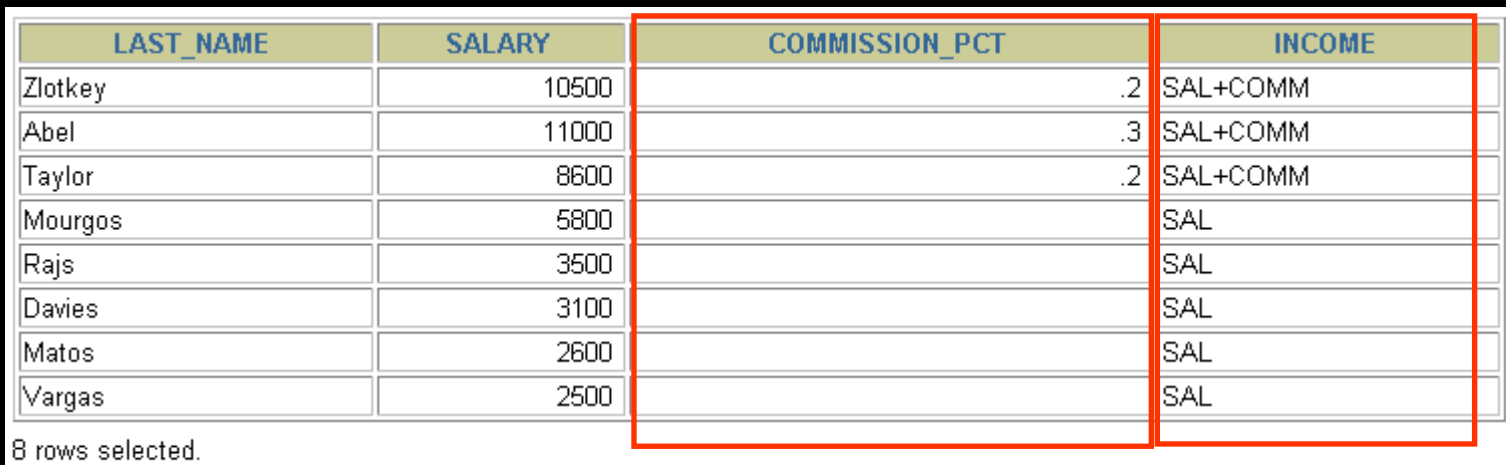
Utiliser la fonction NVL2

```
SELECT last_name, salary, commission_pct,  
       NVL2(commission_pct,  
            'SAL+COMM', 'SAL') income  
FROM   employees WHERE department_id IN (50, 80);
```



LAST_NAME	SALARY	COMMISSION_PCT	INCOME
Zlotkey	10500	.2	SAL+COMM
Abel	11000	.3	SAL+COMM
Taylor	8600	.2	SAL+COMM
Mourgos	5800		SAL
Rajs	3500		SAL
Davies	3100		SAL
Matos	2600		SAL
Vargas	2500		SAL

8 rows selected.



1

2

Utiliser la fonction NULLIF

1

```
SELECT first_name, LENGTH(first_name) "expr1",  
       last_name,  LENGTH(last_name) "expr2",  
       NULLIF(LENGTH(first_name), LENGTH(last_name)) result  
FROM   employees;
```

2

3

FIRST_NAME	expr1	LAST_NAME	expr2	RESULT
Steven	6	King	4	6
Neena	5	Kochhar	7	5
Lex	3	De Haan	7	3
Alexander	9	Hunold	6	9
Bruce	5	Ernst	5	
Diana	5	Lorentz	7	5
Kevin	5	Mourgos	7	5
Trenna	6	Rajs	4	6
Curtis	6	Davies	6	

...

20 rows selected.

1

2

3

Utiliser la fonction COALESCE

- L'avantage de la fonction COALESCE par rapport à la fonction NVL réside dans le fait qu'elle peut accepter plusieurs valeurs différentes.
- La fonction renvoie la première expression si celle-ci n'est pas NULL. Sinon, elle fusionne les expressions restantes.

Utiliser la fonction COALESCE

```
SELECT    last_name,  
          COALESCE(commission_pct, salary, 10) comm  
FROM      employees  
ORDER BY  commission_pct;
```

LAST_NAME	COMM
Grant	.15
Zlotkey	.2
Taylor	.2
Abel	.3
King	24000
Kochhar	17000
De Haan	17000
Hunold	9000

■ ■ ■

20 rows selected.

Expressions conditionnelles

- Ces expressions permettent d'utiliser la logique IF-THEN-ELSE dans une instruction SQL.
- Elles utilisent deux méthodes :
 - Expression CASE
 - Fonction DECODE

Expression CASE

Cette expression, qui fonctionne comme une instruction IF-THEN-ELSE, facilite les interrogations conditionnelles :

```
CASE expr WHEN comparison_expr1 THEN return_expr1  
      [WHEN comparison_expr2 THEN return_expr2  
      WHEN comparison_exprn THEN return_exprn  
      ELSE else_expr]  
END
```

Utiliser l'expression CASE

Cette expression, qui fonctionne comme une instruction IF-THEN-ELSE, facilite les interrogations conditionnelles :

```
SELECT last_name, job_id, salary,  
       CASE job_id WHEN 'IT_PROG' THEN 1.10*salary  
                  WHEN 'ST_CLERK' THEN 1.15*salary  
                  WHEN 'SA_REP' THEN 1.20*salary  
       ELSE salary END "REVISED_SALARY"  
FROM employees;
```

LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

Fonction DECODE

Cette fonction, qui fonctionne comme une instruction CASE ou IF-THEN-ELSE, facilite les interrogations conditionnelles :

```
DECODE(col|expression, search1, result1  
      [, search2, result2, ..., ]  
      [, default])
```

Utiliser la fonction DECODE

```
SELECT last name, job id, salary,  
       DECODE(job_id, 'IT_PROG', 1.10*salary,  
                'ST_CLERK', 1.15*salary,  
                'SA_REP', 1.20*salary,  
                salary)  
       REVISED_SALARY  
FROM   employees;
```

LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

Utiliser la fonction DECODE

Affichez le taux d'imposition applicable à chaque employé du service 80.

```
SELECT last name, salary,  
       DECODE (TRUNC(salary/2000, 0),  
               0, 0.00,  
               1, 0.09,  
               2, 0.20,  
               3, 0.30,  
               4, 0.40,  
               5, 0.42,  
               6, 0.44,  
               0.45) TAX_RATE  
FROM   employees  
WHERE  department_id = 80;
```

Synthèse

Ce chapitre vous à permis d'apprendre à :

- **effectuer des calculs sur des données à l'aide de fonctions**
- **modifier des éléments de données individuels à l'aide de fonctions**
- **manipuler la sortie de groupes de lignes à l'aide de fonctions**
- **modifier des formats de date pour l'affichage à l'aide de fonctions**
- **convertir les types de données de colonnes à l'aide de fonctions**
- **utiliser des fonctions NVL**
- **utiliser la logique IF-THEN-ELSE**

Présentation de l'exercice 3, 2^{ème} partie

Dans cet exercice, vous allez :

- **créer des interrogations utilisant des fonctions numériques, alphanumériques et de date**
- **utiliser la concaténation avec des fonctions**
- **écrire des interrogations qui ne différencient pas les majuscules et les minuscules pour tester l'utilité des fonctions alphanumériques**
- **effectuer des calculs relatifs aux années et aux mois d'ancienneté d'un employé**
- **déterminer la date de révision du salaire d'un employé**

