

# 3

## Funciones de una Sola Fila

ORACLE

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

# Objetivos

**Al finalizar esta lección, debería estar capacitado para:**

- **Describir varios tipos de funciones disponibles en SQL**
- **Utilizar funciones de caracteres, numéricas y fecha en sentencias `SELECT`**
- **Describir el uso de funciones de conversión**

ORACLE

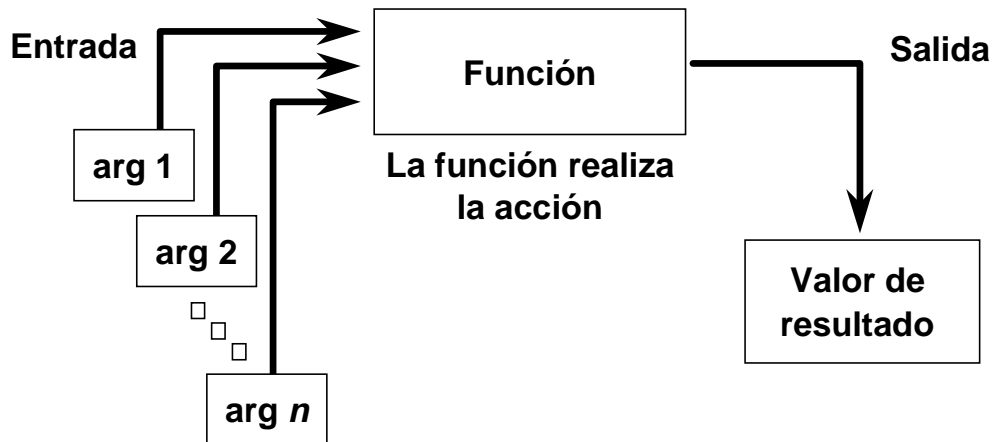
3-2

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Objetivo de la Lección

Las funciones hacen más potente el bloque de consulta básico y se utilizan para manipular valores de datos. Ésta es la primera de las dos lecciones que exploran las funciones. Se centra en funciones de caracteres, numéricas y de fecha de una sola fila, así como en otras funciones que convierten datos de un tipo en otro; por ejemplo, datos de caracteres en datos numéricos.

## Funciones SQL



ORACLE

3-3

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones SQL

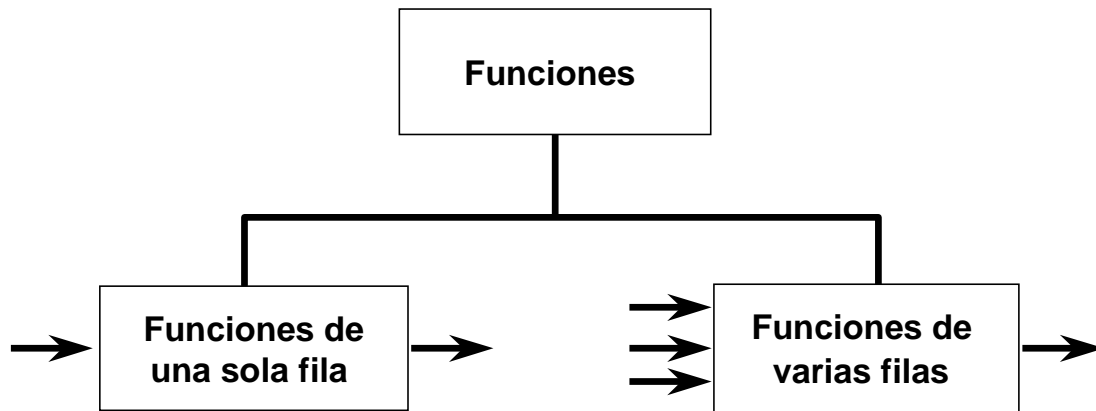
Las funciones son unas funcionalidades potentes de SQL y se pueden utilizar para lo siguiente:

- Realizar cálculos sobre datos
- Modificar elementos de datos individuales
- Manipular el resultado para grupos de filas
- Formatear fechas y números para su visualización
- Convertir tipos de dato de columna

Las funciones SQL a veces toman argumentos y siempre devuelven un valor.

**Nota:** La mayor parte de las funciones descritas en esta lección son específicas a la versión Oracle de SQL.

## Dos Tipos de Funciones SQL



ORACLE

3-4

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones SQL (continuación)

Hay dos tipos distintos de funciones:

- Funciones de una sola fila
- Funciones de varias filas

#### Funciones de una Sola Fila

Estas funciones solamente operan en una fila y devuelven un resultado por fila. Hay distintos tipos de funciones de una sola fila. Esta lección cubre las siguientes:

- Carácter
- Número
- Fecha
- Conversión

#### Funciones de Varias Filas

Las funciones pueden manipular grupos de filas para proporcionar un resultado por cada uno de ellos. Estas funciones se conocen como funciones de grupo y se tratan en una lección posterior.

Para obtener más información, consulte *Oracle9i SQL Reference* para ver la lista completa de funciones disponibles y su sintaxis.

# Funciones de una Sola Fila

## Las funciones de una sola fila:

- Manipulan elementos de datos.
- Aceptan argumentos y devuelven un valor.
- Actúan sobre cada fila devuelta.
- Devuelven un resultado por fila.
- Pueden modificar el tipo de dato.
- Se pueden anidar.
- Aceptan argumentos que pueden ser una columna o una expresión.

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

ORACLE

3-5

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Funciones de una Sola Fila

Las funciones de una sola fila se utilizan para manipular elementos de datos. Aceptan uno o varios argumentos y devuelven un valor para cada fila devuelta por la consulta. El argumento puede ser uno de los siguientes:

- Constante proporcionada por el usuario
- Valor de variable
- Nombre de columna
- Expresión

Las funcionalidades de las funciones de una sola fila incluyen:

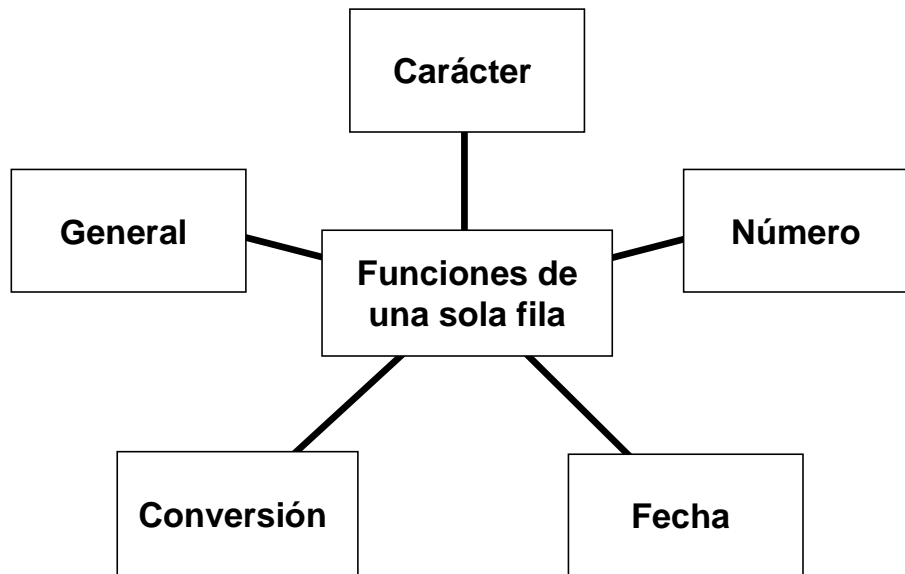
- Actuar sobre cada fila devuelta en la consulta
- Devolver un resultado por fila
- Devolver posiblemente un valor de datos de un tipo diferente al de referencia
- Esperar posiblemente uno o varios argumentos
- Se pueden utilizar en cláusulas SELECT, WHERE y ORDER BY; se pueden anidar

En la sintaxis:

*function\_name* es el nombre de la función.

*arg1, arg2* es cualquier argumento que debe utilizar la función. Puede venir representado por un nombre de columna o una expresión.

## Funciones de una Sola Fila



ORACLE

3-6

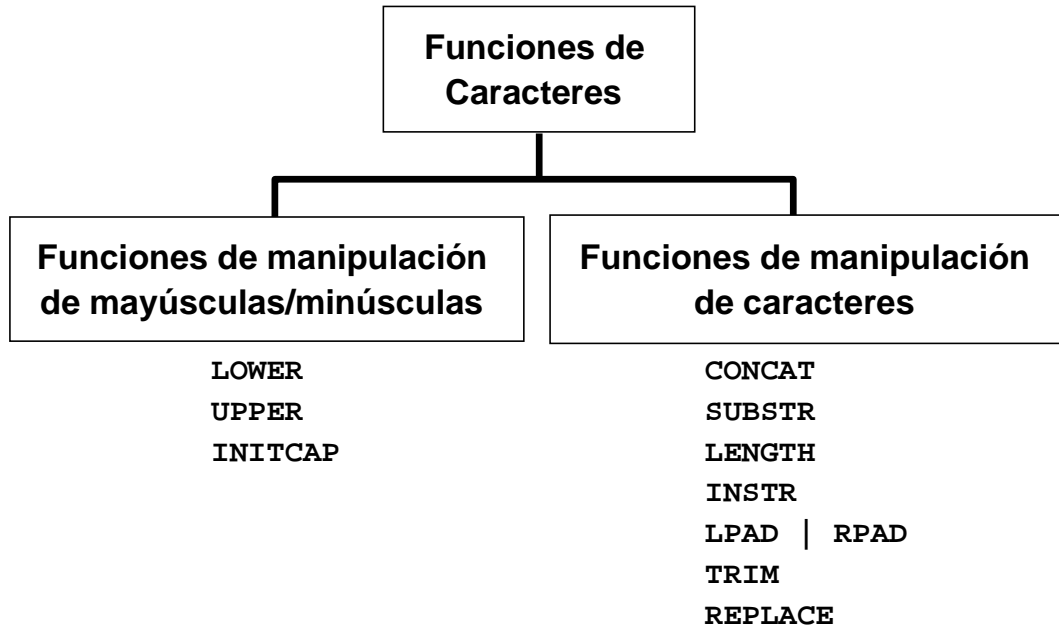
Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de una Sola Fila (continuación)

Esta lección cubre las siguientes funciones de una sola fila:

- Funciones de caracteres: Aceptan entradas de caracteres y pueden devolver valores de caracteres y numéricos.
- Funciones numéricas: Aceptan entradas numéricas y devuelven valores numéricos.
- Funciones de fecha: Operan sobre valores del tipo de dato DATE. (Todas las funciones de fecha devuelven un valor del tipo de dato DATE excepto la función MONTHS\_BETWEEN, que devuelve un número).
- Funciones de conversión: Convierten un valor de un tipo de dato en otro.
- Funciones generales:
  - NVL
  - NVL2
  - NULLIF
  - COALSECE
  - CASE
  - DECODE

# Funciones de Caracteres



ORACLE

3-7

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Funciones de Caracteres

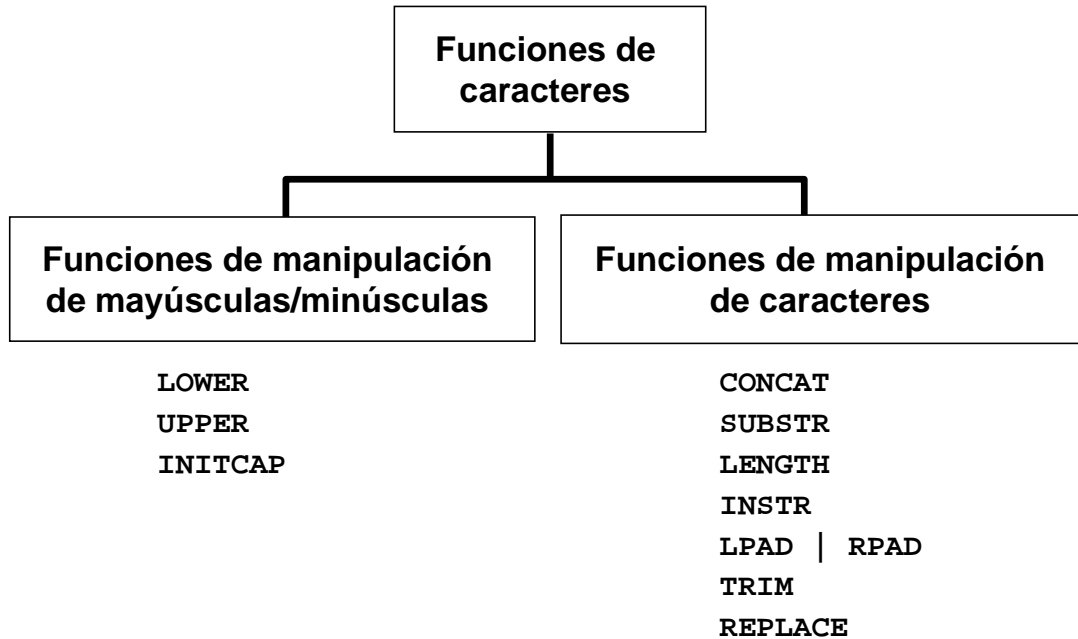
Las funciones de caracteres de una sola fila aceptan datos de caracteres como entrada y pueden devolver valores tanto de caracteres como numéricos. Estas funciones se pueden dividir en las siguientes:

- Funciones de manipulación de mayúsculas/minúsculas
- Funciones de manipulación de caracteres

Función	Objetivo
<code>LOWER(column/expression)</code>	Convierte valores de caracteres alfabéticos a minúsculas.
<code>UPPER(column/expression)</code>	Convierte valores de caracteres alfabéticos a mayúsculas.
<code>INITCAP(column/expression)</code>	Convierte valores de caracteres alfabéticos a mayúsculas para la primera letra de cada palabra, y todas las demás letras a minúsculas.
<code>CONCAT(column1/expression1, column2/expression2)</code>	Concatena el primer valor de caracteres con el segundo valor de caracteres; es equivalente al operador de concatenación ( <code>  </code> ).
<code>SUBSTR(column/expression, m[, n])</code>	Devuelve los caracteres especificados a partir del valor de caracteres que comienza en la posición de carácter <i>m</i> , con una longitud de <i>n</i> caracteres. (Si <i>m</i> es negativo, el recuento comienza desde el final del valor de caracteres. Si <i>n</i> está omitido, se devuelven todos los caracteres hasta el final de la cadena).

**Nota:** Las funciones tratadas en esta lección son solamente algunas de las disponibles.

# Funciones de Caracteres



ORACLE

3-8

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Funciones de Caracteres (continuación)

Función	Objetivo
<code>LENGTH(column expression)</code>	Devuelve el número de caracteres de la expresión.
<code>INSTR(column expression, 'string', [,m], [n] )</code>	Devuelve la posición numérica de una cadena especificada. Opcionalmente, puede proporcionar la posición <i>m</i> para comenzar la búsqueda y la incidencia <i>n</i> de la cadena. <i>m</i> y <i>n</i> toman por defecto el valor 1, lo que significa comenzar la búsqueda desde el principio de la cadena e informar de la primera incidencia.
<code>LPAD(column expression, n, 'string')</code> <code>RPAD(column expression, n, 'string')</code>	Rellena el valor de caracteres justificado a la derecha hasta un ancho total de <i>n</i> posiciones de caracteres. Rellena el valor de caracteres justificado a la izquierda hasta un ancho total de <i>n</i> posiciones de caracteres.
<code>TRIM(leading/trailing/both, trim_character FROM trim_source)</code>	Le permite recortar caracteres iniciales o finales (o ambos) de una cadena de caracteres. Si <i>trim_character</i> o <i>trim_source</i> es un literal de carácter, debe escribirlo entre comillas simples. Esta función está disponible a partir de Oracle8i.
<code>REPLACE(text, search_string, replacement_string)</code>	Busca en una expresión de texto una cadena de caracteres y, si la encuentra, la sustituye por una cadena especificada.



## Funciones de Manipulación de Mayúsculas/Minúsculas

Estas funciones convierten las mayúsculas/minúsculas para cadenas de caracteres.

Function	Result
LOWER( 'SQL Course' )	sql course
UPPER( 'SQL Course' )	SQL COURSE
INITCAP( 'SQL Course' )	Sql Course

ORACLE

3-9

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Manipulación de Mayúsculas/Minúsculas

LOWER, UPPER e INITCAP son las tres funciones de conversión de mayúsculas/minúsculas.

- LOWER: Convierte cadenas de caracteres en mayúsculas o mezclados a minúsculas.
- UPPER: Convierte cadenas de caracteres en minúsculas o mezclados a mayúsculas.
- INITCAP: Convierte la primera letra de cada palabra a mayúsculas y las restantes letras a minúsculas.

```
SELECT 'The job id for ' || UPPER(last_name) || ' is '  
      || LOWER(job_id) AS "EMPLOYEE DETAILS"  
FROM   employees;
```

EMPLOYEE DETAILS
The job id for KING is ad_pres
The job id for KOCHHAR is ad_vp
The job id for DE HAAN is ad_vp
...
The job id for HIGGINS is ac_mgr
The job id for GIETZ is ac_account

20 rows selected.

## Uso de Funciones de Manipulación de Mayúsculas/Minúsculas

**Muestre el número de empleado, el nombre y el número de departamento para el empleado 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

ORACLE

3-10

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Manipulación de Mayúsculas/Minúsculas (continuación)

El ejemplo de la transparencia muestra el número de empleado, el nombre y el número de departamento del empleado Higgins.

La cláusula WHERE de la primera sentencia SQL especifica el nombre de empleado como `higgins`. Como todos los datos de la tabla `EMPLOYEES` se almacenan con las mayúsculas/minúsculas correspondientes, el nombre `higgins` no encuentra ninguna coincidencia en la tabla y no se selecciona ninguna fila.

La cláusula WHERE de la segunda sentencia SQL especifica que el nombre del empleado de la tabla `EMPLOYEES` se compara con `higgins`, convirtiendo la columna `LAST_NAME` en minúsculas para la comparación. Como ahora están en minúsculas los dos nombres, se encuentra una coincidencia y se selecciona una fila. La cláusula WHERE se puede reescribir de la siguiente manera para proporcionar el mismo resultado:

```
...WHERE last_name = 'Higgins'
```

El nombre del resultado aparece como se almacenó en la base de datos. Para mostrar el nombre en mayúsculas, utilice la función `UPPER` en la sentencia `SELECT`.

```
SELECT employee_id, UPPER(last_name), department_id
FROM   employees
WHERE  INITCAP(last_name) = 'Higgins';
```

# Funciones de Manipulación de Caracteres

Estas funciones manipulan cadenas de caracteres:

Función	Resultado
CONCAT('Hello', 'World')	HelloWorld
SUBSTR('HelloWorld',1,5)	Hello
LENGTH('HelloWorld')	10
INSTR('HelloWorld', 'W')	6
LPAD(salary,10,'*')	*****24000
RPAD(salary, 10, '*')	24000*****
TRIM('H' FROM 'HelloWorld')	elloWorld

ORACLE

3-11

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Funciones de Manipulación de Caracteres

CONCAT, SUBSTR, LENGTH, INSTR, LPAD, RPAD y TRIM son las funciones de manipulación de caracteres que se tratan en esta lección.

- CONCAT: Une valores (con esta función está limitado a utilizar dos parámetros.)
- SUBSTR: Extrae una cadena de una longitud determinada.
- LENGTH: Muestra la longitud de una cadena como valor numérico.
- INSTR: Busca la posición numérica de un carácter especificado.
- LPAD: Rellena el valor de caracteres justificado a la derecha.
- RPAD: Rellena el valor de caracteres justificado a la izquierda.
- TRIM: Recorta caracteres iniciales o finales (o ambos) de una cadena de caracteres. (Si *trim\_character* o *trim\_source* es un literal de carácter, debe escribirlo entre comillas simples.)

## Uso de Funciones de Manipulación de Caracteres

**SQL Query:**

```

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';
    
```

**Annotations:**

- 1: Points to the `NAME` alias in the SELECT clause.
- 2: Points to the `LENGTH (last_name)` expression.
- 3: Points to the `"Contains 'a'?"` alias for the `INSTR` function.

**Result Table:**

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

**Table Annotations:**

- 1: Points to the `NAME` column.
- 2: Points to the `LENGTH(LAST_NAME)` column.
- 3: Points to the `Contains 'a'?` column.

ORACLE

3-12

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Manipulación de Caracteres (continuación)

El ejemplo de la transparencia muestra nombres y apellidos de empleados unidos, la longitud del apellido del empleado y la posición numérica de la letra *a* en el apellido del empleado para todos los empleados que tienen la cadena REP en el identificador de trabajo comenzando por la cuarta posición del identificador de trabajo.

#### Ejemplo

Modifique la sentencia SQL de la transparencia para mostrar los datos para los empleados cuyos apellidos terminan en *n*.

```

SELECT employee_id, CONCAT(first_name, last_name) NAME,
       LENGTH (last_name), INSTR(last_name, 'a') "Contains 'a'?"
FROM   employees
WHERE  SUBSTR(last_name, -1, 1) = 'n';
    
```

EMPLOYEE_ID	NAME	LENGTH(LAST_NAME)	Contains 'a'?
102	LexDe Haan	7	5
200	JenniferWhalen	6	3
201	MichaelHartstein	9	2

## Funciones Numéricas

- **ROUND:** Redondea el valor a los decimales especificados.

`ROUND(45.926, 2)`  $\longrightarrow$  45.93

- **TRUNC:** Trunca el valor a los decimales especificados.

`TRUNC(45.926, 2)`  $\longrightarrow$  45.92

- **MOD:** Devuelve el resto de la división.

`MOD(1600, 300)`  $\longrightarrow$  100

ORACLE

### Funciones Numéricas

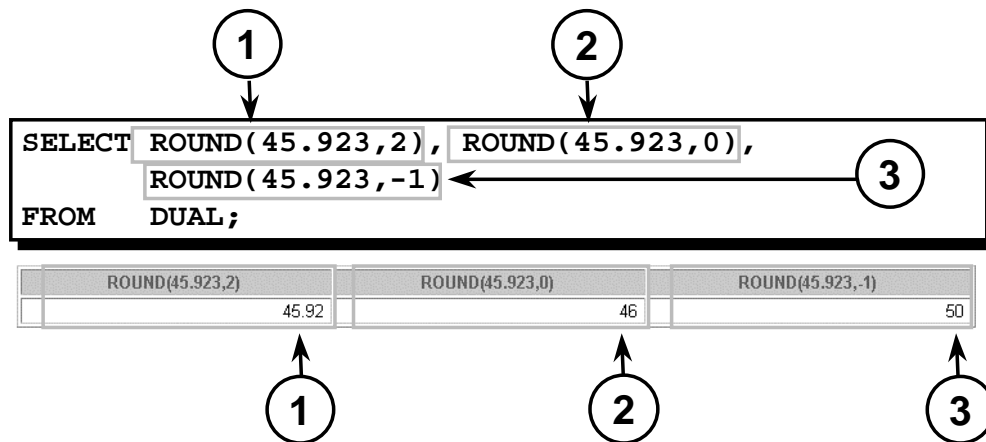
Las funciones numéricas aceptan entradas numéricas y devuelven valores numéricos. Esta sección describe algunas de las funciones numéricas.

Función	Objetivo
<code>ROUND(column expression, n)</code>	Redondea la columna, la expresión o el valor en <i>n</i> Posiciones decimales o, si <i>n</i> está omitido, en cero posiciones decimales. (Si <i>n</i> es negativo, se redondean los números a la izquierda de la coma decimal.)
<code>TRUNC(column expression, n)</code>	Trunca la columna, la expresión o el valor en <i>n</i> Posiciones decimales; si <i>n</i> está omitido, toma el valor por defecto cero.
<code>MOD(m, n)</code>	Devuelve el resto de <i>m</i> dividido por <i>n</i> .

**Nota:** Esta lista sólo contiene algunas de las funciones numéricas disponibles.

Para obtener más información, consulte *Oracle9i SQL Reference*, “Number Functions”.

## Uso de la Función ROUND



**DUAL es una tabla ficticia que puede utilizar para ver los resultados de funciones y cálculos.**

ORACLE

### Función ROUND

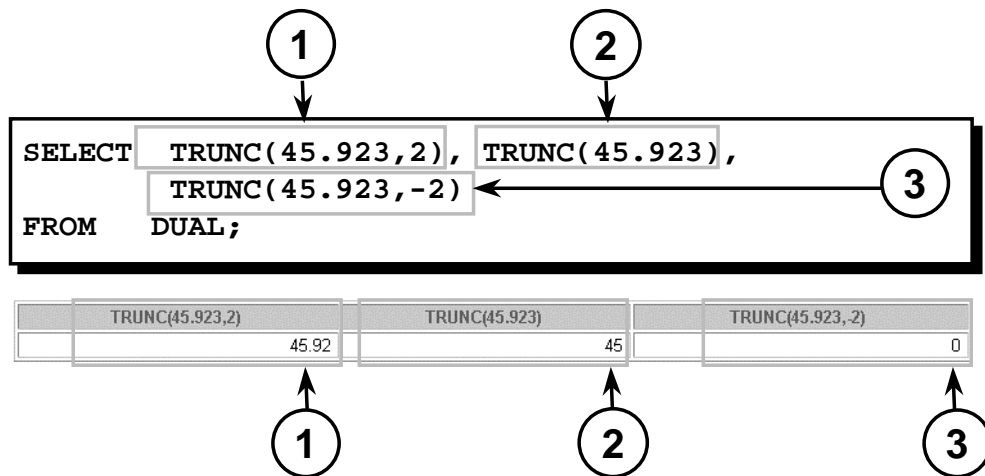
La función ROUND redondea la columna, la expresión o el valor en *n* posiciones decimales. Si el segundo argumento falta o es 0, el valor se redondea a cero posiciones decimales. Si es 2, el valor se redondea a dos posiciones decimales. A la inversa, si el segundo argumento es -2, el valor se redondea a dos posiciones decimales hacia la izquierda.

La función ROUND también se puede utilizar con funciones de fecha. Verá ejemplos más adelante en esta lección.

#### La Tabla DUAL

La tabla DUAL es propiedad del usuario SYS y todos los usuarios pueden acceder a ella. Contiene una columna, DUMMY, y una fila con el valor X. Esta tabla resulta útil si desea devolver un valor una sola vez, por ejemplo, el valor de una constante, una pseudocolumna o una expresión que no esté derivada de una tabla con datos de usuario. La tabla DUAL se utiliza generalmente para completar la sintaxis de la cláusula SELECT, ya que las cláusulas SELECT y FROM son las dos obligatorias y hay algunos cálculos que no necesitan seleccionar desde tablas reales.

## Uso de la Función TRUNC



ORACLE

3-15

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Función TRUNC

La función TRUNC trunca la columna, la expresión o el valor en  $n$  posiciones decimales.

Esta función trabaja con argumentos similares a los de la función ROUND. Si el segundo argumento falta o es 0, el valor se trunca a cero posiciones decimales. Si es 2, el valor se trunca a dos posiciones decimales. A la inversa, si el segundo argumento es -2, el valor se trunca a dos posiciones decimales a la izquierda.

Al igual que la función ROUND, la función TRUNC se puede utilizar con funciones de fecha.

## Uso de la Función MOD

Calcule el resto de un salario después de dividirlo por 5000 para todos los empleados cuyos cargos son representantes de ventas.

```
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

ORACLE

### Función MOD

La función MOD busca el resto de valor1 dividido por valor2. El ejemplo de la transparencia calcula el resto del salario después de dividirlo por 5.000 para todos los empleados cuyos identificadores de cargo son SA\_REP.

**Nota:** La función MOD se utiliza a menudo para determinar si un valor es par o impar.



## Trabajo con Fechas

- La base de datos Oracle almacena fechas en un formato numérico interno: siglo, año, mes, día, horas, minutos, segundos.
- El formato de visualización de fecha por defecto es DD-MON-RR.
  - Le permite almacenar fechas del siglo XXI en el siglo XX especificando solamente los dos últimos dígitos del año.
  - Le permite almacenar fechas del siglo XX en el siglo XXI de la misma forma.

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

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

ORACLE

3-17

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Formato de Fechas de Oracle

La base de datos Oracle almacena fechas en un formato numérico interno, representando el siglo, el año, el mes, el día, las horas, los minutos y los segundos.

El formato de entrada y visualización por defecto de cualquier fecha es DD-MON-RR. Las fechas válidas para Oracle están comprendidas entre el 1 de enero de 4712 a.C. y el 31 de diciembre de 9999 d.C.

En el ejemplo de la transparencia, HIRE\_DATE del empleado Gietz se muestra en el formato por defecto DD-MON-RR. Sin embargo, las fechas no se almacenan en la base de datos con este formato. Se almacenan todos los componentes de fecha y hora. Por ello, aunque una HIRE\_DATE como 07-JUN-94 se muestre como día, mes y año, también hay una información de *hora* y *siglo* asociada a ella. Los datos completos serían 7 de junio de 1994 5:10:43 p.m.

Los datos se almacenan internamente como se indica a continuación:

CENTURY	YEAR	MONTH	DAY	HOURL	MINUTE	SECOND
19	94	06	07	5	10	43

### Siglos y el Año 2000

Oracle Server cumple con el año 2000. Al insertar en una tabla un registro con una columna de fecha, la información de *siglo* se recoge de la función SYSDATE. Sin embargo, cuando se muestra la columna de fecha en pantalla, el componente siglo no aparece por defecto.

El tipo de dato DATE siempre almacena la información de año internamente como número de cuatro dígitos: dos dígitos para el siglo y otros dos para el año. Por ejemplo, la base de datos Oracle almacena el año como 1996 ó 2001 y no simplemente como 96 ó 01.

# Trabajo con Fechas

**SYSDATE es una función que devuelve:**

- Fecha
- Hora

ORACLE

3-18

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## La Función SYSDATE

SYSDATE es una función de fecha que devuelve la fecha y la hora actuales del servidor de base de datos. Puede utilizar SYSDATE de la misma forma que utilizaría cualquier otro nombre de columna. Por ejemplo, puede mostrar la fecha actual seleccionando SYSDATE desde una tabla. Lo habitual es seleccionar SYSDATE desde una tabla ficticia llamada DUAL.

### Ejemplo

Visualice la fecha actual utilizando la tabla DUAL.

```
SELECT SYSDATE  
FROM DUAL;
```

SYSDATE
28-SEP-01

## Aritmética con Fechas

- **Sume o reste un número a/de una fecha para producir un valor de fecha.**
- **Reste dos fechas para buscar el número de días entre ellas.**
- **Sume horas a una fecha dividiendo el número de horas por 24.**

ORACLE

3-19

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Aritmética con Fechas

Como la base de datos almacena las fechas como números, puede realizar cálculos utilizando operadores aritméticos como la suma y la resta. Puede sumar y restar constantes numéricas así como fechas.

Puede realizar las siguientes operaciones:

Operación	Resultado	Descripción
fecha + número	Fecha	Suma un número de días a una fecha.
fecha - número	Fecha	Resta un número de días de una fecha.
fecha - fecha	Número de días	Resta una fecha de otra.
fecha + número/24	Fecha	Suma un número de horas a una fecha.

## Uso de Operadores Aritméticos con Fechas

```
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

ORACLE

### Aritmética con Fechas (continuación)

El ejemplo de la transparencia muestra el apellido y el número de semanas de empleo de todos los trabajadores del departamento 90. Resta la fecha en la que se contrató al empleado de la fecha actual (SYSDATE) y divide el resultado por 7 para calcular el número de semanas que lleva empleado un trabajador.

**Nota:** SYSDATE es una función SQL que devuelve la fecha y la hora actuales. Es posible que los resultados que obtenga difieran del ejemplo.

Si a una fecha se le resta otra más reciente, la diferencia es un número negativo.

## Funciones de Fecha

Función	Descripción
MONTHS_BETWEEN	Número de meses entre dos fechas
ADD_MONTHS	Suma meses de calendario a una fecha
NEXT_DAY	Siguiente día de la fecha especificada
LAST_DAY	Último día del mes
ROUND	Redondea la fecha
TRUNC	Trunca la fecha

ORACLE

3-21

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Fecha

Las funciones de fecha operan sobre fechas de Oracle. Todas las funciones de fecha devuelven un valor del tipo de dato DATE excepto MONTHS\_BETWEEN, que devuelve un valor numérico.

- MONTHS\_BETWEEN(*date1*, *date2*): Busca el número de meses entre *date1* y *date2*. El resultado puede ser positivo o negativo. Si *date1* es posterior a *date2*, el resultado es positivo; si *date1* es anterior a *date2*, el resultado es negativo. La parte no entera del resultado representa una porción del mes.
- ADD\_MONTHS(*date*, *n*): Suma *n* meses de calendario a *date*. El valor de *n* debe ser entero y puede ser negativo.
- NEXT\_DAY(*date*, '*char*'): Busca la fecha del siguiente día de la semana especificado ('*char*') posterior a *date*. El valor de *char* puede ser un número que represente un día o una cadena de caracteres.
- LAST\_DAY(*date*): Busca la fecha del último día del mes en el que está *date*.
- ROUND(*date*[, '*fmt*']): Devuelve *date* redondeado a la unidad especificada por el modelo de formato *fmt*. Si el modelo de formato *fmt* está omitido, *date* se redondea al día más próximo.
- TRUNC(*date*[, '*fmt*']): Devuelve *date* con la parte de hora del día truncada a la unidad especificada por el modelo de formato *fmt*. Si el modelo de formato *fmt* está omitido, *date* se trunca al día más próximo.

Esta lista es un subconjunto de las funciones de fecha disponibles. Los modelos de formato se tratan más adelante en esta lección. Ejemplos de modelos de formato son mes y año.

## Uso de Funciones de Fecha

- 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'

ORACLE

3-22

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Fecha (continuación)

Por ejemplo, muestre el número de empleado, la fecha de contratación, el número de meses empleado, la fecha de revisión de seis meses, el primer viernes después de la fecha de contratación y el último día del mes de contratación para todos los trabajadores que lleven empleados menos de 36 meses.

```
SELECT employee_id, hire_date,
       MONTHS_BETWEEN (SYSDATE, hire_date) TENURE,
       ADD_MONTHS (hire_date, 6) REVIEW,
       NEXT_DAY (hire_date, 'FRIDAY'), LAST_DAY(hire_date)
FROM   employees
WHERE  MONTHS_BETWEEN (SYSDATE, hire_date) < 36;
```

EMPLOYEE_ID	HIRE_DATE	TENURE	REVIEW	NEXT_DAY(	LAST_DAY(
107	07-FEB-99	31.6982407	07-AUG-99	12-FEB-99	28-FEB-99
124	16-NOV-99	22.4079182	16-MAY-00	19-NOV-99	30-NOV-99
149	29-JAN-00	19.9885633	29-JUL-00	04-FEB-00	31-JAN-00
178	24-MAY-99	28.1498536	24-NOV-99	28-MAY-99	31-MAY-99

## Uso de Funciones de Fecha

**Asuma** 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

ORACLE

3-23

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Fecha (continuación)

Las funciones ROUND y TRUNC se pueden utilizar para valores numéricos y de fecha. Cuando se utilizan con fechas, estas funciones redondean o truncan al modelo de formato especificado. Por lo tanto, puede redondear fechas al año o mes más próximo.

#### Ejemplo

Compare las fechas de contratación para todos los empleados que comenzaron en 1997. Visualice el número de empleado, la fecha de contratación y el mes de inicio utilizando las funciones ROUND y TRUNC.

```
SELECT employee_id, hire_date,  
       ROUND(hire_date, 'MONTH'), TRUNC(hire_date, 'MONTH')  
FROM   employees  
WHERE  hire_date LIKE '%97';
```

EMPLOYEE_ID	HIRE_DATE	ROUND(HIR	TRUNC(HIR
142	29-JAN-97	01-FEB-97	01-JAN-97
202	17-AUG-97	01-SEP-97	01-AUG-97

## **Práctica 3, Parte Uno: Visión General**

**Esta práctica cubre los siguientes temas:**

- **Escritura de una consulta que muestre la fecha actual**
- **Creación de consultas que requieran el uso de funciones numéricas, de caracteres y de fecha**
- **Realización de cálculos de meses y años de servicio para un empleado**

ORACLE

3-24

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

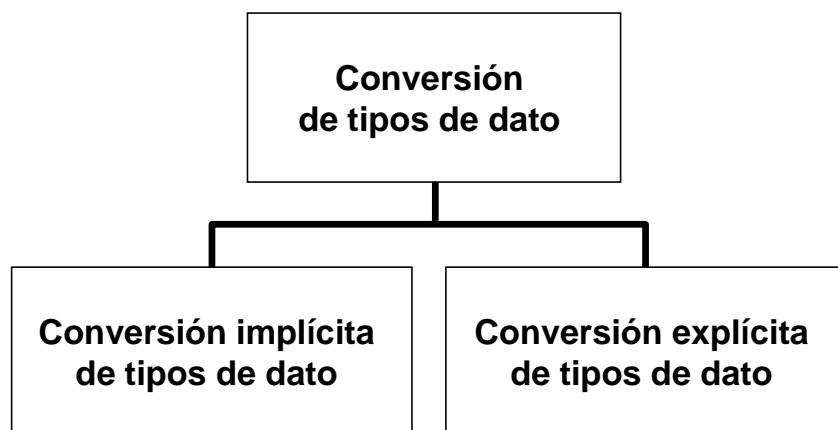
### **Práctica 3, Parte Uno: Visión General**

Esta práctica se ha diseñado para ofrecerle diversos ejercicios que utilicen las distintas funciones disponibles para tipos de dato de caracteres, numéricos y de fecha.

Responda las 5 preguntas al final de esta lección.



## Funciones de Conversión



ORACLE

3-25

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Conversión

Además de los tipos de dato Oracle, las columnas de las tablas de las bases de datos Oracle9i se pueden definir utilizando tipos de dato ANSI, DB2 y SQL/DS. Sin embargo, Oracle Server convierte internamente estos tipos de dato en tipos de dato Oracle.

En algunos casos, Oracle Server utiliza datos de un tipo donde espera datos de otro tipo distinto. Cuando esto ocurre, Oracle Server puede convertir automáticamente los datos al tipo de dato esperado. Esta conversión la puede hacer Oracle Server *implícitamente* o el usuario *explícitamente*.

Las conversiones implícitas de tipos de dato se realizan de acuerdo con las reglas explicadas en las dos transparencias siguientes.

Las conversiones explícitas de tipos de dato se realizan utilizando las funciones de conversión, que convierten un valor de un tipo de dato en otro. Generalmente, la forma de los nombres de función sigue la convención *data type TO data type*. El primer tipo de dato es el de entrada y el último, el de salida.

**Nota:** Aunque la conversión implícita de tipos de dato está disponible, se recomienda que haga conversiones explícitas de tipos de dato para asegurar la fiabilidad de las sentencias SQL.

## Conversión Implícita de Tipos de Dato

Para las asignaciones, Oracle Server puede convertir automáticamente lo siguiente:

De	A
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

ORACLE

### Conversión Implícita de Tipos de Dato

La asignación es correcta si Oracle Server puede convertir el tipo de dato del valor utilizado en la asignación al del destino de la asignación.

## Conversión Implícita de Tipos de Dato

Para la evaluación de la expresión, Oracle Server puede convertir automáticamente lo siguiente:

De	A
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE

ORACLE

3-27

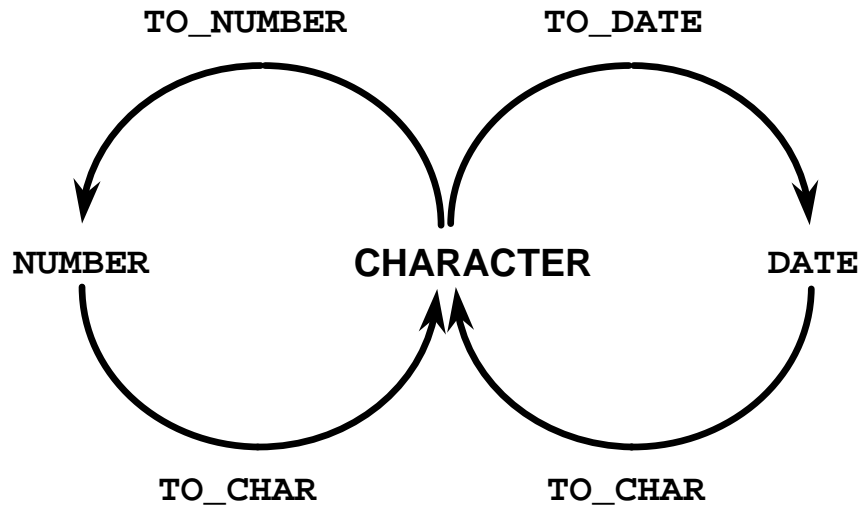
Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Conversión Implícita de Tipos de Dato (continuación)

En general, Oracle Server utiliza la regla para expresiones cuando es necesaria una conversión de tipos de dato en lugares no cubiertos por una regla para conversiones de asignación.

**Nota:** Las conversiones CHAR a NUMBER sólo son correctas si la cadena de caracteres representa un número válido.

## Conversión Explícita de Tipos de Dato



ORACLE

3-28

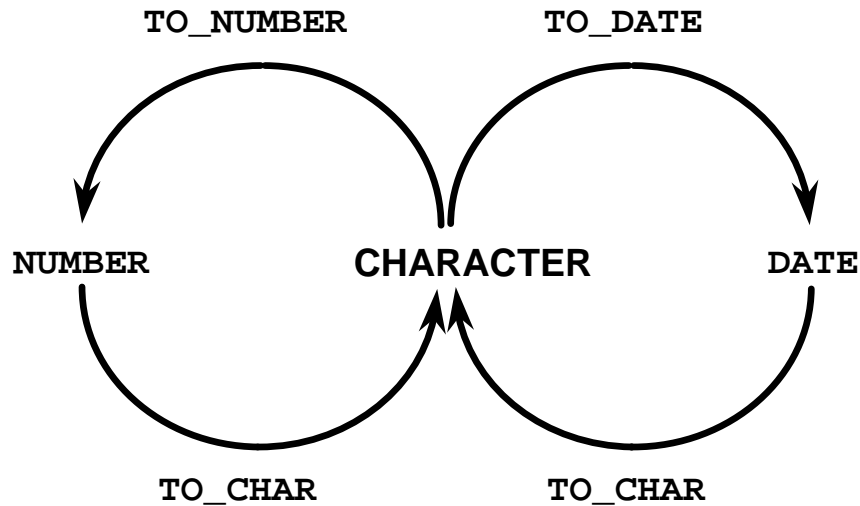
Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Conversión Explícita de Tipos de Dato

SQL proporciona tres funciones para convertir un valor de un tipo de dato en otro:

Función	Objetivo
<code>TO_CHAR(<i>number</i> <i>date</i>, [ <i>fmt</i> ], [ <i>nlsparms</i> ])</code>	<p>Convierte un valor numérico o de fecha en una cadena de caracteres VARCHAR2 con el modelo de formato <i>fmt</i>.</p> <p><b>Conversión Numérica:</b> El parámetro <i>nlsparms</i> especifica los siguientes caracteres, que se devuelven por elementos de formato numérico:</p> <ul style="list-style-type: none"><li>• Carácter decimal</li><li>• Separador de grupo</li><li>• Símbolo de divisa local</li><li>• Símbolo de divisa internacional</li></ul> <p>Si <i>nlsparms</i> o cualquier otro parámetro está omitido, esta función utiliza los valores de parámetro por defecto para la sesión.</p>

## Conversión Explícita de Tipos de Dato



ORACLE

3-29

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Conversión Explícita de Tipos de Dato (continuación)

Función	Objetivo
<code>TO_CHAR(<i>number</i>   <i>date</i>, [ <i>fmt</i> ], [ <i>nlsparams</i> ])</code>	<b>Conversión de Fecha:</b> El parámetro <i>nlsparams</i> especifica el idioma en el que se devuelven los nombres y las abreviaturas de mes y día. Si este parámetro está omitido, la función utiliza los idiomas de fecha por defecto para la sesión.
<code>TO_NUMBER(<i>char</i>, [ <i>fmt</i> ], [ <i>nlsparams</i> ])</code>	Convierte una cadena de caracteres que contenga dígitos en un número en el formato especificado por el modelo de formato opcional <i>fmt</i> .  El parámetro <i>nlsparams</i> tiene el mismo objetivo que en la función <code>TO_CHAR</code> para la conversión numérica.
<code>TO_DATE(<i>char</i>, [ <i>fmt</i> ], [ <i>nlsparams</i> ])</code>	Convierte una cadena de caracteres que representa una fecha en un valor de fecha de acuerdo con <i>fmt</i> especificado. Si <i>fmt</i> está omitido, el formato es DD-MON-YY.  El parámetro <i>nlsparams</i> tiene el mismo objetivo que en la función <code>TO_CHAR</code> para la conversión de fecha.

## Conversión Explícita de Tipos de Dato (continuación)

**Nota:** La lista de funciones mencionada en esta lección sólo incluye algunas de las funciones de conversión disponibles.

Para obtener más información, consulte *Oracle9i SQL Reference*, “Conversion Functions”.

## Uso de la Función TO\_CHAR con Fechas

```
TO_CHAR(date, 'format_model')  

```

### El modelo de formato:

- Se debe escribir entre comillas sencillas y es sensible a mayúsculas/minúsculas.
- Puede incluir cualquier elemento de formato de fecha válido.
- Tiene un elemento *fm* para eliminar espacios rellenos o suprimir ceros a la izquierda.
- Se separa del valor de fecha con una coma.

ORACLE

3-31

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Visualización de una Fecha en un Formato Específico

Anteriormente, todos los valores de fecha de Oracle se mostraban en formato DD-MON-YY. Puede utilizar la función TO\_CHAR para convertir una fecha de este formato por defecto al que especifique.

#### Instrucciones

- El modelo de formato se debe escribir entre comillas sencillas y es sensible a mayúsculas/minúsculas.
- El modelo de formato puede incluir cualquier elemento de formato de fecha válido. Asegúrese de separar el valor de fecha del modelo de formato mediante una coma.
- Los nombres de los días y los meses en la salida se rellenan automáticamente con espacios en blanco.
- Para eliminar los espacios rellenos o suprimir los ceros a la izquierda, utilice el elemento *fm* de modo de relleno.
- Puede formatear el campo de caracteres resultante con el comando COLUMN de iSQL\*Plus que se cubre en una lección posterior.

```
SELECT employee_id, TO_CHAR(hire_date, 'MM/YY') Month_Hired  
FROM   employees  
WHERE  last_name = 'Higgins';
```

EMPLOYEE_ID	MONTH
205	06/94

## Elementos del Modelo de Formato de Fecha

<b>YYYY</b>	<b>Año completo en números</b>
<b>YEAR</b>	<b>Años en letra</b>
<b>MM</b>	<b>Valor de dos dígitos para el mes</b>
<b>MONTH</b>	<b>Nombre completo del mes</b>
<b>MON</b>	<b>Abreviatura de tres letras del mes</b>
<b>DY</b>	<b>Abreviatura de tres letras del día de la semana</b>
<b>DAY</b>	<b>Nombre completo del día de la semana</b>
<b>DD</b>	<b>Día del mes en número</b>

ORACLE



## Elementos de Formato de Ejemplo de Formatos de Fecha Válidos

Elemento	Descripción
SCC o CC	Siglo; el servidor pone el signo - delante de las fechas a.C.
Años en fechas YYYY o SYYYY	Año; el servidor pone el signo - delante de las fechas a.C.
YYY o YY o Y	Los últimos tres dígitos del año (tres, dos o uno)
Y,YYY	Año con coma en esta posición
IYYYY, IYY, IY, I	Año con cuatro dígitos (cuatro, tres, dos o uno) basado en el estándar ISO
SYEAR o YEAR	Año en letra; el servidor pone el signo - delante de las fechas a.C.
BC o AD	Indicador B.C./A.D. (a.C./d.C.)
B.C. o A.D.	Indicador B.C./A.D. (a.C./d.C.) con puntos
Q	Trimestre del año
MM	Mes: valor de dos dígitos
MONTH	Nombre del mes rellenado con espacios en blanco con un máximo de nueve caracteres
MON	Nombre del mes, abreviatura de tres letras
RM	Mes en numerales romanos
WW o W	Semana del año o del mes
DDD o DD o D	Día del año, del mes o de la semana
DAY	Nombre del día rellenado con espacios en blanco con un máximo de nueve caracteres
DY	Nombre del día; abreviatura de tres letras
J	Día juliano; el número de días desde el 31 de diciembre de 4713 a.C.

## Elementos del Modelo de Formato de Fecha

- Los elementos de hora formatean la porción de hora de la fecha.

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

- Agregue cadenas de caracteres escribiéndolas entre comillas dobles.

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

- Los sufijos numéricos escriben los números en letra.

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

ORACLE

3-34

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Elementos de Formato de Fecha: Formatos de Hora

Utilice los formatos listados en las siguientes tablas para mostrar información de hora y literales y para cambiar numerales a números en letra.

Elemento	Descripción
AM o PM	Indicador de meridiano
A.M. o P.M.	Indicador de meridiano con puntos
HH o HH12 o HH24	Hora del día, u hora (1-12) u hora (0-23)
MI	Minuto (0-59)
SS	Segundo (0-59)
SSSSS	Segundos tras las 12 de la noche (0-86399)

## Otros Formatos

Elemento	Descripción
/ . ,	La puntuación se reproduce en el resultado.
“of the”	La cadena entrecomillada se reproduce en el resultado.

## Especificación de Sufijos que Influyen en la Visualización de Números

Elemento	Descripción
TH	Número ordinal (por ejemplo, DDTH para 4TH)
SP	Número en letra (por ejemplo, DDSP para FOUR)
SPTH o THSP	Números ordinales en letra (por ejemplo, DDSPTH para FOURTH)

## 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
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.

ORACLE

3-36

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### La Función TO\_CHAR con Fechas

La sentencia SQL de la transparencia muestra los apellidos y las fechas de contratación de todos los empleados. La fecha de contratación aparece como 17 June 1987.

#### Ejemplo

Modifique el ejemplo de la transparencia para mostrar las fechas en un formato que aparezca como Seventh of June 1994 12:00:00 AM.

```
SELECT last_name,  
       TO_CHAR(hire_date,  
               'fmDdspth "of" Month YYYY fmHH:MI:SS AM')  
       AS HIREDATE  
FROM   employees;
```

LAST_NAME	HIREDATE
King	Seventeenth of June 1987 12:00:00 AM
Kochhar	Twenty-First of September 1989 12:00:00 AM
...	
Higgins	Seventh of June 1994 12:00:00 AM
Gietz	Seventh of June 1994 12:00:00 AM

20 rows selected.

Observe que el mes sigue el modelo de formato especificado: en otras palabras, la primera letra va en mayúsculas y el resto en minúsculas.

## Uso de la Función TO\_CHAR con Números

```
TO_CHAR(number, 'format_model')  

```

Estos son algunos de los elementos de formato que puede utilizar con la función TO\_CHAR para mostrar un valor numérico como carácter:

9	Representa un número.
0	Obliga a mostrar un cero.
\$	Coloca un signo de dólar flotante.
L	Utiliza el símbolo de divisa local flotante.
.	Imprime una coma decimal.
,	Imprime un indicador de miles.

ORACLE

3-37

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### La Función TO\_CHAR con Números

Al trabajar con valores numéricos como, por ejemplo, cadenas de caracteres, debe convertir estos números al tipo de dato de caracteres mediante la función TO\_CHAR, que traduce un valor del tipo de dato NUMBER al tipo de dato VARCHAR2. Esta técnica resulta especialmente útil con la concatenación.

## Elementos de Formato Numérico

Si está convirtiendo un número al tipo de dato de caracteres, puede utilizar los siguientes elementos de formato:

Elemento	Descripción	Ejemplo	Resultado
9	Posición numérica (el número de nueves determina el ancho de visualización)	999999	1234
0	Muestra ceros a la izquierda	099999	001234
\$	Signo de dólar flotante	\$999999	\$1234
L	Símbolo de divisa local flotante	L999999	FF1234
.	Coma decimal en la posición especificada	999999.99	1234.00
,	Coma en la posición especificada	999,999	1,234
MI	Signos menos a la derecha (valores negativos)	999999MI	1234-
PR	Números negativos entre paréntesis	999999PR	<1234>
EEEE	Notación científica (el formato debe especificar cuatro Es)	99.999EEEE	1.234E+03
V	Multiplica por 10 <i>n</i> veces ( <i>n</i> = número de nueves después de V)	9999V99	123400
B	Muestra valores cero como espacios en blanco y no como 0	B9999.99	1234.00

## Uso de la Función TO\_CHAR con Números

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

SALARY
\$6,000.00

ORACLE

### Instrucciones

- Oracle Server muestra una cadena de signos almohadilla (#) en lugar de un número entero cuyos dígitos excedan el número de dígitos que se proporciona en el modelo de formato.
- Oracle Server redondea el valor decimal almacenado al número de espacios decimales proporcionados en el modelo de formato.

## Uso de las Funciones TO\_NUMBER y TO\_DATE

- Convierta una cadena de caracteres en formato numérico utilizando la función TO\_NUMBER:

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

- Convierta una cadena de caracteres en formato de fecha utilizando la función TO\_DATE:

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

- Estas funciones tienen un modificador **fx** que especifica la coincidencia exacta para el argumento de caracteres y un modelo de formato de fecha de una función TO\_DATE.

ORACLE

3-40

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones TO\_NUMBER y TO\_DATE

Puede convertir una cadena de caracteres en un número o en una fecha. Para ello, utilice las funciones TO\_NUMBER o TO\_DATE. El modelo de formato que elija se basa en los elementos de formato mostrados previamente.

El modificador “x” especifica una coincidencia exacta para el argumento de caracteres y el modelo de formato de fecha de una función TO\_DATE:

- La puntuación y el texto entrecomillado del argumento de caracteres deben coincidir exactamente (excepto en las mayúsculas/minúsculas) con las partes correspondientes del modelo de formato.
- El argumento de caracteres no puede tener espacios en blanco adicionales. Sin **fx**, Oracle ignora los espacios en blanco adicionales.
- Los datos numéricos del argumento de caracteres deben tener el mismo número de dígitos que el elemento correspondiente del modelo de formato. Sin **fx**, los números del argumento de caracteres pueden omitir los ceros a la izquierda.



## Funciones TO\_NUMBER y TO\_DATE (continuación)

### Ejemplo

Visualice los nombres y las fechas de contratación de todos los empleados que empezaron a trabajar el 24 de mayo de 1999. Como se utiliza el modificador `fx`, se requiere una coincidencia exacta y los espacios después de la palabra 'May' no se reconocen.

```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date = TO_DATE('May 24, 1999', 'fxMonth DD, YYYY');
```

```
WHERE hire_date = TO_DATE('May 24, 1999', 'fxMonth DD, YYYY')
      *
```

ERROR at line 3:

ORA-01858: a non-numeric character was found where a numeric was expected

## Formato de Fecha RR

Año Actual	Fecha Especificada	Formato RR	Formato 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 el año especificado de dos dígitos es:	
		0–49	50–99
Si los dos dígitos del año actual son:	0–49	La fecha que se devuelve está en el siglo actual.	La fecha que se devuelve está en el siglo anterior al actual.
	50–99	La fecha que se devuelve está en el siglo siguiente al actual.	La fecha que se devuelve está en el siglo actual.

ORACLE

3-42

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### El Elemento de Formato de Fecha RR

El formato de fecha RR es similar al elemento YY, pero se puede utilizar para especificar siglos distintos. Puede utilizar el elemento de formato de fecha RR en lugar del YY, de forma que el siglo del valor de retorno varía en función del año especificado de dos dígitos y los dos últimos dígitos del año actual. La tabla de la transparencia resume el comportamiento del elemento RR.

Año Actual	Fecha Dada	Interpretado (RR)	Interpretado (YY)
1994	27-OCT-95	1995	1995
1994	27-OCT-17	2017	1917
2001	27-OCT-17	2017	2017

## Ejemplo de Formato de Fecha RR

Para buscar empleados contratados antes de 1990, utilice el formato RR, que produce los mismos resultados tanto si se ejecuta el comando en 1999 o ahora:

```
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

ORACLE

3-43

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Ejemplo del Elemento de Formato de Fecha RR

Para buscar empleados contratados antes de 1990, se puede utilizar el formato RR. Como el año es posterior a 1999, el formato RR interpreta la parte de año de la fecha desde 1950 a 1999.

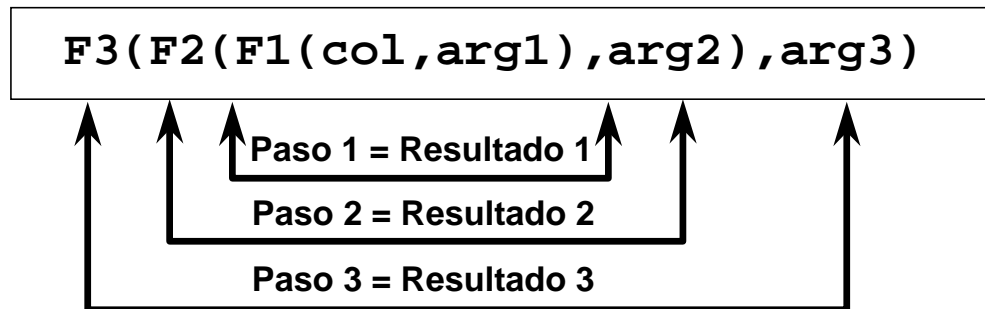
Por otra parte, con el siguiente comando no se selecciona ninguna fila porque el formato YY interpreta la parte de año de la fecha en el siglo actual (2090).

```
SELECT last_name, TO_CHAR(hire_date, 'DD-Mon-yyyy')
FROM employees
WHERE TO_DATE(hire_date, 'DD-Mon-yy') < '01-Jan-1990';
```

no rows selected

## Funciones de Anidamiento

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



ORACLE

### Funciones de Anidamiento

Las funciones de una sola fila se pueden anidar a cualquier profundidad. Las funciones anidadas se evalúan desde el nivel más interno al más externo. A continuación, se indican algunos ejemplos que muestran la flexibilidad de estas funciones.

## Funciones de Anidamiento

```
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

ORACLE

3-45

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Funciones de Anidamiento (continuación)

El ejemplo de la transparencia muestra la cabeza de la compañía, que no tiene director. La evaluación de la sentencia SQL implica dos pasos:

1. Evaluar la función interna que convierte un valor numérico en una cadena de caracteres.
  - Result1 = TO\_CHAR(manager\_id)
2. Evaluar la función externa que sustituye el valor nulo por una cadena de texto.
  - NVL(Result1, 'No Manager')

La expresión completa se convierte en la cabecera de columna al no haberse proporcionado alias de columna.

### Ejemplo

Visualice la fecha del primer viernes transcurridos seis meses desde la fecha de contratación. La fecha resultante debe aparecer como Friday, August 13th, 1999. Ordene los resultados por fecha de contratación.

```
SELECT TO_CHAR(NEXT_DAY(ADD_MONTHS  
                    (hire_date, 6), 'FRIDAY'),  
            'fmDay, Month DDth, YYYY')  
      "Next 6 Month Review"  
FROM   employees  
ORDER BY hire_date;
```

# Funciones Generales

Estas funciones trabajan con cualquier tipo de dato y están relacionadas con el uso de valores nulos.

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

ORACLE

3-46

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Funciones Generales

Estas funciones trabajan con cualquier tipo de dato y están relacionadas con el uso de valores nulos en la lista de expresiones.

Función	Descripción
NVL	Convierte un valor nulo en uno real.
NVL2	Si expr1 no es nulo, NVL2 devuelve expr2. Si expr1 es nulo, NVL2 devuelve expr3. El argumento expr1 puede tener cualquier tipo de dato.
NULLIF	Compara dos expresiones y devuelve un valor nulo si son iguales, o la primera expresión si no lo son.
COALESCE	Devuelve la primera expresión no nula de la lista de expresiones.

**Nota:** Para obtener más información sobre los cientos de funciones disponibles, consulte *Oracle9i SQL Reference*, “Functions”.

# Función NVL

Convierte un valor nulo en un valor real.

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

ORACLE

3-47

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## La Función NVL

Para convertir un valor nulo en uno real, utilice la función NVL.

### Sintaxis

`NVL (expr1, expr2)`

En la sintaxis:

`expr1` es el valor o la expresión de origen que puede contener un valor nulo.

`expr2` es el valor de destino para convertir el valor nulo.

Puede utilizar la función NVL para convertir cualquier tipo de dato, pero el valor de retorno siempre es del mismo tipo que `expr1`.

### Conversiones de NVL para Varios Tipos de Dato

Tipo de Dato	Ejemplo de Conversión
NUMBER	<code>NVL(number_column,9)</code>
DATE	<code>NVL(date_column, '01-JAN-95')</code>
CHAR or VARCHAR2	<code>NVL(character_column, 'Unavailable')</code>

## 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
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

3-48

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### La Función NVL

Para calcular la compensación anual de todos los empleados, debe multiplicar el salario mensual por 12 y sumarle el porcentaje de comisión.

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

LAST_NAME	SALARY	COMMISSION_PCT	AN_SAL
...			
Vargas	2500		
Zlotkey	10500	.2	151200
Abel	11000	.3	171600
Taylor	8600	.2	123840
...			
Gietz	8300		

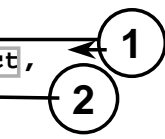
20 rows selected.

Observe que la compensación anual se calcula solamente para los empleados que perciben una comisión. Si es nulo algún valor de columna en una expresión, el resultado será nulo. Para calcular valores para todos los empleados, debe convertir el valor nulo en un número antes de aplicar el operador aritmético. En el ejemplo de la transparencia, se utiliza la función NVL para convertir valores nulos en cero.



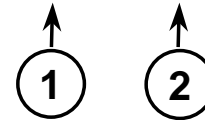
## Uso de la Función NVL2

```
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
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.



ORACLE

### La Función NVL2

La función NVL2 examina la primera expresión. Si no es nula, devuelve la segunda expresión. Si es nula, devuelve la tercera.

#### Sintaxis

`NVL(expr1, expr2, expr3)`

En la sintaxis:

*expr1* es el valor o la expresión de origen que puede contener un valor nulo.

*expr2* es el valor devuelto si *expr1* no es nulo.

*expr3* es el valor devuelto si *expr1* es nulo.

En el ejemplo mostrado, se examina la columna COMMISSION\_PCT. Si se detecta un valor, se devuelve la segunda expresión, SAL+COMM. Si la columna COMMISSION\_PCT contiene un valor nulo, se devuelve la tercera expresión, SAL.

El argumento *expr1* puede tener cualquier tipo de dato. Los argumentos *expr2* y *expr3* pueden tener cualquier tipo de dato excepto LONG. Si los tipos de dato de *expr2* y *expr3* son diferentes, el servidor Oracle convierte *expr3* en el tipo de dato de *expr2* antes de compararlos a menos que *expr3* sea una constante nula, en cuyo caso no es necesaria una conversión del tipo de dato.

El tipo de dato del valor de retorno es siempre el mismo que el de *expr2*, a menos que *expr2* sea un dato de caracteres, en cuyo caso el tipo de dato del valor de retorno es VARCHAR2.

## Uso de la Función NULLIF

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

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.

ORACLE

3-50

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### La Función NULLIF

La función NULLIF compara dos expresiones: si son iguales, la función devuelve nulo; si no lo son devuelve la primera expresión. No se puede especificar el literal NULL para la primera expresión.

#### Sintaxis

```
NULLIF (expr1, expr2)
```

En la sintaxis:

*expr1* es el valor de origen que se compara con *expr2*.

*expr2* es el valor de origen que se compara con *expr1* (si no es igual a *expr1*, se devuelve *expr1*).

En el ejemplo mostrado, se compara el identificador de cargo de la tabla EMPLOYEES con el de la tabla JOB\_HISTORY para cualquier empleado que esté en ambas tablas. La salida muestra el cargo actual de cada empleado. Si el empleado aparece más de una vez, significa que ha tenido anteriormente al menos dos cargos.

**Nota:** La función NULLIF es el equivalente lógico de la siguiente expresión CASE, que se trata más adelante:

```
CASE WHEN expr1 = expr 2 THEN NULL ELSE expr1 END
```

## Uso de la Función COALESCE

- La ventaja de la función COALESCE sobre la función NVL es que puede tomar varios valores alternativos.
- Si la primera expresión no es nula, devuelve dicha expresión; en caso contrario, realiza una fusión (COALESCE) de las expresiones restantes.

ORACLE

3-51

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### La Función COALESCE

La función COALESCE devuelve la primera expresión no nula de la lista.

#### Sintaxis

COALESCE (*expr1*, *expr2*, ... *exprn*)

En la sintaxis:

<i>expr1</i>	devuelve esta expresión si no es nula.
<i>expr2</i>	devuelve esta expresión si la primera es nula y ésta no lo es.
<i>exprn</i>	devuelve esta expresión si las expresiones precedentes son nulas.

## Uso de la Función 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.

ORACLE

### La Función COALESCE

En el ejemplo mostrado, si el valor de COMMISSION\_PCT no es nulo, se muestra. Si es nulo, se muestra SALARY. Si ambos valores son nulos, se muestra el valor 10.

## Expresiones Condicionales

- Proporcionan el uso de la lógica IF-THEN-ELSE dentro de una sentencia SQL.
- Utilizan dos métodos:
  - Expresión CASE
  - Función DECODE

ORACLE

### Expresiones Condicionales

Los dos métodos utilizados para implementar el procesamiento condicional (lógica IF-THEN-ELSE) dentro de una sentencia SQL son la expresión CASE y la función DECODE.

**Nota:** La expresión CASE es nueva en la versión Oracle9i Server y cumple con ANSI SQL; DECODE es específica de la sintaxis de Oracle.

# La 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
```

ORACLE

3-54

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## La Expresión CASE

Las expresiones CASE le permiten utilizar la lógica IF-THEN-ELSE en sentencias SQL sin tener que llamar a procedimientos.

En una expresión CASE sencilla, Oracle busca el primer par WHEN . . . THEN para el que *expr* es igual a *comparison\_expr* y devuelve *return\_expr*. Si ninguno de los pares WHEN . . . THEN cumplen esta condición y existe una cláusula ELSE, Oracle devuelve *else\_expr*. En caso contrario, Oracle devuelve un valor nulo. No puede especificar el literal NULL para todas las expresiones *return\_expr* ni para *else\_expr*.

Todas las expresiones (*expr*, *comparison\_expr* y *return\_expr*) deben tener el mismo tipo de dato, que puede ser CHAR, VARCHAR2, NCHAR o NVARCHAR2.

## 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
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

ORACLE

### Uso de la Expresión CASE

En la sentencia SQL precedente, se descodifica el valor de JOB\_ID. Si JOB\_ID es IT\_PROG, el incremento salarial es del 10 %; si es ST\_CLERK, el incremento es del 15 %; si es SA\_REP, el incremento es del 20 %. Para todos los demás cargos, no hay incremento salarial.

La misma sentencia se puede escribir con la función DECODE.

## La Función DECODE

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

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

ORACLE

3-56

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### La Función DECODE

La función DECODE descodifica una expresión de forma similar a la lógica IF-THEN-ELSE utilizada en varios idiomas. Esta función descodifica *expression* después de compararla con cada valor *search*. Si la expresión es la misma que *search*, se devuelve *result*.

Si el valor por defecto está omitido, se devuelve un valor nulo cuando un valor de búsqueda no coincide con ninguno de los valores resultantes.



## 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
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

ORACLE

3-57

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Uso de la Función DECODE

En la sentencia SQL precedente, se prueba el valor de `JOB_ID`. Si `JOB_ID` es `IT_PROG`, el incremento salarial es del 10 %; si es `ST_CLERK`, el incremento es del 15 %; si es `SA_REP`, el incremento es del 20 %. Para todos los demás cargos, no hay incremento salarial.

La misma sentencia se puede expresar en pseudocódigo como sentencia IF-THEN-ELSE:

```
IF job_id = 'IT_PROG'      THEN salary = salary*1.10  
IF job_id = 'ST_CLERK'    THEN salary = salary*1.15  
IF job_id = 'SA_REP'      THEN salary = salary*1.20  
ELSE salary = salary
```

## Uso de la Función DECODE

**Muestre el tipo impositivo 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;
```

ORACLE

3-58

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Ejemplo

Esta transparencia muestra otro ejemplo que utiliza la función DECODE. En este ejemplo, determinamos el tipo impositivo de cada empleado del departamento 80 en función del salario mensual. Los tipos impositivos son según los valores mencionados a continuación.

<i>Rango de Salario Mensual</i>	<i>Tipo</i>
\$0,00 - 1999,99	00%
\$2.000,00 - 3.999,99	09%
\$4.000,00 - 5.999,99	20%
\$6.000,00 - 7.999,99	30%
\$8.000,00 - 9.999,99	40%
\$10.000,00 - 11.999,99	42%
\$12.200,00 - 13.999,99	44%
\$14.000,00 o superior	45 %

LAST_NAME	SALARY	TAX_RATE
Zlotkey	10500	.42
Abel	11000	.42
Taylor	8600	.4

# Resumen

**En esta lección, debería haber aprendido a:**

- **Realizar cálculos sobre datos utilizando funciones**
- **Modificar elementos de datos individuales utilizando funciones**
- **Manipular la salida para grupos de filas utilizando funciones**
- **Alterar 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**

ORACLE

3-59

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

## Funciones de una Sola Fila

Las funciones de una sola fila se pueden anidar a cualquier nivel. Pueden manipular lo siguiente:

- Datos de caracteres: LOWER, UPPER, INITCAP, CONCAT, SUBSTR, INSTR, LENGTH.
- Datos numéricos: ROUND, TRUNC, MOD.
- Datos de fecha: MONTHS\_BETWEEN, ADD\_MONTHS, NEXT\_DAY, LAST\_DAY, ROUND, TRUNC.
- Los valores de fecha también puede utilizar operadores aritméticos.
- Las funciones de conversión pueden convertir valores de caracteres, de fecha y numéricos: TO\_CHAR, TO\_DATE, TO\_NUMBER.
- Hay varias funciones que están relacionadas con valores nulos, como NVL, NVL2, NULLIF y COALESCE.
- Se puede aplicar la lógica IF-THEN-ELSE dentro de una sentencia SQL utilizando la expresión CASE o la función DECODE.

## SYSDATE y DUAL

SYSDATE es una función de fecha que devuelve la fecha y la hora actuales. Lo habitual es seleccionar SYSDATE desde una tabla ficticia llamada DUAL.

## Práctica 3, Parte Dos: Visión General

Esta práctica cubre los siguientes temas:

- Creación de consultas que requieran el uso de funciones numéricas, de caracteres y de fecha
- Uso de la concatenación con funciones
- Escritura de consultas sensibles a mayúsculas/minúsculas para probar la utilidad de las funciones de caracteres
- Realización de cálculos de meses y años de servicio para un empleado
- Determinación de la fecha de revisión para un empleado

ORACLE

3-60

Copyright © Oracle Corporation, 2001. Todos los derechos reservados.

### Práctica 3, Parte Dos: Visión General

Esta práctica se ha diseñado para ofrecerle diversos ejercicios que utilicen las distintas funciones disponibles para tipos de dato de caracteres, numéricos y de fecha.

Recuerde que, para funciones anidadas, los resultados se evalúan desde la función más interna hasta la más externa.

### Práctica 3, Parte Uno (continuación)

1. Escriba una consulta para mostrar la fecha actual. Etiquete la columna como Date.

Date
28-SEP-01

2. Para cada empleado, visualice su número, apellido, salario y salario incrementado en el 15 % y expresado como número entero. Etiquete la columna como New Salary. Ponga la sentencia SQL en un archivo de texto llamado lab3\_2.sql.
3. Ejecute la consulta en el archivo lab3\_2.sql.

EMPLOYEE_ID	LAST_NAME	SALARY	New Salary
100	King	24000	27600
101	Kochhar	17000	19550
102	De Haan	17000	19550
103	Hunold	9000	10350
...			
202	Fay	6000	6900
205	Higgins	12000	13800
206	Gietz	8300	9545

20 rows selected.

4. Modifique la consulta lab3\_2.sql para agregar una columna que reste el salario antiguo del nuevo. Etiquete la columna como Increase. Guarde el contenido del archivo como lab3\_4.sql. Ejecute la consulta revisada.

EMPLOYEE_ID	LAST_NAME	SALARY	New Salary	Increase
100	King	24000	27600	3600
101	Kochhar	17000	19550	2550
102	De Haan	17000	19550	2550
103	Hunold	9000	10350	1350
104	Ernst	6000	6900	900
107	Lorentz	4200	4830	630
124	Mourgos	5800	6670	870
141	Rajs	3500	4025	525
142	Davies	3100	3565	465
143	Matos	2600	2990	390
...				
201	Hartstein	13000	14950	1950
202	Fay	6000	6900	900
205	Higgins	12000	13800	1800
206	Gietz	8300	9545	1245

20 rows selected.

### Práctica 3, Parte Uno (continuación)

5. Escriba una consulta que muestre los apellidos de los empleados con la primera letra en mayúsculas y todas las demás en minúsculas, así como la longitud de los nombres, para todos los empleados cuyos nombres comienzan por J, A o M. Asigne a cada columna la etiqueta correspondiente. Ordene los resultados según los apellidos de los empleados.

Name	Length
Abel	4
Matos	5
Mourgos	7

### Práctica 3, Parte Dos

6. Para cada empleado, muestre su apellido y calcule el número de meses entre el día de hoy y la fecha de contratación. Etiquete la columna como MONTHS\_WORKED. Ordene los resultados según el número de meses trabajados. Redondee el número de meses hacia arriba hasta el número entero más próximo.

**Nota:** Los resultados que obtenga diferirán.

LAST_NAME	MONTHS_WORKED
Zlotkey	20
Mourgos	22
Grant	28
Lorentz	32
Vargas	39
Taylor	42
Matos	42
Fay	49
Davies	56
Abel	65
Hartstein	67
Rajs	71
Higgins	88
Gietz	88
LAST_NAME	MONTHS_WORKED
De Haan	105
Ernst	124
Hunold	141
Kochhar	144
Whalen	168
King	171

20 rows selected.

### Práctica 3, Parte Dos (continuación)

7. Escriba una consulta que produzca lo siguiente para cada empleado:  
`<employee last name> earns <salary> monthly but wants <3 times salary>`. Etiquete la columna como `Dream Salaries`.

Dream Salaries
King earns \$24,000.00 monthly but wants \$72,000.00.
Kochhar earns \$17,000.00 monthly but wants \$51,000.00.
De Haan earns \$17,000.00 monthly but wants \$51,000.00.
Hunold earns \$9,000.00 monthly but wants \$27,000.00.
Ernst earns \$6,000.00 monthly but wants \$18,000.00.
Lorentz earns \$4,200.00 monthly but wants \$12,600.00.
Mourgos earns \$5,800.00 monthly but wants \$17,400.00.
Rajs earns \$3,500.00 monthly but wants \$10,500.00.
Davies earns \$3,100.00 monthly but wants \$9,300.00.
Matos earns \$2,600.00 monthly but wants \$7,800.00.
Vargas earns \$2,500.00 monthly but wants \$7,500.00.
■ ■ ■
Gietz earns \$8,300.00 monthly but wants \$24,900.00.

20 rows selected.

Si dispone de más tiempo, realice los siguientes ejercicios:

8. Cree una consulta para mostrar el apellido y el salario de todos los empleados. Formatee el salario para que tenga 15 caracteres, rellenando a la izquierda con \$. Etiquete la columna como `SALARY`.

LAST_NAME	SALARY
King	\$\$\$\$\$\$\$\$\$24000
Kochhar	\$\$\$\$\$\$\$\$\$17000
De Haan	\$\$\$\$\$\$\$\$\$17000
Hunold	\$\$\$\$\$\$\$\$\$9000
Ernst	\$\$\$\$\$\$\$\$\$6000
Lorentz	\$\$\$\$\$\$\$\$\$4200
Mourgos	\$\$\$\$\$\$\$\$\$5800
Rajs	\$\$\$\$\$\$\$\$\$3500
■ ■ ■	
Higgins	\$\$\$\$\$\$\$\$\$12000
Gietz	\$\$\$\$\$\$\$\$\$8300

20 rows selected.



### Práctica 3, Parte Dos (continuación)

9. Muestre el apellido de cada empleado, así como la fecha de contratación y la fecha de revisión de salario, que es el primer lunes después de cada seis meses de servicio. Etiquete la columna REVIEW. Formatee las fechas para que aparezca en un formato similar a “Monday, the Thirty-First of July, 2000”.

LAST_NAME	HIRE_DATE	REVIEW
King	17-JUN-87	Monday, the Twenty-First of December, 1987
Kochhar	21-SEP-89	Monday, the Twenty-Sixth of March, 1990
De Haan	13-JAN-93	Monday, the Nineteenth of July, 1993
Hunold	03-JAN-90	Monday, the Ninth of July, 1990
Ernst	21-MAY-91	Monday, the Twenty-Fifth of November, 1991
Lorentz	07-FEB-99	Monday, the Ninth of August, 1999
Mourgos	16-NOV-99	Monday, the Twenty-Second of May, 2000
Rajs	17-OCT-95	Monday, the Twenty-Second of April, 1996
Davies	29-JAN-97	Monday, the Fourth of August, 1997
■ ■ ■		
Gietz	07-JUN-94	Monday, the Twelfth of December, 1994

20 rows selected.

10. Muestre el apellido, la fecha de contratación y el día de la semana en el que comenzó el empleado. Etiquete la columna DAY. Ordene los resultados por día de la semana, comenzando por el lunes.

LAST_NAME	HIRE_DATE	DAY
Grant	24-MAY-99	MONDAY
Ernst	21-MAY-91	TUESDAY
Mourgos	16-NOV-99	TUESDAY
Taylor	24-MAR-98	TUESDAY
Rajs	17-OCT-95	TUESDAY
Gietz	07-JUN-94	TUESDAY
Higgins	07-JUN-94	TUESDAY
King	17-JUN-87	WEDNESDAY
De Haan	13-JAN-93	WEDNESDAY
■ ■ ■		
Abel	11-MAY-96	SATURDAY
Lorentz	07-FEB-99	SUNDAY
Fay	17-AUG-97	SUNDAY
Matos	15-MAR-98	SUNDAY

20 rows selected.

### Práctica 3, Parte Dos (continuación)

Si desea una comprobación adicional, complete los siguientes ejercicios:

11. Cree una consulta que muestre el apellido y las comisiones de los empleados. Si un empleado no gana comisión, ponga "No Commission". Etiquete la columna COMM.

LAST_NAME	COMM
King	No Commission
Kochhar	No Commission
De Haan	No Commission
Hunold	No Commission
Ernst	No Commission
Lorentz	No Commission
Mourgos	No Commission
Rajs	No Commission
Davies	No Commission
Matos	No Commission
Vargas	No Commission
Zlotkey	.2
Abel	.3
Taylor	.2
■ ■ ■	
Gietz	No Commission

20 rows selected.

12. Cree una consulta que muestre el apellido de los empleados y que indique las cantidades de sus salarios anuales con asteriscos. Cada asterisco significa mil dólares. Ordene los datos por salario en orden descendente. Etiquete la columna EMPLOYEES\_AND\_THEIR\_SALARIES.

EMPLOYEE_AND_THEIR_SALARIES	
King	*****
Kochhar	*****
De Haan	*****
Hartstei	*****
Higgins	*****
Abel	*****
■ ■ ■	
Vargas	**

20 rows selected.

### Práctica 3, Parte Dos (continuación)

13. Utilizando la función DECODE, escriba una consulta que muestre el grado de todos los empleados basándose en el valor de la columna JOB\_ID, según los datos siguientes:

<i>Cargo</i>	<i>Grado</i>
AD_PRES	A
ST_MAN	B
IT_PROG	C
SA_REP	D
ST_CLERK	E
Ninguno de los anterior	0

JOB_ID	G
AD_PRES	A
AD_VP	0
AD_VP	0
IT_PROG	C
IT_PROG	C
IT_PROG	C
ST_MAN	B
ST_CLERK	E
ST_CLERK	E
ST_CLERK	E
■ ■ ■	
AC_MGR	0
AC_ACCOUNT	0

20 rows selected.

14. Vuelva a escribir la sentencia de la pregunta anterior utilizando la sintaxis CASE.

