

4

Uso de Funciones de Conversión y Expresiones Condicionales

Objetivos

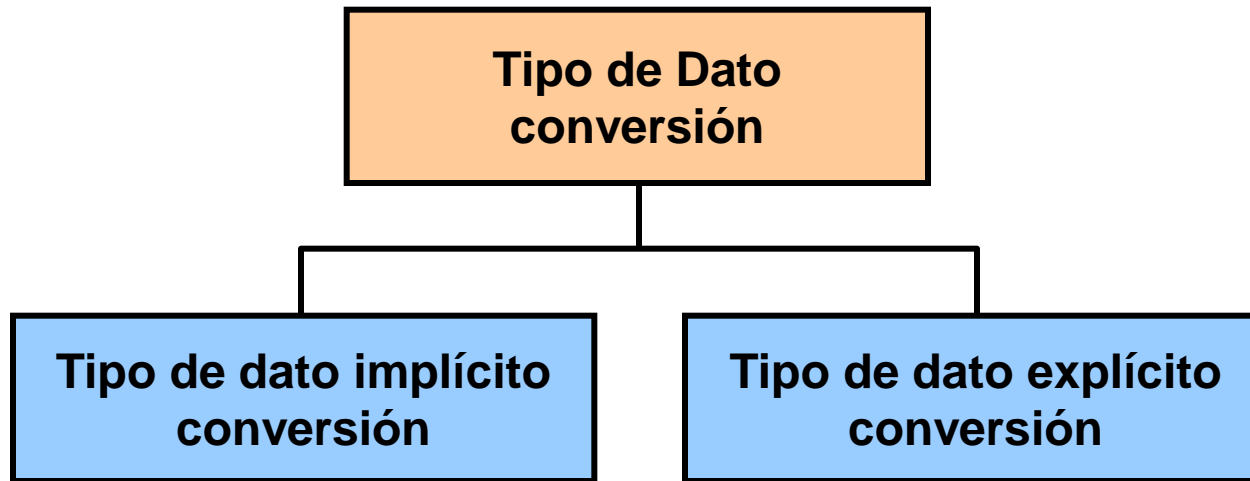
Al finalizar esta lección, debería estar capacitado para lo siguiente:

- Describir varios tipos de funciones de conversión que están disponibles en SQL
- Utilizar las funciones de conversión `TO_CHAR`, `TO_NUMBER` y `TO_DATE`
- Aplicar expresiones condicionales en una sentencia `SELECT`

Agenda

- Conversión de tipo de dato implícito y explícito
- Funciones `TO_CHAR`, `TO_DATE`, `TO_NUMBER`
- Funciones de anidación
- Funciones generales:
 - `NVL`
 - `NVL2`
 - `NULLIF`
 - `COALESCE`
- Expresiones condicionales:
 - `CASE`
 - `DECODE`

Funciones de Conversión



Conversión Implícita del Tipo de Dato

En expresiones, el servidor de Oracle puede convertir automáticamente:

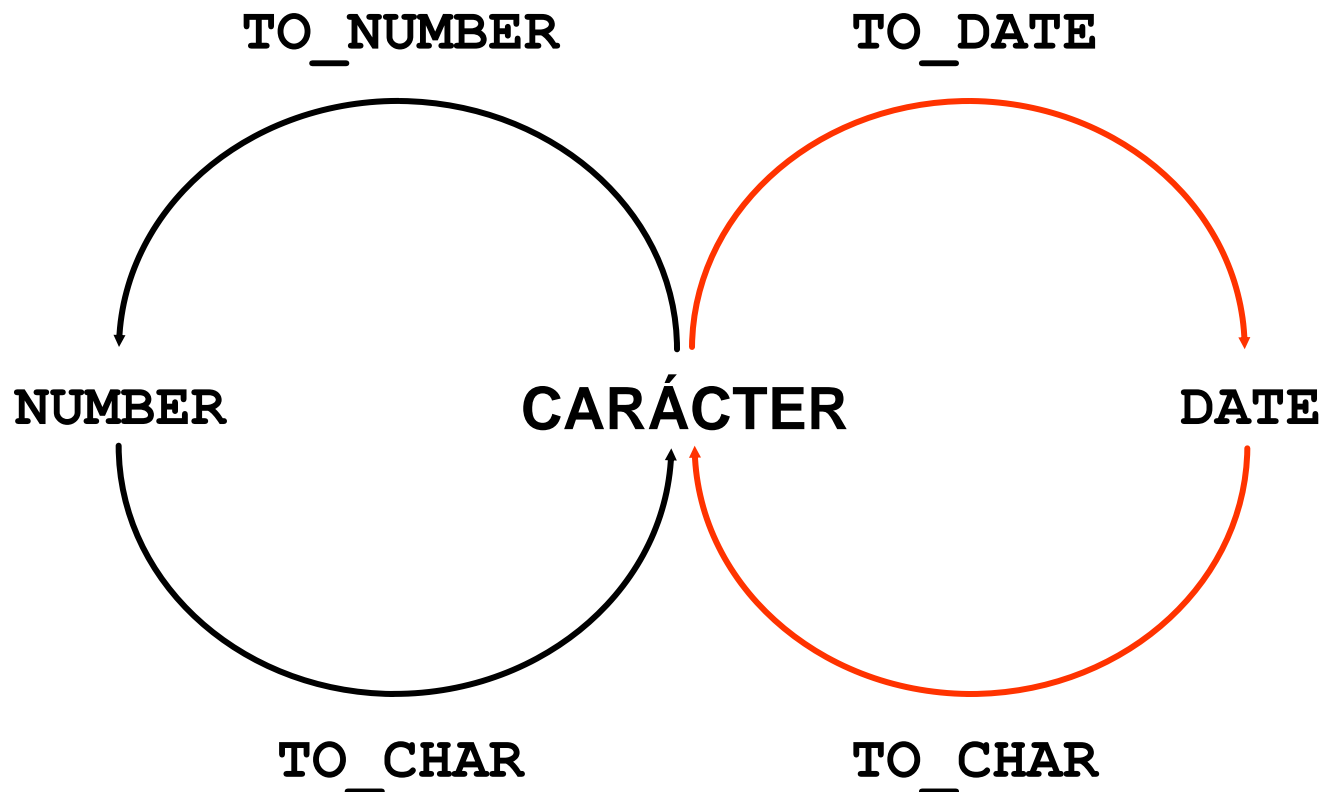
A	De
VARCHAR2 o CHAR	NUMBER
VARCHAR2 o CHAR	DATE

Conversión Implícita del Tipo de Dato

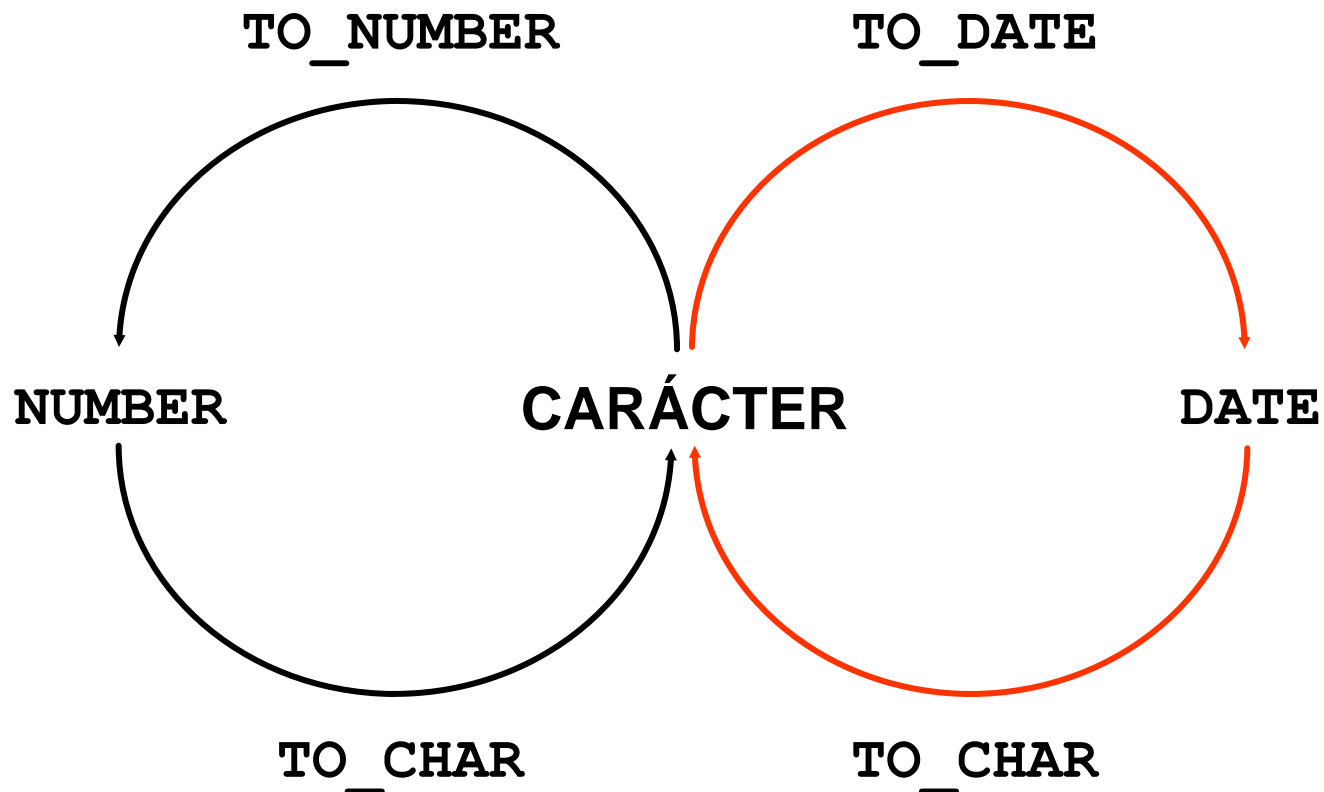
Para la evaluación de expresiones, el servidor de Oracle puede convertir automáticamente:

A	De
NUMBER	VARCHAR2 o CHAR
DATE	VARCHAR2 o CHAR

Conversión Explícita del Tipo de Dato



Conversión Explícita del Tipo de Dato



Agenda

- Conversión de tipo de dato implícito y explícito
- **Funciones** TO_CHAR, TO_DATE, TO_NUMBER
- Funciones de anidación
- Funciones generales:
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Expresiones condicionales:
 - CASE
 - DECODE

Uso de la Función TO_CHAR con Fechas

```
TO_CHAR(date, 'format_model')  

```

El modelo de formato:

- Debe estar entre comillas simples
- Es sensible a mayúsculas/minúsculas
- Puede incluir cualquier elemento de formato de fecha válido
- Tiene un elemento `fm` para eliminar los espacios en blanco o suprimir ceros iniciales
- Está separado del valor de fecha por una coma

Elementos del Modelo de Formato de Fecha

Elemento	Resultado
YYYY	Año completo en números
YEAR	Año en letra (en inglés)
MM	Valor de dos dígitos del mes
MONTH	Nombre completo del mes
MON	Abreviatura de tres letras del mes
DY	Abreviatura de tres letras del día de la semana
DAY	Nombre completo del día de la semana
DD	Día numérico del mes

Elementos del Modelo de Formato de Fecha

- Los elementos de tiempo formatean la parte de la hora de la fecha:

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

- Agregan cadenas de caracteres entre comillas dobles:

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

- El número se agrega como sufijo de los números en letra:

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

Uso de la Función TO_CHAR con Fechas

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

	LAST_NAME	HIREDATE
1	Whalen	17 September 1987
2	Hartstein	17 February 1996
3	Fay	17 August 1997
4	Higgins	7 June 1994
5	Gietz	7 June 1994
6	King	17 June 1987
7	Kochhar	21 September 1989
8	De Haan	13 January 1993
9	Hunold	3 January 1990
10	Ernst	21 May 1991

...

Uso de la Función TO_CHAR con Números


```
TO_CHAR(number, 'format_model') 
```

Éstos son algunos de los elementos de formato que puede utilizar con la función TO_CHAR para mostrar un valor de número como un carácter:

Elemento	Resultado
9	Representa un número
0	Fuerza para que aparezca un cero
\$	Coloca un signo de dólar flotante
L	Utiliza el símbolo de divisa local flotante
.	Imprime un punto decimal
,	Imprime una coma como indicador de miles

Uso de la Función TO_CHAR con Números

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

	 SALARY
1	\$6,000.00

Uso de Funciones TO_NUMBER y TO_DATE

- Convertir una cadena de caracteres a un formato de número que utiliza la función TO_NUMBER:

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

- Convertir una cadena de caracteres a un formato de fecha que utiliza la función TO_DATE:

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

- Estas funciones tienen un modificador \mathbb{f}_x . Este modificador especifica la coincidencia exacta para el argumento de carácter y el modelo de formato de fecha de una función TO_DATE.

Uso de las Funciones TO_CHAR y TO_DATE con el Formato de Fecha RR

Para buscar los empleados contratados antes de 1990, utilice el formato de fecha RR, que produce los mismos resultados si se ejecutara el comando en 1999 o en la actualidad:

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

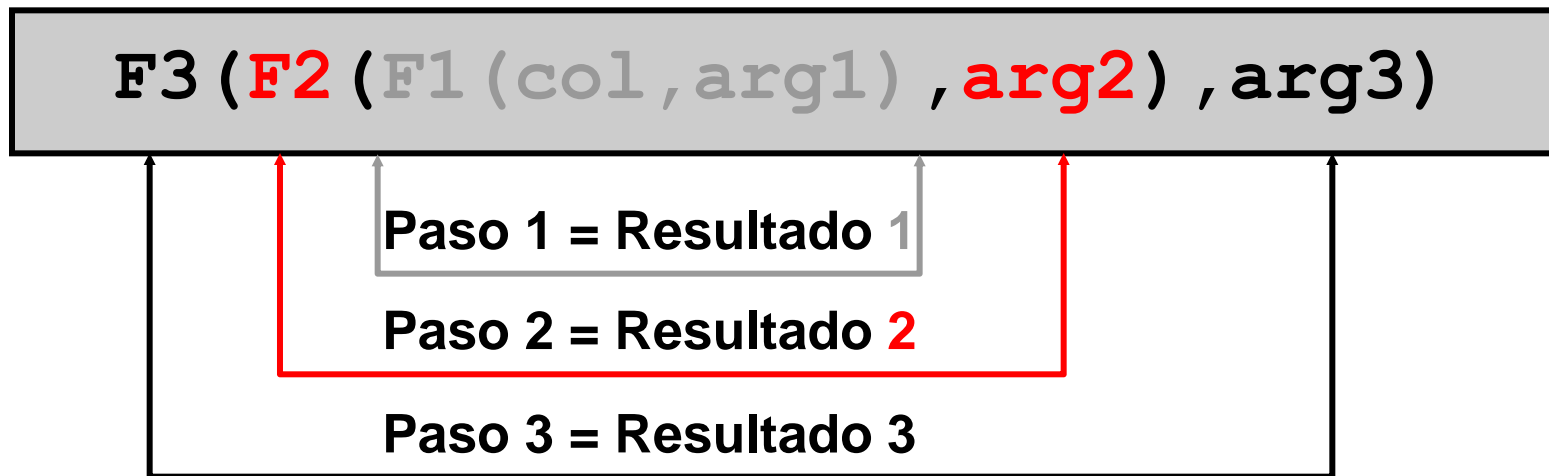
	LAST_NAME	TO_CHAR(HIRE_DATE,'DD-MON-YYYY')
1	Whalen	17-Sep-1987
2	King	17-Jun-1987
3	Kochhar	21-Sep-1989

Agenda

- Conversión de tipo de dato implícito y explícito
- Funciones TO_CHAR, TO_DATE, TO_NUMBER
- **Funciones de anidación**
- Funciones generales:
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Expresiones condicionales:
 - CASE
 - DECODE

Funciones de Anidación

- Las funciones de una sola fila se pueden anidar en cualquier nivel.
- Las funciones anidadas se evalúan desde el nivel más profundo hasta el nivel menos profundo.



Funciones de Anidación: Ejemplo 1

```
SELECT last_name,  
       UPPER(CONCAT(SUBSTR (LAST_NAME, 1, 8), '_US'))  
FROM   employees  
WHERE  department_id = 60;
```

	LAST_NAME	UPPER(CONCAT(SUBSTR(LAST_NAME,1,8),'_US'))
1	Hunold	HUNOLD_US
2	Ernst	ERNST_US
3	Lorentz	LORENTZ_US

Funciones de Anidación: Ejemplo 2

```
SELECT      TO_CHAR(ROUND((salary/7), 2), '99G999D99',  
              'NLS_NUMERIC_CHARACTERS = ','.' )  
            "Formatted Salary"  
FROM employees;
```

	Formatted Salary
1	628,57
2	1.857,14
3	857,14
4	1.714,29
5	1.185,71
6	3.428,57

...

Agenda

- Conversión de tipo de dato implícito y explícito
- Funciones TO_CHAR, TO_DATE, TO_NUMBER
- Funciones de anidación
- **Funciones generales:**
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Expresiones condicionales:
 - CASE
 - DECODE

Funciones Generales

Las siguientes funciones funcionan con cualquier tipo de dato y pertenecen al uso de valores nulos:

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

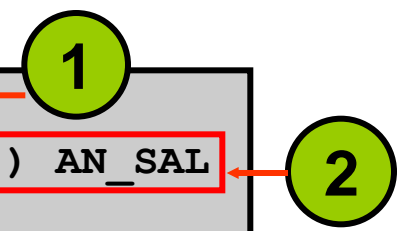
Función NVL

Convierte un valor nulo a un valor real:

- Los tipos de dato que se pueden utilizar son fecha, carácter y número.
- Los tipos de dato deben coincidir con:
 - `NVL(commission_pct, 0)`
 - `NVL(hire_date, '01-JAN-97')`
 - `NVL(job_id, 'No Job Yet')`

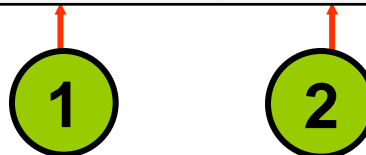
Uso de la Función NVL

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



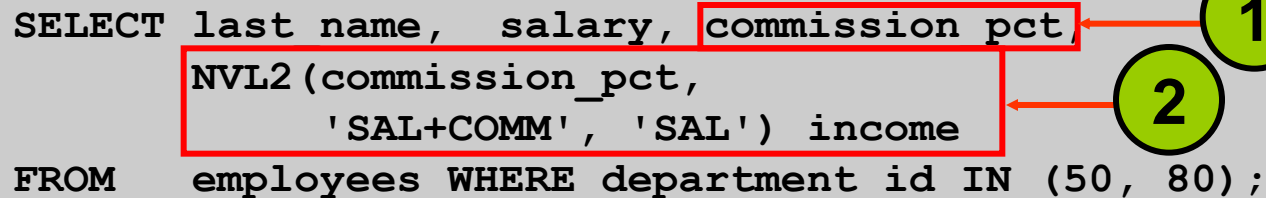
	LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
1	Whalen	4400	0	52800
2	Hartstein	13000	0	156000
3	Fay	6000	0	72000
4	Higgins	12000	0	144000
5	Gietz	8300	0	99600
6	King	24000	0	288000
7	Kochhar	17000	0	204000
8	De Haan	17000	0	204000
9	Hunold	9000	0	108000
10	Ernst	6000	0	72000

...

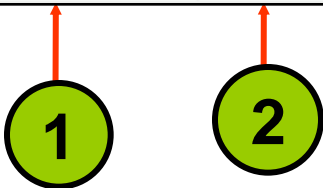


Uso de la Función NVL2

```
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
1	Mourgos	5800	(null)	SAL
2	Rajs	3500	(null)	SAL
3	Davies	3100	(null)	SAL
4	Matos	2600	(null)	SAL
5	Vargas	2500	(null)	SAL
6	Zlotkey	10500	0.2	SAL+COMM
7	Abel	11000	0.3	SAL+COMM
8	Taylor	8600	0.2	SAL+COMM



Uso de la Función 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

	1	FIRST_NAME	2	expr1	3	LAST_NAME	4	expr2	5	RESULT
1		Ellen		5		Abel		4		5
2		Curtis		6		Davies		6		(null)
3		Lex		3		De Haan		7		3
4		Bruce		5		Ernst		5		(null)
5		Pat		3		Fay		3		(null)
6		William		7		Gietz		5		7
7		Kimberely		9		Grant		5		9
8		Michael		7		Hartstein		9		7
9		Shelley		7		Higgins		7		(null)

...

1

2

3

Uso de la Función COALESCE

- La ventaja de la función COALESCE con respecto a la función NVL es que la función COALESCE puede obtener múltiples valores alternativos.
- Si la primera expresión no es nula, la función COALESCE devuelve esa expresión; de lo contrario, aplica la función COALESCE de las expresiones restantes.

Uso de la Función COALESCE

```
SELECT last name, employee id,  
COALESCE(TO_CHAR(commission_pct),TO_CHAR(manager_id) ,  
          'No commission and no manager')  
FROM employees;
```

	LAST_NAME	EMPLOYEE_ID	COALESCE(TO_CHAR(COMMISSION...
1	Whalen	200	101
2	Hartstein	201	100
3	Fay	202	201
4	Higgins	205	101
5	Gietz	206	205
6	King	100	No commission and no manager

...

17	Zlotkey	149	.2
18	Abel	174	.3
19	Taylor	176	.2
20	Grant	178	.15

Agenda

- Conversión de tipo de dato implícito y explícito
- Funciones `TO_CHAR`, `TO_DATE`, `TO_NUMBER`
- Funciones de anidación
- Funciones generales:
 - `NVL`
 - `NVL2`
 - `NULLIF`
 - `COALESCE`
- Expresiones condicionales:
 - `CASE`
 - `DECODE`

Expresiones Condicionales

- Proporcionar el uso de la lógica `IF-THEN-ELSE` en una sentencia SQL
- Utilizar dos métodos:
 - Expresión `CASE`
 - Función `DECODE`

Expresión CASE

Facilita las consultas condicionales realizando el trabajo de una sentencia IF-THEN-ELSE:

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


Uso de la Expresión CASE

Facilita las consultas condicionales realizando el trabajo de una sentencia IF-THEN-ELSE:

```
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
1	Whalen	AD_ASST	4400	4400
...				
9	Hunold	IT_PROG	9000	9900
10	Ernst	IT_PROG	6000	6600
11	Lorentz	IT_PROG	4200	4620
12	Mourgos	ST_MAN	5800	5800
13	Rajs	ST_CLERK	3500	4025
14	Davies	ST_CLERK	3100	3565
...				
19	Taylor	SA_REP	8600	10320
20	Grant	SA_REP	7000	8400

Función DECODE

Facilita las consultas condicionales realizando el trabajo de una expresión CASE o una sentencia IF-THEN-ELSE:

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

Uso de la Función 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
...				
10	Ernst	IT_PROG	6000	6600
11	Lorentz	IT_PROG	4200	4620
12	Mourgos	ST_MAN	5800	5800
13	Rajs	ST_CLERK	3500	4025
...				
19	Taylor	SA_REP	8600	10320
20	Grant	SA_REP	7000	8400

Uso de la Función DECODE

Mostrar el impuesto aplicable para cada empleado del departamento 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;
```

Prueba

La función `TO_NUMBER` convierte las cadenas de caracteres o los valores de fecha a un número con el formato especificado por el modelo de formato opcional.

1. True
2. Falso

Resumen

En esta lección, debe haber aprendido lo siguiente:

- Modificar formatos de fecha para su visualización utilizando funciones
- Convertir tipos de dato de columna utilizando funciones
- Utilizar funciones `NVL`
- Utilizar la lógica `IF-THEN-ELSE` y otra expresión condicional en una sentencia `SELECT`

Práctica 4: Visión General

En esta práctica se abordan los siguientes temas:

- Creación de consultas que utilizan `TO_CHAR`, `TO_DATE` y otras funciones `DATE`
- Creación de consultas que utilizan expresiones condicionales como `DECODE` y `CASE`